A TAXONOMY OF PROGRAMMABLE HTTP PROXIES FOR ADVANCED EDGE SERVICES

Delfina Malandrino
Dipartimento di Informatica ed Applicazioni "R.M. Capocelli"
Università di Salerno
Via S. Allende, Baronissi (SA), 84081, Italy

Vittorio Scarano
Dipartimento di Informatica ed Applicazioni "R.M. Capocelli"
Università di Salerno
Via S. Allende, Baronissi (SA), 84081, Italy

Keywords: HTTP proxy, programmable proxy, edge services.

Abstract: In this paper, we present the state of the art in the field of programmability in HTTP proxies. In particular, we first deal with programmability and show how it is a crucial requirement to easily realize and assemble edge services that can enhance the quality and the user perception of the navigation into a crowded and confusing World Wide Web. Then, we compare some of the most used HTTP proxies to provide an analysis of their programmability and, finally, show some evidence of successful edge services realized on top of existing programmable HTTP proxy frameworks.

1 INTRODUCTION

The architecture of the World Wide Web has undergone a significant change in the last 10 years. The simple client-server architecture, that was one of the key factors of the World Wide Web success, has been integrated with multiple interconnected nodes disseminated across the Internet so that end-users can experience advanced services with better performances (low latency), fault tolerance and high availability.

Despite the success of these architectures, the introduction of new services into existing networks is often an expensive and time-consuming process. In order to improve and simplify this process, crucial has become the requirement of *programmability* in all the Internet infrastructures. A programmable infrastructure offers a quick prototyping and assembling of new services as well as easy and responsive modifications of existing ones.

In this research area, the interest is mainly devoted to frameworks that make possible an easy development of new services as well as enhancing the existing core functionalities of the distributed environments that provide such services.

Proxy servers are intermediary entities that can be placed anywhere along an information stream in the *content path* between origin server and client systems aiming to improve the performance of the WWW and its scalability.

Intermediaries can be developed in order to provide *intelligent* services beyond simple and traditional caching and content replication services. These value-added services, also called *edge* services, include content adaptation and personalization, localization, content aggregation, mobility and ubiquitous access to the Internet content.

The *edge* of the network is a point that resides in the path between clients and servers where content processing is requested. Many are the potential places where edges can store and change content or provide services, in particular (for our purposes) examples are proxy servers.

The goal of the paper is to show how proxy servers represent the ideal place for adding programmability functionalities. For example, intermediaries could choose to handle the request immediately, execute content processing to produce and deliver value-added content, access resources on remote locations and so on. By splitting the issuing of the requests from making and managing them, it is possible to increase the flexibility (new functionalities can be added without changing the existing ones) and the power of the Web.

Organization of the paper: In the following section,

Malandrino D. and Scarano V. (2005).
A TAXONOMY OF PROGRAMMABLE HTTP PROXIES FOR ADVANCED EDGE SERVICES.
In *Proceedings of the First International Conference on Web Information Systems and Technologies*, pages 231-238
DOI: 10.5220/0001233802310238
Copyright © SciTePress

we briefly described how simple proxy services have evolved into complex edge services. Then we detail, in Section 3, what we mean by a programmable infrastructure. In Section 4 we describe how each programmable proxy infrastructure is rated against the dimensions we outlined before. Finally, in Section 5 we describe some advanced edge services that can be obtained only by leveraging on highly configurable and programmable proxy infrastructures and conclude the paper in Section 6 with some final remarks.

2 FROM PROXY TO EDGE SERVICES

As the nature of the services delivered by the Web grew more and more rich and heterogeneous, it became necessary to design and implement more and more elaborate caching strategies and infrastructures that later evolved into the "Content Delivery Networks" (CDNs) (B. Krishnamurthy, 2001) that, first, introduced the concept of "*Edge*" of the network.

While content networks create a virtual overlay of *Edge intermediary servers*, on the top of the packet network, for an efficiently delivering of customized content to the end users, the *Edge* Services Overlay Network moves one step further, by providing a services infrastructure that involve the development of another intelligent overlay network around CDNs.

