Chapter 1

DISTRIBUTED ARCHITECTURES FOR WEB CONTENT ADAPTATION AND DELIVERY

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

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

Philip S. Yu
IBM T.J. Watson Research center
psyu@us.ibm.com

1. Introduction

The success of the Web in the last decade has caused an evolution of the distributed resources. The initial static contents are now enriched by complex contents, with an increasing amount of multimedia resources and of dynamically generated contents. This evolution has shifted the research focus from Web *content delivery* that is, a mature technology, to *service delivery* that introduces new scalability and performance problems to the hardware/software infrastructure that has to generate and distribute content.

In the last years, the overall complexity of the service delivery scenario is further increased by the so called *ubiquitous Web access.* The challenge is to allow access to the Web-based services by any user, from any location through every device, ranging from wired desktop PCs to Internet-enabled mobile phones [Vanderheiden, 1997, Saha and Mukherjee, 2003]. To enable ubiquitous Web accesses, the software infrastructure must support a new set of *adaptation services.* They include a wide spectrum of services that may require complex interactions with different applications and databases. We find convenient to distinguish two main categories of adaptation services that is,

- *Transcoding.* They are services that tailor Web content to the capabilities of the client device and of the network connection. A typical example is to reduce the definition of an image with the goal of reducing the size of the file to be delivered.

- *Personalization.* They include more sophisticated services that are oriented to adapt the content to (a combination of) user preferences, locations and behaviors. An example of personalization is the insertion of banners tailored to the user gender, country and preferences in the visible pages of a user. The clear difference with the previous services is that the storage and maintenance of an updated archive of user profiles is required.

We will provide a more accurate taxonomy of adaptation services in Section 3.

Adaptation and delivery services enabling ubiquitous Web access are deployed by introducing new technologies on top of pre-existing Internet services. However, the final result aims to be much richer than the simple sum of the individual components.

From a technological point of view, ubiquitous Web access exacerbates the scalability and performance issues that characterize present Web-based services because adaptation services have much higher computing and storage requirements. The heterogeneity and richness of present and future adaptation services increase the complexity of the hardware/software system that must support them, with immediate consequences on the scalability of present infrastructures that support delivery and adaptation services [Canali et al., 2003]. To investigate architectural solutions for efficient content generation, adaptation and delivery through the Internet is the main goal of this chapter that is organized in five sections.

In Section 2, we outline the main services that must be provided by the infrastructure for the ubiquitous Web and some architectural directions that seem more promising. In Section 3, we classify the main adaptation services. This taxonomy is important to describe the actual distributed infrastructures that can provide an efficient service and that are described in Section 4. In Section 5, we conclude the chapter with some final remarks and notes on open issues.

2. Infrastructures for supporting ubiquitous Web access

The directions to address scalability and performance issues are well known: caching and replication. However, it is important to anticipate how we can apply these possible solutions to the world of content generation and delivery.

- *Replication of system resources.* System resources include hardware resources (i.e., computing power, storage capacity, network bandwidth), and system software, such as operating system and HTTP servers. Hardware and system software replication can be required due to the possible exhaustion of hardware and software resources, such as memory, file descriptors and process identifiers.

- *Replication of content generators.* In this approach, we consider the possibility of completely replicating the sources of the Web resources that is, static contents but also Web applications and databases that are necessary to generate dynamic contents.

- *Replication of Web resources.* This is commonly known as a *caching* technique and includes static resources that is, files that are static in nature or previously generated resources.

Replication of system resources helps addressing performance and scalability issues by augmenting the power of the infrastructure that must perform generation, adaptation and delivery tasks. Replication of content generators allows us to fully exploit the advantages of replication of systems resources by creating independent replicated systems that provide complete functions of adaptation and delivery services. Caching allows to take advantage from already adapted Web resources by leveraging local and temporal locality of client requests. Caching aims to reduce the response time in a twofold way: it limits the amount of adaptations that often are CPU bound operations; when the cache server node is placed in convenient locations, it reduces the delivery time because the distance from the client and the content repository is closer.

In Section 4 we give major details about the ways in which replication of system resources, replication of content generators and caching can improve delivery and adaptation services. By now, it is important to anticipate that we should also consider the space dimension for the previous solutions. The replication scale can go from a LAN to a WAN scale. In a local replication of system resources and content generators the nodes are tightly connected. They are placed on the same LAN and usually share a single upstream link connecting the system to the rest of the Internet. The common term to describe such a system is *cluster*. Nodes within a cluster provide increased computing power because of the replication of system resources. They can interact in a fast and effective way [Cardellini et al., 2002]; and replication tends to improve fault tolerance because a faulty node can be easily bypassed.

WAN-based replication is a geographical replication of system resources that are distributed among multiple Autonomous Systems. These systems address scalability issues related mainly to network congestion in peering points and WAN links. We usually refer to WAN-based replicates systems as *geographically distributed systems*. The replicated resources are typically clusters of nodes. Indeed, building a geographic infrastructure is an expensive task and it would be meaningless to replicate single nodes that have limited computational power, storage capacity and are not fault tolerant.

