

# Mobile Aware Server Architecture: A distributed proxy architecture for content adaptation

SÉBASTIEN ARDON<sup>1,4</sup>, PER GUNNINGBERG<sup>3</sup>,  
YURI ISMAILOV<sup>2</sup>, BJÖRN LANDFELDT<sup>2</sup>, MARIUS PORTMANN<sup>1,5</sup>,  
ARUNA SENEVIRATNE<sup>1</sup>, BINH THAI<sup>1</sup>

<sup>1</sup>School of Electrical Engineering and  
Telecommunications  
University of New South Wales  
Sydney NSW 2051 Australia  
{aruna.seneviratne, ardon,  
marius.portmann}@unsw.edu.au

<sup>2</sup>Ericsson research Networks and Systems  
Torshamnsgatan 23 - 164 80 Kista Sweden  
{bjorn.landfeldt, yuri.ismailov}@era.ericsson.se

<sup>3</sup>Uppsala University  
Department of Computer Systems  
Box 325, S-751 05 Uppsala, Sweden  
perg@docs.uu.se

<sup>4</sup>LIP6 laboratory  
Pierre and Marie Curie University  
4, place Jussieu 75252 Paris Cedex 05, France

<sup>5</sup>Computer and Networks Laboratory TIK Swiss  
Federal Institute of Technology ETH CH-8092  
Zürich, Switzerland

## Abstract

We present a novel proxy architecture which provides a middleware solution for adapting Internet Content and its transmission to client and network characteristic variations. We present details of the Mobile Aware Server Architecture (MARCH), a framework for automatic session customization and proxy deployment which operates on top of the current Internet. Our framework differs from earlier work in that it is server-centric, i.e. the decisions

as to what adaptations to perform and how to perform them are made under the content server administration authority. The framework is particularly designed for the current client/server model, in particular, it does not cater for interpersonal communication applications.

## 1 Introduction

Over the past years, the number of hosts connected to the Internet has grown tremendously. With the introduction of the 3rd generation and next mobile systems, this number is expected to grow further. At the same time, terminals connected to the Internet have and will have very different capabilities in terms of media presentation, processing power and power autonomy. With the constant demand for new networked applications and the desire to use these applications with any type of terminals through any type of access network, we face new challenges in the distribution and presentation of content to the users. Furthermore, we can expect this heterogeneity to increase as new terminal types and new access network types emerge.

We believe that maintaining several copies of the content at the source to cater for each application/terminal/network is not practical and scalable.

The March framework (early ideas presented in [10]) is a novel Content Adaptation Architecture which uses application-level entities that we call *proxies*, deployed dynamically on carefully chosen nodes across the network. These proxies are used to perform content transformation(s), or some protocol-specific performance enhancement func-

tions.

The remaining of this paper is organised as follows: section 2 review current related proposals, section 3 introduces the March framework and finally section 4 present our conclusions and future work.

## 2 Related work

Problems associated with the heterogeneity of the Internet have lead to various proposals in the areas of Active Networks, Active Services and Application Layer Active Networking. These proposals address the heterogeneity problem by adapting the content and transmission characteristics to suit the session characteristics. In this section, we review the existing approaches in this area.

The problem can be adressed by using various technologies operating at different layers in the networking procol stack. In this section, we review previous work, which we have ordered depending on the layer at which the solution operates.

### **Content Adaptation with Active Network or Active Services**

Content adaptation schemes have been proposed that operate at the network level and are often referred to as "active network" based proposals. An Active network, as originally introduced in [15], is defined as a network in which the "routers or switches of the network perform customized computation on the message flowing through them". This concept can be applied to both, the extreme capsule model where the packet headers carry the code to be executed on the routers or switches, or to the programmable switch model in which switches can be extended to perform computations on packets, but the code is not necessarily carried in each packet itself. Some proposals such as [16, 11] use this approach to perform content adaptation functionalities However the scalability and feasibility of these schemes are still being discussed.

Active services [3] or Application-Layer active networks [8], propose to deploy such high-level services at the application layer. This allows to retain the existing robust and simple internet routing subsystem and network layer untouched, therefore facilitating incremental deployment over the current Internet architecture [3]. These proposals

extend the active network model in that the decision process for what adaptation mechanisms to deploy is still embedded in the network in-between the client and the server, but they operate at a higher layer. We believe that the genericity of this approach makes the decision-making process too complex and therefore inefficient.

Work in progress within the Internet Engineering Task Forces (IETF) aims at defining an open platform to execute services at arbitrary intermediaries in-between the client and the server [1]. The Internet Content Adaptation Protocol (ICAP) [6] is considered as one possible signalling protocols to convey content adaptation, but is limited to HTTP traffic only.