The idea of the Edge overlay networks is to develop and deploy software entities, strategically distributed through the network, to provide content adaptation and other complex functionalities (personalization, location-aware data insertion, etc) on the data flow exchanged between clients and servers. These software entities must be able to provide a *value-added* on the content delivered to end users and, in the meantime, ensure robustness, high availability and scalability performance.

Intermediary software infrastructures exhibit features that match the above requirements. They realize a transparent deployment by appearing as a client for existing servers and as a server for existing clients, reduce requirements on clients and complexity on servers and, finally, they are able to provide complex intermediary functionalities, without involving software/hardware modifications both on client and server systems. In addition, it is well-know that proxies are essential to improve Internet access by reducing user-perceived latency and assure a better quality of services, to which WWW users are most interested in.

Intermediary systems, layered on existing network infrastructures, can be easily programmed and deployed to operate as an overlay network of connected or cooperative components in order to efficiently provide advanced edge services.

Many are the application fields of edge computing services that our vision encompasses: from the *geographical personalization* of the navigation of pages, with insertion/emphasis of content that can be related to user geographical location, to *translation services* (IBM Almaden Research Center, 1999); from support for *group navigation and awareness* (Barrett and Maglio, 1998; Calabrò et al., 2003) for social navigation (Barra et al., 2002) to advanced services for *bandwidth optimization* such as adaptive compression and format transcoding (Ardon et al., 2003a; Hori et al., 2000; WebSphere,). A particularly useful application field for edge services is the accessibility of Web sites. These applications are usually heavyweight and therefore, assuring their scalability is particularly critical. A typical example is an ubiquitous service that provides text-to-speech (Barra et al., 2003) translation of the pages navigated by the user, regardless of the location and configuration of the local machine (i.e. without installing particular software).

Of course "traditional" proxy services like caching can be further enhanced by combination with advanced edge services: an example can be offering the user a "search engine" that operates only on cached/personalized/translated pages.

3 PROGRAMMABILITY OF PROXIES

In this section we introduce a detailed taxonomy of characteristics that intermediary systems should exhibit for quick development and easy deployment of advanced edge computing services. In particular we identify and investigate the following important features.

• *Programmability.*
With this term we mean the requirement of changing or updating the functionalities provided by a software infrastructure as well as the possibility to entirely develop new intermediary services. The core functionalities must be initially provided and off-loaded into the intermediary infrastructure and, after programmers must develop and deploy new application services.

Programming under existing intermediaries cannot involve the same power, efficiency, and generality like developing an open system from scratch. This means that we need of an execution environment, in which a compositional framework should provide the basic components for developing new services, and a programming model to make this execution environment a *programmable* environment. Such programming model should provide APIs, software libraries and languages (object-oriented or others), for programming and deploying new services into the intermediary infrastructure.

Moreover, by enhancing the distinction between the core network infrastructure and the programming model for developing new services, programmers can develop services without taking care of details connected to scalability, fault tolerance and availability.

• *Configurability.*
With this term we mean the requirement of configuring the set of services that can be invoked during Web transactions. In particular we can distinguish into two types of possible configurations: (*a*) a simple configuration for adding, removing, enabling or disabling functionalities and (*b*) a more detailed configuration in which service parameters can be specified by asking the user to fill-out Web-based forms (for example by providing a downgrade quality image for an image transcoding service). In this context another important requirement concerns the possibility to load, install and execute new services at run-time without recompiling and restarting the running software infrastructure.

Finally service parameters could be specified through a Graphical User Interface (GUI) to simplify the process of services configuration. Moreover, that GUI could to appear useful for monitoring the behavior of the system (invoked services, HTTP requests and responses, etc).

• *Adaptability.*
With this term we mean the need to address the issues of the existing and emerging wireless and mobile technologies that require content adaptation and dynamic reconfiguration based on the specific context: limited devices capabilities (i.e. small screen sizes) limited or intermittent connection to the wired network, etc.

• *Centralized or Distributed System Architecture.*
The architecture of a software intermediary architecture can be expressed by components that can be in execution on a single host (centralized architecture) or, rather, by components that are distributed on different hosts and that collaborate to provide Internet services and other complex functionalities such as load balancing, fault tolerance, high availability and performance scalability.