The combination of replication and caching allows a large set of design options to deploy generation, adaptation and delivery of Web-based services through LAN/WAN distributed architectures. To this purpose, it is important to distinguish the following *servers* that may mapped on the nodes of the infrastructure.

- *Content generators* provide Web contents and services either by storing or by dynamically generating them.
- *Adapters* provide adaptation services that is, they have to classify a client request, to define the necessary adaptation operations to be performed, and to carry out the consequent transcoding and/or personalization function(s).
- *Cache servers* provide partial content repositories. Their main function is to store already generated and possibly adapted contents and to retrieve them. If a client request can be satisfied through a previously adapted and cached content, system performance is improved because the client does not have to wait for the completion of adaptation operations. Even better, if the response can be got from a node that is closer to the client.

In the design of infrastructures for efficient adaptation and delivery services we have multiple choices that depend on how the servers are distributed over the infrastructure for generation, adaptation and delivery. The number of possible combinations is high and we will provide a more detailed discussion on these architectures in Section 4. By now, we can anticipate on outline of the four classes of architectures that we consider most appropriate for providing efficient generation and delivery of Web-based adaptation services. It is worth to anticipate that hybrid architectures may exist as well, but the most important solutions come from these basic *building blocks.*

- *Cluster-based system.*
- *Cluster integrated with geographic replication of cache server nodes.*
- *Cluster integrated with geographic replication of adapters and cache servers.*
- *Geographic replication of clusters* (multi-clusters).

Let us describe the main characteristics of the four basic architectures. In the related four figures (from Figure 1.1 to Figure 1.4), we use the following notation. Each node is represented through a symbol that denotes the server it runs: white diamonds are content generators, black- and white diamonds are cache servers, and circles are adapters. The box enclosing the nodes represents the cluster border. The big dashed circle denotes the Internet edge.

Cluster-based system. In this architecture all servers are mapped within one cluster of nodes. No other function related to generation, adaptation and caching of content for this Web site is delegated to nodes outside the cluster. In most cases, the nodes of the cluster are differentiated according to a multi-tier scheme: a first layer of nodes provides partial content repository (such nodes are also called HTTP accelerators [Cardellini et al., 2002]), a second layer executes adaptation servers, and a third layer (back-end) is devoted to the generation of dynamic contents. Figure 1.1 gives an abstract view of this cluster-based architecture.

Cluster integrated with geographic replication of cache server nodes. This architecture consists of a cluster of nodes, providing all functions content generation, local caching and adaptation, that is enriched by a set of cache server nodes that are distributed in different Autonomous Systems with the purpose of being closer to the clients. (For this reason, they have often been called edge nodes.) Figure 1.2 shows this architecture. The origin cluster is at the center of the infrastructure and is surrounded by a large number of nodes that execute cache servers hopefully in locations that are closer to the clients.

Cluster integrated with geographic replication of adapters and cache servers. This infrastructure is similar to the previously proposed scheme, but the geographically distributed nodes execute both adaptation and caching [Maheshwari et al., 2002]. The goal is to limit the load and the traffic reaching the cluster. The higher scalability of this scheme is obtained by increasing the complexity of the adaptation process. Indeed, as we

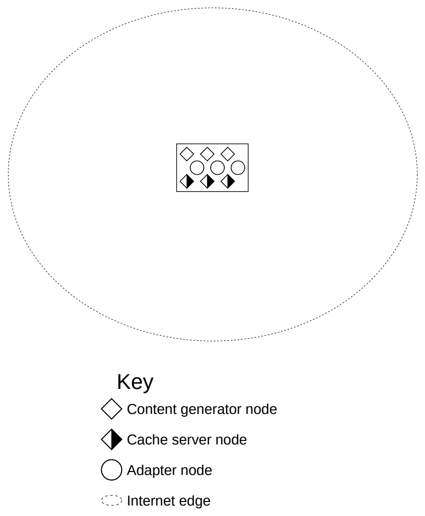

Figure 1.1. Cluster-based architecture

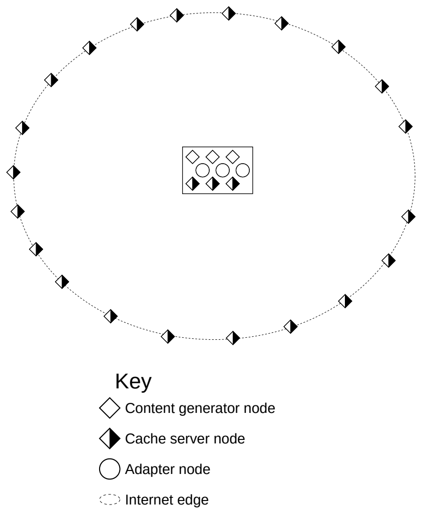

Figure 1.2. Cluster integrated with geographic replication of cache server nodes

will see in greater detail in Section 4, complex adaptation services involving personalization can require interaction with content generator servers. Enabling interaction between adapters and content generators while preserving data security and consistency requires considerable additional efforts. Figure 1.3 shows the geographic replication of adapters and cache server. The overall organization of the architecture is similar to Figure 1.2. The difference is the presence of clusters composed by nodes acting as adapters and cache servers.

Geographic replication of clusters (multi-clusters). In this architecture we replicate and distribute geographically the clusters. They are typically mirrors that are organized with a *primary* cluster with a master role and multiple *secondary* clusters. The number of clusters is limited to some units, because of the difficulties of managing complex systems distributed over the world and of preserving consistency of mirrored resources. The issues related to data security and consistency are further increased because of the need to guarantee interactions among the clusters. This architecture offers great scalability and performance because every element that can become a bottleneck is replicated.

Performance of a multi-cluster architecture can be further improved by enriching it through a large set of geographically distributed cache servers and adapters, as illustrated in Figure 1.4.

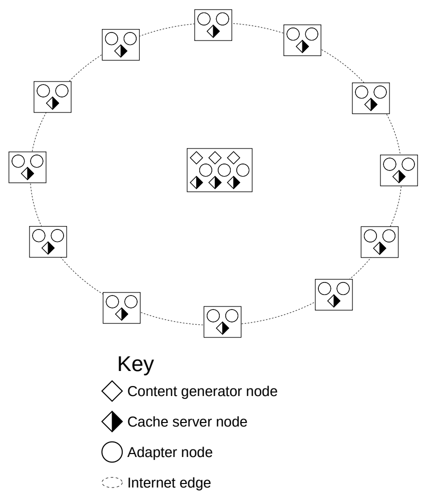

Figure 1.3. Cluster integrated with geographic replication of adapters and cache servers

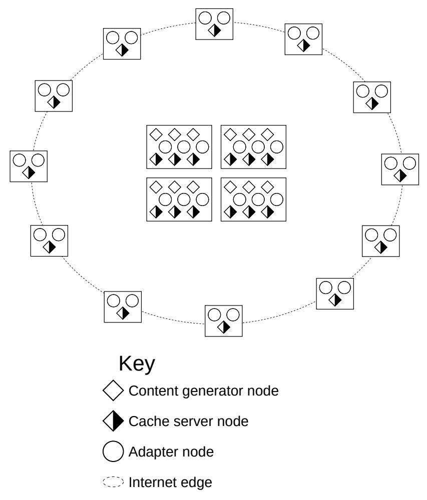

Figure 1.4. Geographic replication of clusters (multi-clusters)

3. Adaptation services

The advent of the ubiquitous Web requires on-the-fly transformation of Web-based resources possibly obtained through complex operations, and their delivery to diverse destination devices. The *adaptation service* term spans various types of functionalities, that for the goal of this chapter focused on system infrastructures, we classified in two main categories, each with two sub-categories. *Personalization* services adapt the resources to the user characteristics. We will see that it is quite important also to distinguish whether a personalization function requires some sort of stored information to be completed or not. *Transcoding* services are typically less sophisticated than personalization services. They adapt the content by taking into account the characteristics related to the user device and/or those related to the client interconnection. Figure 1.5 summarizes these categories by considering the implicit/explicit promoter of the adaptation service, i.e., the user or the client device.

Both personalization and transcoding services can be applied to static and dynamically generated content. Hence, a simple response to an explicit query to a database accessible via Web that does not include some additional adaptation service is not considered in the personalization category, even if the delivered resource is specific for that user.

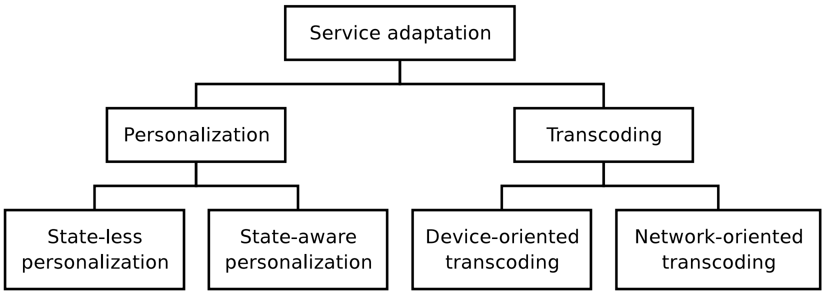
Service adaptation
Personalization
Transcoding
State-less personalization
State-aware personalization
Device-oriented transcoding
Network-oriented transcoding

Figure 1.5. Taxonomy of adaptation services

3.1 Personalization services

A personalized service implies that some additional information is added to the user request for the generation of the resource to be delivered. There are several ways to transfer this information about user preferences. For example, it may be explicitly communicated by the user through a form, or by the client device together with the request (e.g., by means of cookies or HTTP headers). Alternatively, this additional information may be inferred by the present and/or past user behavior or the system infrastructure may implicitly get it from the client request (e.g., user location, connection protocol). We prefer to distinguish personalization services into two subclasses based on the characteristics of the information that is used for personalization, because the two subclasses introduces different issues to be addressed by the infrastructure for content generation, adaptation and delivery.

- *State-less personalization.* It basically refers to the adaptation services that extract user preferences from the user or client request without combining them with previously stored information. Some (not exhaustive) examples of these personalization services include:
  - *Language translation.* It allows the translation of textual information according to user preferences. The user communicates its preferred language and, if the language is not within the set of available texts, the infrastructure executes a run-time translation process.
  - *Virus scanning.* It protects the user against the download of resources (executable files, documents with scripting extensions, and archives) containing harmful viral code, such as viruses or macro-viruses. The requested resource is scanned before its delivery. This process looks for known virus sig-

natures and, in case of infected files, it stops the resource download or sanitizes the resource.