• *Lifecycle support of applications.*
This requirement is needed to support the deployment and un-deployment of proxy services, by making these tasks automatic and accessible by both locally and remote locations. Application deployment is an important system functionality that allows client to anytime-anywhere access to provided services and applications. By making the deployment an automatic task (i.e. wizard) it is possible to add new functionalities into the intermediary system without taking care of the complexity of the system itself, and without manipulating and changing system configuration files (which is a typical error-prone task).

Active deployment allow to add new services into the network or to customize existing ones to math new application demands without restarting or reconfiguring Web proxy systems that host the services.

• *Personalization.*
User Profiling represents the an efficiently way, within mobile and ubiquitous environments, to provide personalized services according to the more and more different and heterogeneous client devices that have access to the more and more dynamic and interactive Internet content. Users profiles are also important to make difference between groups of users. If the intermediary system supports this functionality, some authentication mechanism must be provided.

• *Services activation.*
An important feature that a distributed service infrastructure could exhibit is the mechanism for services activation. A rule-based services activation can be useful to dynamically determining the data path through the various components (that provide the services) allowing arbitrarily complex boolean expressions instead of simple URL and content-type matching. Such rules should be dynamically modified to add or remove services, or to change their conditions.

• *Maintainability of software.*
Finally, an important feature concerns the availability (source code) of the developed software in order to ensure a swift adaptation of the software to the demands for new advanced services. In fact, open-source software is considered the most efficient solution when system infrastructure has to be easily extended and enhanced with new system functionalities.

4 INTERMEDIARY SYSTEMS

In this section we briefly describe the interemediary systems that are, then, compared in Table 1.

RabbIT. RabbIT (RabbIT,) is a Web HTTP proxy that accelerates the delivery of Web contents to end users by compressing text pages and images, by removing unnecessary parts of HTML pages (background images, advertisements, banners, etc.) and finally by caching filtered documents before forwarding them to the clients.

RabbIT proxy, written in Java, has a modular architecture that allow an easy definition and implementation of new functionalities. In addition, these modules can be configured at runtime (no parameter configuration is provided), but we need to restart the proxy to keep all chosen settings.

RabbIT Web proxy is characterized by a centralized structure, it is programmable, partially configurable (no services parameters are allowed), it does not provide support for adaptability, user profiling and automatic deployment. New services (filters) can be added to the proxy but we need to access configuration files and manually specify new settings.

Muffin. Muffin (Muffin, 2004) is a Web HTTP proxy that provides functionalities to remove cookies, kill GIF animations, remove advertisements, add, remove, or modify any HTML tag, remove Java applets and Javascript code, etc.

It is programmable and configurable proxy, new services or filters can be developed through a set of provided APIs and can be added at runtime, using the provided graphical interface, without restarting the proxy.

WebCleaner. WebCleaner (Webcleaner,) is a filtering HTTP proxy that provides functionalities for removing advertisements, banners, flash and javascript code, for reducing image (by size) and compressing HTML pages (with gzip), etc.

WebCleaner proxy is programmable and highly configurable, but it does not provide support for user profiling. New services can be added into the system through a graphical user interface. Through this GUI new *Filter Modules* can be added by specifying appropriate configuration parameters.

FilterProxy. FilterProxy (FilterProxy, 2002) is a modular HTTP proxy that provides functionalities for compressing HTML pages, image transcoding, etc. It is programmable and highly configurable, new services (or filters) can be dynamically added to the system, and their configuration is realized through web-based forms, or editing a configuration file. Services can be chained in order to provide complex functionalities. FilterProxy is written in perl, and is quite fast.

Privoxy Web Proxy. Privoxy (Privoxy, 2004) is a Web proxy with advanced filtering capabilities for protecting privacy, modifying Web page content, controlling access, and removing advertisements, banners, pop-ups, etc.