  - *Insertion of random banners.* The requested resource is integrated with some ads that are randomly generated links from a pre-selected list of banners. More sophisticated banner insertion techniques take into account dynamic or pre-registered user preferences, but this case falls in the following category.

- *State-aware personalization.* It refers to personalization services that are based on some previously stored information. This information is typically contained in some database(s) and can be extracted as a consequence of explicit information coming from the user request or inferred through the run-time or off-line analysis of the user behavior (e.g., through data mining on log files of a Web site). The use of stored information allows the infrastructure to deploy all the above state-less operations, but also more complex personalization services, such as:

  - Personalization based on previously registered user profile. It represents a wide category of personalization services ranging from user filled-in forms to explicit subscription to news feeds (i.e., a personalized portal page is composed by aggregating information from different and heterogeneous sources such as XML-RSS news feeds), to request filtering that protects from harmful or unwanted content. The user profile is typically stored in the infrastructure of the service provider and is used to direct adaptation.

  - Cryptographic signature and encoding of multimedia documents. This service can be at the basis of a Digital Right Management service [LaMacchia, 2002].

  - Adaptation to the user navigation style. It means that information is dynamically rearranged to help the user reaching the information of interest as soon as possible. Analysis of user click history allows the infrastructure to identify typical behavior patterns such as paths in the navigation graph that leads from a page towards another through a specific sequence of links. Once a known pattern is recognized, the service delivery infrastructure can open directly the target page, thus saving some clicks to the user.

  - Adaptation to the user interests. It allows the system to figure out the preferences of the user and to tune services, such as content presentation and ad banner insertion, according to these inferred or specified interests.

- Location and surrounding-based services that achieve personalization on the basis of the user geographic location. The user position is compared with geographic data managed by the service provider and the generation and delivery of static and dynamic content (e.g., queries) is carried out according to the user location and surrounding, possibly combined with user preferences. There are myriads of possible applications for the mobile users ranging from tourist information, commercial information, meetings, blind dates, banner insertion, advertisements. If you do not care much about privacy issues, you can share the opinion that this service may represent the killer application for the ubiquitous Web.

It is worth to observe that the indicated adaptation services are not mutually exclusive. They can be combined to form a chain of adaptation services that must be completed before the resource delivery.

3.2 Transcoding services

The so called pervasive computing devices that characterize the ubiquitous Web show an enormous variability in processing power, storage, display, and connectivity capabilities. The main consequence is that Web resources (including video, image, audio and text) may need to be summarized, translated and converted because of two main requirements related to the client platform, i.e., the features of the physical devices and the characteristics of the connection to Internet. Hence we distinguish the following transcoding services that imply a different dynamic behavior and different architectural needs.

- *Device-oriented transcoding* that is mainly related to the characteristics of the user device interface. The goal is to provide the user with the best quality for content access and Web services that are compatible with the physical capabilities of the device.
- *Network-oriented transcoding* that is mainly due to the medium and protocol characteristics of the connections of the devices to the Internet. As network-oriented transcoding should take into account network link status, the deployment of those services requires the adaptation and delivery service infrastructure to be integrated with a distributed network monitoring system.

It is worth to note that the previous classification does not allow us to partition the transcoding services, because adaptation to device capabilities can impact network resource usage and adaptation to network features may impact the Web resource representation. Due to the heterogeneity of devices, services and network protocols, it is necessary to find techniques and innovative systems that adapts (customizes) the same multimedia content and/or Web service to different client devices, still preserving its semantic attributes.

The conversion task is implemented by software applications, called transcoders, which are able to filter and convert information. A video transcoder can resize, change color palette, format and compression type (e.g., from GIF to JPEG or from AVI to MPEG), transmit only selected spots, single frames or part of them. A text transcoder can summarize the textual content, apply various style sheets to XML documents, and transform HTML objects into WML objects for wireless devices. Intelligent transcoding systems can extract the most meaningful information from a video, and compress data while maintaining its semantic value (for example, by keeping all details on moving objects and limiting background information, according to the MPEG4 and MPEG7 standards). A content filter may discard images or transcode them in textual description if the information has to be delivered to a device that can only visualize text, it may eliminate or compress all the objects larger than a given size for low bandwidth connections and so on.

If we look at the content, we can further distinguish two main classes of transformations for both device- and network-oriented transcoding:

- "within" media types (e.g., changing size and color depth of an image or converting from JPEG to GIF format);
- "between" media types (e.g., from speech to text, from video item to a set of images).

In addition, network-oriented transcoding may be characterized by more dynamic requirements, such as the necessity of

- tunneling/conversion of HTTP protocol over/to different protocol (e.g., to exploit the better characteristics of WAP in the case of wireless links);
- adapting object conversion parameters (e.g., quality factor of JPG images) according to network capability of the device and to the network status [Chandra et al., 2000];
- adapting conversion semantics to the connection characteristics (e.g., changing from a video stream to a sequence of images when an UMTS smartphone switches to a GPRS connection).

4. Distributed architectures for adaptation and delivery services

In section 2 we have identified four basic architectures that can be used to create an efficient infrastructure for content generation, adaptation and delivery services, namely: *cluster-based system*, *cluster integrated with geographic replication of cache server nodes*, *cluster integrated with geographic replication of adapters and cache servers*, *geographic replication of clusters* (multi-clusters).

Although the considered infrastructures are not the only viable solutions, they represent different trends that allow us to discuss a large spectrum of related issues. In particular, the goal of this section is to investigate which adaptation service can be successfully deployed by each of the four architectures.

4.1 Cluster-based system

The first solution to address the performance issues related to adaptation and delivery services is the replication of computing resources over a local area. A LAN-based system (usually called *cluster*) is a tightly coupled architecture providing a single system interface to the clients. These clusters are typically organized as multi-tier systems, where the front-end tier is often represented by a Web switch that dispatches the client requests among a layer of HTTP servers acting as cache servers that services requests when no adaptation is required. A second layer is composed of adapters that provide transcoding, state-aware and state-less personalization for Web content and services. In the back-end tier, they typically have the content generators that provide the original version of the requested information. These resources may be stored in file systems or dynamically generated through some database servers. Other databases can store information that is useful for some adaptation service and is necessary for state-aware personalization. We reconsider the cluster-based architecture described in Figure 1.1 by highlighting in Figure 1.6 the different types of adaptation: the “T” and the “P” inside the adapter circle denotes transcoding and state-less personalization, respectively. The state-aware personalization that typically requires some previously stored information is evidenced by adding a database symbol near the circle.

This figure also shows the steps to service a client request through the cluster-based architecture: a solid black line represents the client request, a dashed line is the interaction between an adapter and the back-end tier of the cluster, and the black square represents the client. Client requests received by the cluster are analyzed and classified to determine what

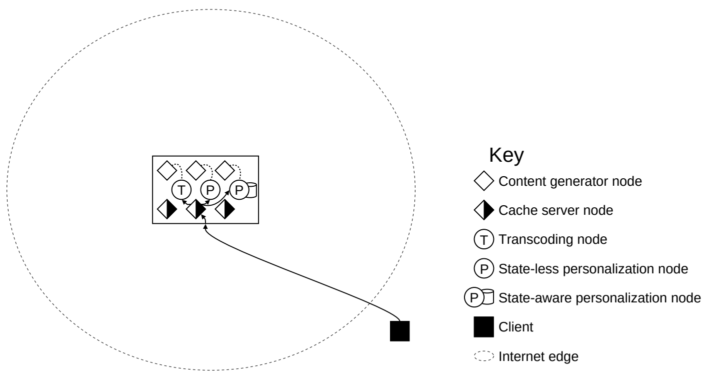

Figure 1.6. Adaptation services implemented in a cluster-based system

adaptation is required, if any. This fist step determines if the request can be satisfied by means of Web resources that are stored in the cache servers or if it is necessary to carry out some adaptation operation(s). The former case occurs when no adaptation is required or when the cache servers store a suitable and already adapted Web resources. If adaptation is required, the request is forwarded to the adapter layers. These servers carry out the adaptation tasks by possibly interacting with content generators and user profile information. It is quite important to observe that the cluster-based architecture allows the deployment of any adaptation task: transcoding and state-less personalization are easily carried out because they only require the information already extracted from the client request. On the other hand, state-aware personalization can be carried out without any problem because user profiles and other sensitive information are stored in the databases of the cluster nodes that are easily accessible by the adapters.

4.2 Clusters integrated with geographically distributed cache servers

LAN-based systems have many pros, but they have scalability problems related to efficient generation, adaptation and delivery of resources when the Web site is highly popular. The first problem that affects replication on a local scale is the so called first mile, i.e., the network link connecting the cluster to the Internet. This link can represent the system bottleneck for the end-to-end performance, moreover it is a potential single point of failure. Traffic on the Web-based cluster zone, failures on an external router, and denial-of-service attacks may cause the service to be unreachable independently of the computational power of the provider platform. When better scalability and performance are needed, it is useful to replicate some elements of the infrastructure over a geographic scale.

The simplest solution towards geographic replication is to distribute the function of cache servers. To improve performance, the nodes hosting cache servers should be placed as close as possible to the clients. The goal is to reduce the response latency and the number of possible bottlenecks between the client and the cache server node. For this reason cache servers are usually located on the *network edge*, for example within the networks of large organizations, such as ISP, providing Internet access to users. Cache servers must also intercept client requests. Multiple techniques are available to this purpose. For example, client devices can be explicitly configured to use cache servers as a proxy, or the cache server layer can rely on traffic redirection mechanisms. When cache servers are on the same network of the Internet access point used by the clients, network-level packet interception is the most popular solution (this approach is called transparent proxy [Wessels, 2001]). When transparent proxies are impracticable, the best alternative is to use DNS-based redirection [Cardellini et al., 2003]. This approach, widely used in CDNs [Rabinovich and Spatscheck, 2002], is based on modified DNS-servers that resolve queries for the site hostname with the IP address of a suitable cache server (the algorithm used to detect the most suitable cache server is usually complex and takes into account geographic and network distance, network link and cache server status).