Privoxy has a very flexible configuration and can be customized according to individual user needs. Specifically, Privoxy can be configured with various configuration files. By typing a specific URL into the Web browser it is possible to select a set of actions and Privoxy's configuration parameters can also be viewed at the same page.

It provides a set of API and a detailed documentation on how to develop and deploy new services.

WBI. Web Based Intermediaries (WBI) is a dynamic and programmable framework, developed at IBM Almaden Research Center (IBM Almaden Research Center, 1999), whose main goal is to personalize the Web by realizing an architecture that simplify the develop of intermediary applications. WBI defines a programming model that can be used to implement all form of intermediaries, from simple server functions to complex distributed applications. WBI is freely usable but its sources are not publicly available.

A WBI transaction is defined as a complete HTTP request-response and flows through a combination of the four type of basic stages, called MEGs: RequestEditor and Editor MEGs, that modify HTTP requests and responses respectively, the Generator MEG that given an HTTP request produce the corresponding response and finally, the Monitor MEG that *monitors* HTTP requests and response without modify them at all. A group of such MEGs is called WBI plugin.

WBI provides a rule-based system for dynamically determining the data path through the various MEGs allowing complex boolean expressions. With WBI, rules can be dynamically altered to add or delete MEGs, or to change their firing condition. Finally the WBI GUI provides a convenient way to manage and administer WBI, and to help users to debug the plugins loaded into the proxy.

AT&T Mobile Network iMobile. iMobile (Rao et al., 2001) is a proxy-based platform designed to provide personalized mobile services. It provides a modular architecture that support accesses from various mobile devices to various information spaces.

The main component of iMobile platform is iProxy (AT&T Labs-Research, 2002), a programmable proxy server that provides an environment for developing personalized services, which are implemented as building blocks in Java.

The modularity into the iMobile framework is achieved through three different abstractions: (*a*) devlets, that support the provision of iMobile services to different type of mobile devices, (*v*) infolets, that provide a communication mechanism between iMobile servers and the information sources, and (*c*) the applets, that encapsulate the services functionalities.

The iMobile project was subsequently extended into e new project, iMobile EE (Chen et al., 2003), to address the requirements of security, high availability and performance scalability.

eRACE. The *extensible Retrieval Annotation and Caching Engine* (eRACE) (Dikaiakos and Zeinalipour-Yiazti, 2001a) is a modular, programmable and distributed intermediary infrastructure whose main goal is to provide personalized services for a wide range of client devices (Desktop PC, mobile and thin clients), such as, personalization, customization, filtering, aggregation, transformation both on wireline and wireless Internet. The client requests are scheduled for execution by an eRACE scheduler which can implement different policies for load balancing. Authentication and profiling are managed by a Service Manager integrated into the system.

The most important component of the system is WebRACE (Dikaiakos and Zeinalipour-Yiazti, 2001b) a proxy system that deals with WWW information. It consists of a Web crawler, a filtering processor and an object cache. These components can be distributed on several nodes and communicate through socket links.

TACC. The BARWAN project (Katz et al., 1996) by UC/Berkeley has the goal to provide intermediary systems to support ubiquitous access to Internet services from mobile and thin clients. An important system component of this project is the programmable proxy architecture, TACC, that acts as intermediary between servers and mobile clients.

The main component of the TACC compositional framework is the worker, that encapsulates the service functionality. Workers, that represent the building blocks of TACC applications, can be chained in order to provide more complex functionalities, such as transformation, aggregation, caching and customization of Web content (Brewer et al., 1998; Fox et al., 1998a). Other modules handle user profiling and provide mechanisms for load balancing and fault-tolerance.

ALAN. ALAN, acronym of Application Level Active Network (Fry and Ghosh, 1999), is a network architecture that provides mechanisms to dynamically load code, or proxylet, to facilitates Web content delivery (MacLarty and Fry, 2001). Communications is enhanced by Dynamic Proxy Server that are located at strategic points along the content path between client and server.

Proxylet can be downloaded, as JAR archive, from HTTP servers into a Dynamic Proxy Server infrastructure, making their deployment very easy (URL to reference proxylets). The main contributions of this approach are: (a) the fast deployment of new communication services on demand without the drawbacks of the Active Networks (deployment of new elements at network routers) and (b) a platform that provide flexible and value-added services (WWW streaming audio, WWW compression, etc).