Figure 1.7 describes an infrastructure for content generation, adaptation and delivery that is based on a cluster and a geographic replication of cache servers. This figure also shows how client requests are handled by the infrastructure: each request first reaches a cache server node that checks if it holds a valid resource to satisfy that request. In the positive case, it services the request without interacting with the cluster. In the case of miss, the cache server typically acts as a client by issuing a request to the cluster (black arrows in Figure 1.7) that hosts all necessary servers and resources.

As a third alternative, it is possible to implement some sort of cooperation among the cache servers. In this case, a miss does not cause an immediate request to the cluster, but it activates a lookup phase to check whether a (nearby) cache server holds a suitable Web resource (dashed arrows). Several strategies are proposed in the field of cooperative Web caching, such as ICP, Cache Digest, Summary Cache, etc. [Wessels, 2001, Davison, 2001], but they are out of the scope of this chapter. From the point of view of adaptation services that is the focus of

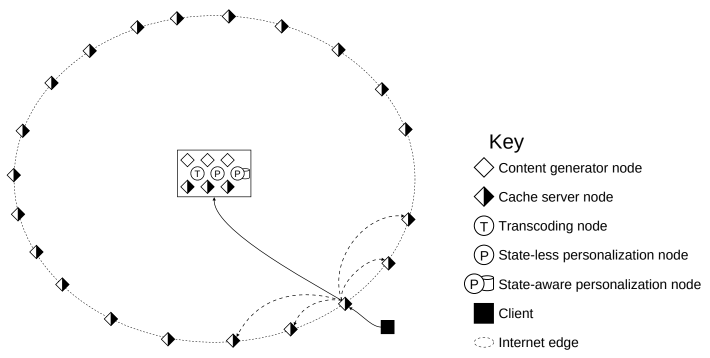

Figure 1.7. Adaptation services implemented by a cluster integrated with geographically distributed cache servers

this chapter, there is no difference with the pure cluster-based solution, because all adaptation services are still provided by the cluster.

On the other hand, the geographically distributed nature of this architecture introduces novel management issues. While it is rather easy to think that a cluster is managed by the same content/service provider (if we exclude housing solutions), the deployment and management of architectures distributed over different Autonomous Systems require large investments in structures and technical knowledge. Only a few restricted number of companies may be able to afford this investment. Hence, we can identify three main alternatives that are characterized by the entity that manages the geographically distributed nodes.

- In the *content/service provider-based* solution, the provider deploys and manages this distributed infrastructure of cache servers by using internal resources. In this case, the cache servers are often called *reverse proxies* because the provider can drive the content of its cache servers through some *push-based* mechanism. This allows a high control on which resources are stored in each cache server and also permits the use of cache content invalidation protocols that improve consistency of distributed information.

- On the opposite hand, the provider may refer to a third party entity that has no direct relationship with it. The customers of the third party infrastructure are the Web users. No additional services are offered to the content/service provider. A similar infrastructure of distributed cache servers works for all Web resources, but it does not have any control on the resource location/placement, because a pull-based mechanism populates the cache servers. This *third party independent* infrastructure can be supported by a public organization or by an independent ISP that offers a Web caching service to its customers.

- In the middle between the previous two extremes, it is possible to refer to a third party entity that works in strict relationship with the infrastructure of the service provider. The third party infrastructure provides cache services on behalf of their customer providers. This *third party custom* solution is the core business of CDN companies, such as Akamai [Akamai Inc., 2004] and Speedera [Speedera Networks inc, 2004]. A similar infrastructure has two big advantages with respect to the independent third party: its working set is limited to the resources of its customer providers; it can use some push-based mechanism as in the provider-based solution due to the strict relationship between the provider and the third party operating the distributed infrastructure of cache server nodes.

4.3 Clusters integrated with geographically distributed cache and adaptation servers

The next evolutionary step towards better performance is to replicate also the adapter servers on nodes that are close to the network edge. This architecture is shown in Figure 1.8: it has a central cluster providing all generation, adaptation and caching services, and geographically distributed edge nodes hosting cache and adaptation servers. A client request is processed by cache servers as in the previous architecture with a main difference. In the previous case, the cache server could fulfill a client request only if they held an exact copy of the requested resource. On the other hand, in this case an edge node experiences an hit that does not require an interaction with the cluster even when it has a valid copy that can be adapted locally before its delivery. It is important to observe that a local miss may be due to the absence of some sort of the requested resource or to not being able to adapt the resource on the edge node. To this purpose, we need to consider the three main types of adaptation (transcoding, state-less personalization, state-aware personalization) and to the entity that manages these edge nodes. Typically, transcoding and state-less personalization can be provided even by the edge adapters almost independently of the management organization. The deployment of state-less personalization and transcoding services by an edge node infrastructure managed by the provider or by a custom intermediary is facilitated by the fact that these services do not require previously stored information other than that directly supplied by the client. Hence, they can take full advantage of the highly distributed

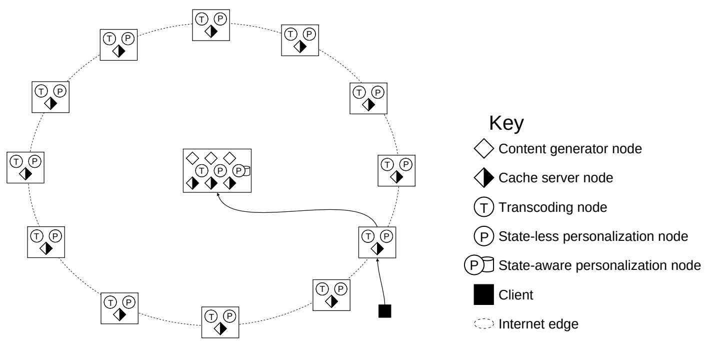

Figure 1.8. Adaptation services implemented by a cluster integrated with geographically distributed cache and adaptation servers

and replicated nature of the edge nodes. A third party independent infrastructure has some difficulties in providing all types of state-less personalization services because an edge adapter has to figure out in an indirect way the nature and the semantics of Web resources. For example, automated translation requires the identification of resource original language before attempting the translation process. The reduced direct interaction with the service provider can be solved when Web resources are accompanied by metadata providing additional information on the resource content and its semantics. For example, HTML pages may contain explicit meta tags (keywords describing the page content), and XML resources may be even richer about meta information. The use of metadata has been proposed for *server-directed adaptation* [Knutsson et al., 2002], where Web resources are bundled with additional information that describes how to carry out adaptation, especially transcoding-oriented.

The most serious difficulties arise for the state-aware personalization services that require an interaction with some database(s). Clearly, it is difficult, if not impossible, to replicate all state-aware personalization services among replicated adapters. The problems do not derive from the replication of the personalization algorithms, but from the distribution of the information on which these algorithms should operate. The well known problems related to the information replication over some units (e.g., caching [Fan et al., 2000], database management [Gray et al., 1996]) affect the possibility of multiplying services requiring dynamic content generation. Moreover, it seems impracticable to replicate information about users profiles and preferences stored on the content generator of the providers on each adapter node. The user information may be highly volatile, hence it is very difficult to guarantee its consistency among geographically distributed nodes. For third party based solutions an additional problem arises: a user profile database that is necessary to achieve personalized services is often characterized by a commercial value and privacy requirements. Hence, no content/service provider likes its replication over hundreds or thousands of nodes that are not strictly controlled by itself.

All these considerations lead us to conclude that most state-aware personalization services cannot be delegated to a distributed infrastructure consisting of a huge number of nodes. Actually, some solutions exist for personalization like random advertisement banners personalized according to the user language. These adaptation services are oriented to categories of users, and generate dynamic resources on the basis of (portion of) databases containing read-only nonvolatile information. Under these conditions, the databases can be easily replicated over many nodes of the intermediate infrastructure because their information is not affected by the above apparent consistency and privacy problems.

4.4 Geographic replication of clusters

The key feature of this architecture is the geographical replication of the cluster hosting content generators, adapters and cache servers. It is possible to refer to these architectures also as *multi-cluster architectures*. In such an infrastructure, it is possible to distinguish a *primary* cluster and multiple *secondary* clusters. The primary cluster acts as a master, while the secondary clusters are replicas of the primary cluster.

The multi-cluster architecture can be (and usually is) integrated with an infrastructure of edge nodes hosting cache servers and adapters as in the previous architecture. A typical multi-cluster architecture is shown in Figure 1.9, where it can be appreciated that there are few clusters (in terms of some units) providing every function, and a large number of other distributed nodes providing only a subset of the available functions such as caching, transcoding and/or state-less personalization.

The client request service in a multi-cluster architecture is similar to the previously described case of the architecture with one cluster integrated with multiple edge nodes. For this reason, we focus on the primary and secondary clusters. Let us assume that a client request is forwarded to a secondary cluster. Multiple dispatching algorithms can be used to select the secondary cluster that should receive the request. Dispatching can take into account multiple factors such as network and geographic distance, network link status, and cluster nodes load (where node load should take into account hardware resources such as CPU and memory and software resources such as file descriptors) [Cardellini et al., 2003].

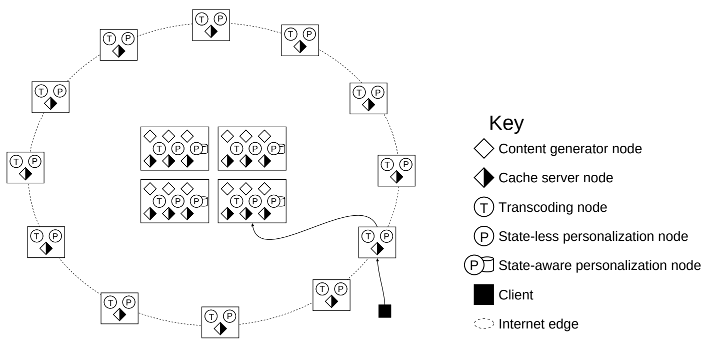

Figure 1.9. Adaptation services implemented by an infrastructure of geographically replicated clusters integrated with cache and adaptation servers on the edge

An alternative approach is to specialize secondary clusters to handle only a specific subset of the global working set [Canali et al., 2005]. Each content generator in secondary clusters handle only a subset of the resources that can be obtained by the content generator of the primary cluster. The dispatching algorithm allows each request to reach the appropriate cluster.