WebPADS. WebPADS (Chuang and Chan, 2005), acronym of Web Proxy for Actively Deployable Services, provides a framework that facilitates the development of add-on services or *mobilets*, which can be actively deployed and migrated across Web proxies, according to the dynamic changes in wireless environments.

WebPADS, developed in Java, provides a set of APIs for developing mobilets and mobile applications. Prototype implementations, to prove the framework's benefits, include compression and image transcoding services.

Finally, mobilets can be chained together in a nested order to allow service aggregation and an easy reconfiguration if changes in the mobile environments occur.

MARCH. MARCH (Ardon et al., 2003b) is a distributed content adaptation architecture that provides functionalities for adapting Web content according to the capabilities of client devices that access to Internet services.

While MARCH enables the creation of an overlay network of intermediary entities, deployed between clients and servers, it adopts a server-centric approach, by delegating, for example, the decision of which proxy invoke for a given service (front end component of the system) to a centralized mobile aware server.

An important feature of this system architecture is that it allows the dynamic composition of services (*proxy chain*) for a given set of operating conditions (CPU, memory, etc). MARCH is implemented in Java and prototype implementations include examples of transcoding services and a compression service.

Squid. Among the "classical" (i.e. non programmable) proxy, particularly popular is SQUID (Squid Proxy,). It is a high-performance proxy caching server for web clients, supporting FTP, gopher, and HTTP data objects. Unlike traditional caching software, Squid handles all requests in a single, non-blocking, I/O-driven process. Squid acts as an agent, accepting requests from clients (such as browsers) and passes them to the appropriate Internet server. It stores a copy of the returned data in an on-disk cache. The real benefit of Squid emerges when the same data is requested multiple times, since a copy of the on-disk data is returned to the client, speeding up Internet access and saving bandwidth.

Despite its efficiency in dealing as proxy cache, Squid is not explicitly mentioned here in the comparison, since it lacks a real support for programmability. In fact its main functionalities include caching of HTTP, FTP and other URLs, transparent caching, proxy-ing for SSL, HTTP server acceleration. Most of SQUID-based products focus on caching, firewall technologies, security, network monitoring and content filtering (Cerberian,).

Table 1: Proxy Comparison. Notice that RabbIT is only partially configurable, i.e. it does not allow to specify parameters for services.

	Programmability	Configurability	Adaptability	Centralized / Distributed	Lifecycle Support	Personalization	Activation Services	Maintainability
RabbIT	√	~	-	C	-	-	-	√
Muffin	√	√	-	C	√	-	-	√
Privoxy	√	√	-	C	-	-	-	√
WebCleaner	√	√	-	C	√	-	-	√
FilterProxy	√	√	-	C	√	-	-	√
WBI	√	√	-	C	√	-	√	Free
iMobile	√	√	√	D	√	√	√	-
eRACE	√	√	√	D	√	√	√	-
TACC	√	√	√	D	√	√	√	-
ALAN	√	-	-	D	√	√	√	Free
WebPADS	√	√	√	D	√	-	√	-
MARCH	√	√	√	D	√	√	√	-

5 ADVANCED EDGE SERVICES

The current trend in the field of pervasive computing is how to enable universal access to the Internet content, that is, how to address the critical issue of developing technologies that are able to allow the transformation, or content adaptation, of Web content for adapting it to the different and heterogeneous client devices that exhibit different capabilities in terms of network connectivity, processing power, storage, display capabilities, etc.

More precisely, content adaptation or *transcoding* represents the process of converting Web content from one form to another, and it is typically carried out by content servers or, on-the-fly, by proxy servers at the *Edge* of the network.

Content adaptation refers to two different abstractions, that is, *personalization*, the adaptation of the Web content according to user preferences, locations and contexts, and *transcoding*, the process of of tailoring Web content to the capabilities of the client device and the network connections.

Adaptation can take place at client-side, server-side or at intermediary entities in between. In a proxy-based approach, content adaptation can be performed on-the-fly, without overload the clients (as in client-based approaches) and without allow multiple variants of the same content (as in server-side approaches).

Several proposal, academic (Fox et al., 1998a; Fox et al., 1998b; Fox and Brewer, 1996; Zenel, 1999) and commercial products (WebSphereEdgeServices,), are HTTP proxy-based, since they explore proxy systems to adapt multimedia Internet content.

Scone (Weinreich et al., 2001) is a modular Java framework whose main gaol is to improve the navigation and orientation on the Web, by generating new views of Web documents, offering workgroup tools to support collaborative navigation, enriching Web pages with new navigational elements, etc.

Extraction of useful and relevant part of Web documents has many applications, such as displaying for devices with limited size capabilities, speech rendering for users with visual disabilities, summarizing of Web contents, etc.

The process of extracting part of a document requires two important steps: (*a*) the parsing of HTML Web pages (to remove unnecessary elements, links, images, etc), and (*b*) the building of the DOM representation of such pages (for their reformatting). These operation can be efficiently realized on the top of intermediary systems (Gupta et al., 2004) in order to reduce complexity both on client and server systems.

Content aggregation refers to the generation of content at intermediary systems. More precisely, an *aggregator* queries one or more Web sites for specific content (news, search. etc) and collects and formats the results for the presentation to the clients. In the TOP GUN Wingman proxy (Fox et al., 1998b), the aggregation service allows the user to enter some information describing what is being requested. The request is processed by a service module into the proxy and the response is formatted by an HTML processor. This aggregator works (and produce results) for Yahoo, HotBot, AltaVista, DejaNews, Yahoos stock quotes service, and TripQuest.

Due to its popularity and its ease of use, the Web is an attractive platform to support distributed cooperative works (Cabri et al., 1999). These systems can help users to quickly find information among the growing amount of information available on the WWW.

A proxy-based architecture can be employed to support synchronous cooperation on the Web. By programming a proxy-framework, many cooperative functionalities can be provided to allow users within a common workgroup to share and cooperate toward a common goal.

Proxy-based frameworks can be also programmed to operate as recommender systems, assisting users in their navigation and making users aware of others's information activities, interests and knowledge. The WebMemex system (Macedo et al., 2003) was developed on top of a high level architecture, an augmented Web proxy server, that provides capture, linking, user authentication, storage, retrieval and access capabilities.

Another interesting functionality that can be offered by proxy systems concerns the recording of Web navigation. An example is WebMacros (Safonov et al., 2001) a proxy-based system for automating repetitive user interactions with the Web by recording and replaying user navigation.

Finally, intermediary systems can be used as visualization systems (Hong and Landay, 2001), in order to help Web design teams to run usability tests and analyze the collected data, and Web users for revisitation of Web pages by allowing dynamic drawing of animated graphs of the user's paths as the surf the Web.

6 CONCLUSION

The examples shown in the last section witness the usefulness of programmability in intermediary infrastructures. Without a full support toward the development of new services, advanced mechanisms cannot be easily realized, deployed and managed, while quick prototyping and manageability represent crucial requirements to assure that programmers can quickly respond to mutations of data format, content or standards that are so common in the Web, nowadays. Our comparison shows how this field is now proposing several platforms that share a common objective, to ensure that advanced edge services can be efficiently and effectively realized.

An important direction toward standardization is being conducted by the IETF Working Group for "Open Pluggable Edge Services" (OPES)[1]. Their goal is to define an open standard that allows intermediaries to provide services on the data flow. In OPES, intermediaries can also employ local or remote servers (called "callout servers") in order to facilitate the efficient delivery of complex services. OPES rule-set are applied in order to choose which service to apply to the data flow. Communication between dispatchers and callout servers occurs through a protocol that must obey the requirements for OPES Callout Protocols specified in (Beck et al., 2002).

Of course, as OPES gains momentum, programmability of proxies will get an additional characteristics to be measured against, namely the compatibility with an open standard to interconnect components. Therefore, it is clearly foreseeable that, if OPES gets enough support, the environments for programmable proxies will take into account their openness to standards.