Multi-cluster architectures allow the deployment of any type of adaptation service. Transcoding and state-less personalization can be easily deployed over distributed architectures, as we have seen for the previous architecture. Multi-clusters also allow to provide state-aware personalization. The limited and controlled replication of the clusters addresses the issues of consistency and security in data management: if the number of replicas is reduced, strict data consistency protocols can be used. Moreover guaranteeing the security of few controlled sites is an affordable task [Canali et al., 2005].

5. Conclusions

The increased penetration of wired and wireless accesses to the Internet from highly heterogeneous client devices is a visible reality. There are major differences among these devices regarding network access methods, computational and memory powers, ability to accept and manage a large variety of data types. The ubiquitous accessibility combined with the dynamic generation of complex services and personalized content adds a new dimension to the problem of efficient generation and delivery of Web resources. The ubiquitous Web includes a huge literature on applications, middleware, protocols, transcoders, programmable proxies, standards, that are considered in other chapters of this book. Here, we focus on the (typically distributed) infrastructures that can provide efficient generation, adaptation and delivery of static and dynamic resources. The motivation is that, independently of the type of adaptation, the ubiquitous Web requires resource intensive operations that place significant overhead on CPU and/or disk if compared to the delivery of traditional Web resources. For example, efficient adaptation and delivery imply several functions, such as replication and caching of resources, client request management, lookup, and adaptation (including transcoding and personalization). Hence, the first question is where to map all these functions on the nodes of the distributed infrastructure. In particular, it seems interesting to solve the trade-off between completely centralized and extremely replicated schemes. Purely centralized architectures show scalability and fault-tolerance limits. On the other hand, replication solutions over a large geographic area are often affected by data consistency problems and management difficulties.

In this chapter, we have demonstrated that any research in the field of systems for the ubiquitous Web must take into account a third dimension of the problem which is the type of adaptation services to be deployed. Due to the heterogeneity of adaptations, an architecture that is optimal for transcoding may result poor for personalization services.

As a final observation, we consider that all analyzed architectures assume a well-known infrastructure of nodes that are typically stable and available most of the time. A radical shift in delivering services for the ubiquitous Web may be inspired to the novel peer-to-peer paradigms. It may be interesting to investigate a non-organized infrastructure of nodes that may join and leave the service infrastructure in a dynamic way, as it usually occurs in overlay peer-to-peer networks. Too many modifications in the infrastructure may degrade absolute performance, but self-organization properties of peer-to-peer networks can improve availability in critical conditions, hence there is space for further investigations.

References

Akamai Inc. (2004). http://www.akamai.com.

Canali, C., Cardellini, V., Colajanni, M., Lancellotti, R., and Yu, P. S. (2003). Cooperative architectures and algorithms for discovery and transcoding of multi-version content. In *Proc. of the 8th Intl. Workshop on Web Content Caching and Distribution (WCW03).*

Canali, Claudia, Cardellii, Valeria, Colajanni, Michele, Lancellotti, Riccardo, and Yu, Philip S. (2005). A two-level distributed architecture for web content adaptation and delivery. In *Proc. of The IEEE/IPSJ Symposium on Applications and the Internet (SAINT 2005).*

Cardellini, V., Casalicchio, E., Colajanni, M., and Yu, P. S. (2002). The state of the art in locally distributed web-server systems. *ACM Computing Surveys*, 34(2).

Cardellini, V., Colajanni, M., and Yu, P.S. (2003). Request redirection algorithms for distributed web systems. *IEEE Tran. on Parallel and Distributed Systems*, 14(5).

Chandra, S., Ellis, C. Schaltter, and Vahdat, A. (2000). Application-level differentiated multimedia web services using quality aware transcoding. *IEEE Journal on Selected Areas in Communication*, 18(12).

Davison, B. D. (2001). A Web caching primer. *IEEE Internet Computing*, 5(4).

Fan, L., Cao, P., Almeida, J., and Broder, A. Z. (2000). Summary cache: a scalable wide area web cache sharing protocol. *IEEE/ACM Trans. on Networking*, 8(3).

Gray, J., Helland, P., O'Neil, P. E., and Shasha, D. (1996). The dangers of replication and a solution. In *Proc. of the 1996 ACM SIGMOD International Conference on Management of Data*, volume Jun.

Knutsson, B., Lu, H., and Mogul, J. (2002). Architectures and pragmatics of server-directed transcoding. In *Proc. of 7th Int'l Workshop on Web Content Caching and Distribution.*

LaMacchia, B. (2002). Key challenges in drm: An industry perspective. In *Proc. of 2002 ACM Workshop on Digital Rights Management.*

Maheshwari, A., Sharma, A., Ramamritham, K., and Shenoy, P. (2002). Transsquid: Transcoding and caching proxy for heterogeneous e-commerce environments. In *Proc. of 12th IEEE Int'l Workshop on Research Issues in Data Engineering*.

Rabinovich, M. and Spatscheck, O. (2002). *Web Caching and Replication*. Addison Wesley.

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

Speedera Networks inc (2004). http://www.speedera.com.

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

Wessels, D. (2001). *Web caching*. O'Reilly.