[1]See (Barbir et al., 2003) for the description of the architecture and the Web site `http://standards.nortelnetworks.com/opes` for the overall activities of this WG.

REFERENCES

Ardon, S., Gunningberg, P., LandFeldt, B., Y. Ismailov, M. P., and Seneviratne, A. (2003a). MARCH: a distributed content adaptation architecture. *International Journal of Communication Systems, Special Issue: Wireless Access to the Global Internet: Mobile Radio Networks and Satellite Systems.*, 16(1).

Ardon, S., Gunningberg, P., LandFeldt, B., Y. Ismailov, M. P., and Seneviratne, A. (2003b). March: a distributed content adaptation architecture. *International Journal of Communication Systems, Special Issue: Wireless Access to the Global Internet: Mobile Radio Networks and Satellite Systems.*, 16(1).

AT&T Labs-Research (2002). iProxy: a Programmable Proxy. http://www.research.att.com/sw/tools/iproxy/.

B. Krishnamurthy, C. Wills, Y. Z. (2001). On the use performance of contend distribution network. In *Proceedings of SIGCOMM IMW*.

Barbir, A., Chen, R., Hofmann, M., H.Orman, and Penno, R. (2003). An Architecture for Open Pluggable Edge services (OPES). http://www.ietf.org/internet-drafts/draft-ietf-opes-architecture-04.txt.

Barra, M., Grieco, R., Malandrino, D., Negro, A., and Scarano, V. (May 2003). TextToSpeech: an heavy-weight Edge computing Service. In *Poster Proc. of* 12^{th} *International World Wide Web Conference*. ACM Press.

Barra, M., Maglio, P., Negro, A., and Scarano, V. (2002). GAS: Group Adaptive System. In *Proceedings of International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH 2002)*. ACM Press.

Barrett, R. and Maglio, P. P. (1998). Adaptive Communities and Web Places. In *Proceedings of* 2^{th} *Workshop on Adaptive Hypertext and Hypermedia, HYPERTEXT 98.*, Pittsburgh (USA). ACM Press.

Beck, A., Hofmann, M., Orman, H., Penno, R., and Terzis, A. (2002). Requirements for OPES Callout Protocols An Architecture for Open Pluggable Edge Services

(OPES). http://www.faqs.org/ftp/internet-drafts/draft-ietf-opes-protocol-reqs-02.txt.

Brewer, E., Katz, R., Amir, E., Balakrishnan, H., Chawathe, Y., Fox, A., Gribble, S., Hode, T., Nguyen, G., Padmanabhan, V., Stemm, M., Seshan, S., and Henderson., T. (1998). A Network Architecture for Heterogeneous Mobile Computing. *In IEEE Personal Communication Magazine*, 5(5):8–24.

Cabri, G., Leonardi, L., and Zambonelli, F. (1999). A proxy-based framework to support synchronous cooperation on the web. *Softw. Pract. Exper.*, 29(14):1241–1263.

Calabrò, M. G., Malandrino, D., and Scarano, V. (2003). Group Recording of Web Navigation. In *Proceedings of the HYPERTEXT'03*. ACM Press.

Cerberian. Cerberian web filtering software. http://www.cerberian.com/main.html.

Chen, Y.-F., Huang, H., Jana, R., Jim, T., Hiltunen, M., John, S., Jora, S., Muthumanickam, R., and Wei, B. (2003). iMobile EE: an enterprise mobile service platform. *Wirel. Netw.*, 9(4):283–297.

Chuang, S. N. and Chan, A. T. (2005). Active service for mobile middleware. *WWW: Internet and Web Information Systems Journal*.

Dikaiakos, M. and Zeinalipour-Yiazti, D. (2001a). A distributed middleware infrastructure for personalized services. Technical Report TR-2001-4, University of Cyprus.

Dikaiakos, M. and Zeinalipour-Yiazti, D. (2001b). WebRACE: A Distributed WWW Retrieval, Annotation, and Caching Engine. In *Proceedings of PADDA01: International Workshop on Performance-oriented Application Development for Distributed Architectures*.

FilterProxy (2002). FilterProxy. http://filterproxy.sourceforge.net/.

Fox, A. and Brewer, E. A. (1996). Reducing WWW latency and bandwidth requirements by real-time distillation. In *Proceedings of the 5^{th} International World-Wide Web Conference*. ACM Press.

Fox, A., Chawathe, Y., and Brewer, E. A. (1998a). Adapting to Network and Client variation using active proxies: Lessons and perspectives. *IEEE Personal Communications*, 5(4):10–19.

Fox, A., Goldberg, I., Gribble, S., Lee, D., Polito, A., and Brewer, E. (September 1998b). Experience with top gun wingman, a proxy-based graphical web browser for the usr palmpilot. In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98)*.

Fry, M. and Ghosh, A. (1999). Application level active networking. *Comput. Networks*, 31(7):655–667.

Gupta, S., Kaiser, G. E., Grimm, P., Chiang, M. F., and Starren, J. (2004). Autmatic content extraction of html Documents. *World Wide Web Journal*.

Hong, J. I. and Landay, J. A. (2001). Webquilt: A framework for capturing and visualizing the web experience. In *Proceedings of the 10^{th} International World Wide Web Conference*, Hong Kong. ACM Press.

Hori, M., Kondoh, G., Ono, K., Hirose, S., and Singhal, S. (2000). Annotation-Based Web Content Transcoding. In *Proc. of the 9^{th} Intl. World Wide Web Conf.*, Amsterdam (The Netherland). ACM Press.

IBM Almaden Research Center (1999). Web Based Intermediaries (WBI). http://www.almaden.ibm.com/cs/wbi/.

Katz, R. H., Brewer, E. A., Amir, E., Balakrishnan, H., Fox, A., Gribble, S., Hodes, T., Jiang, D., Nguyen, G. T., Padmanabhan, V., and Stemm, M. (1996). The Bay Area Research Wireless Access Network (BARWAN). In *Proceedings of the 41st IEEE International Computer Conference*, page 15. IEEE Computer Society.

Macedo, A. A., Truong, K. N., Camacho-Guerrero, J. A., and da Gra&#199;a Pimentel, M. (2003). Automatically sharing web experiences through a hyperdocument recommender system. In *HYPERTEXT '03: Proc. of the 14th ACM Conf. on Hypertext and Hypermedia*, pages 48–56. ACM Press.

MacLarty, G. and Fry, M. (2001). Policy-based content delivery: an active network approach. *Computer Communications*, 24(2):241–248.

Muffin (2004). Muffin Proxy. http://muffin.doit.org/.

Privoxy (2004). Privoxy Web Proxy. http://www.privoxy.org/.

RabbIT. RabbIT proxy. http://rabbit-proxy.sourceforge.net/.

Rao, H. C.-H., Chang, D.-F., Chen, Y.-F., and Chen, M.-F. (2001). iMobile: a proxy-based platform for mobile services. In *Wireless Mobile Internet*, pages 3–10.

Safonov, A., Konstan, J. A., and Carlis, J. V. (May 2001). Webmacros - a proxy-based system for automating user interactions with the web. In *Poster Proc. of 10^{th} International World Wide Web Conference*. ACM Press.

Squid Proxy. Squid Proxy. http://www.squid-cache.org/.

Webcleaner. Webcleaner Filter Proxy. http://webcleaner.sourceforge.net/.

WebSphere. IBM Websphere Transcoding Publisher. http://www-3.ibm.com/software/webservers/transcoding.

WebSphereEdgeServices. Websphere Edge Server. http://www-3.ibm.com/software/webservers/edgeserver.

Weinreich, H., Obendorf, H., and Lamersdorf, W. (2001). The look of the link, concepts for the user interface of extended hyperlinks. In *Proceedings of the 12th ACM Conference on Hypertext and Hypermedia, University of Aarhus, Denmark*, pages 19–28. The Association for Computing Machinery, ACM New York.

Zenel, B. (1999). A general purpose proxy filtering mechanism applied to the mobile environment. *Wirel. Netw.*, 5(5):391–409.