### **Content Adaptation with static Proxy services**

Proxy services are application-layer software entities that request or process content on behalf of the clients. Some proposals such as [7] and even commercial products explore the use of proxies to adapt multimedia content to client and network variations. In these fixed or 'static' proxy architectures, content adaptation is realized by proxies, software components placed at strategically chosen points around the network. (e.g. at or near network performance discontinuities). Client application requests are then directed to the proxy either explicitly (e.g. the user manually configures the proxy settings in the client application) or transparently as described in [5]. This static operation has several drawbacks. Firstly, the scalability is limited: as new terminals, access networks and applications emerge, the fixed architecture makes the deployment of new proxies services difficult. Secondly, the reliability of the system is diminished: a fixed proxy provides a single point of failure for all clients/applications using it. Thirdly, the clients have to be manually configured. This can lead to configuration errors, especially if the users are mobile and changing access network. Finally, because of its static nature, load-balancing (cpu, network) between several proxies is difficult to achieve.

## 3 The March framework

### 3.1 Overview

The Mobile Aware Server Architecture is a distributed system, which adapts content and its delivery to client and network characteristics. The framework operates in three main dimensions:

- Protocol enhancements are mechanisms to enhance the performance of a protocol over a particular network path. A typical example is the Berkeley snoop scheme [4] which improves TCP performance over lossy links such as wireless links. Recent work at the IETF in the Performance Implications of Link Characteristics (pilc) working group summarizes the advances in this area. A more complete survey of available techniques can be found in [2]. Proxies can also implement tailored communications protocols to cater for particular link characteristics. Examples of such systems is presented in [14].
- We refer to content distillation as a mechanism that reduces the bandwidth requirement of data objects by applying transformations that utilize the knowledge of the content semantics. An example of content distillation is bitrate reduction in streaming video by dropping entire frames or reducing the resolution. Distillation is often used in situations involving low bitrate networks such as GSM [7, 8].
- Media transcoding is the transformation of one media type to another, e.g. HTML to WML. This type of transformation is often used when the terminal characteristics do not allow to present the content in its original format.

The type of adaptation to be performed for a given session<sup>1</sup> depends on the *situation* of the session. We define a situation as the set of conditions under which a session operates. These conditions include but are not limited to: the type of access network the client is using, the available and/or bottleneck

<sup>1</sup>The notion of session is dependent on the application used. For example, in the case of an audio-on-demand application, the session is defined by the duration of the transmission and presentation of the audio file. In the case of a web browsing application, the session could be defined by the duration a user is using a particular website services

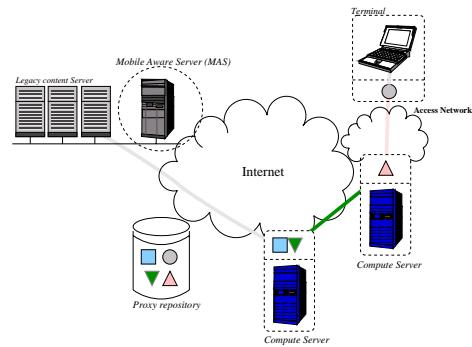


Figure 1: the March framework architecture

bandwidth between the client and the server, the characteristics (CPU/Memory/Screen Size) of the client, and finally the user's preferences.

One of the key design goals of MARCH is to allow service composition. Service composition requires proxy functions to be concatenated to form a global, multifunctional service. An example of such a service is a "content distillation and encryption" service. The architecture needs to maintain the integrity of the system if multiple proxy functions are carried out simultaneously. For example, in the case of the "content distillation and encryption" service, the content distillation needs to take place before the data is encrypted. Once the data is encrypted, functions requiring knowledge of the data semantics cannot be used as these semantics are hidden. In the March framework, the decision as to which proxy modules to use and in which order to use them is made centrally. This allows the framework to maintain the integrity of the system.

### 3.2 Architectural view

The architecture of the March framework is illustrated in figure 1. It is composed of four main components distributed across the network. In this section, we describe these components and their functional interrelationship.

#### The Compute Servers

Content adaptation for a given situation is realized by proxy-modules that operate within an execution context called Compute Server. Depending on the type of applications, duration of sessions and preferences of the operators, proxies can be dynam-

ically inserted into the Compute Servers' execution environment on a per-session basis. It is also possible for Compute Servers to have some fixed components or even combining fixed and dynamic components. The Compute Servers can be located anywhere in the network, at the client and/or at the server. There are several possible locations for Compute Servers such as:

- Network boundaries / performance discontinuities
- Content Provider premises
- Within the access network ISP
- At a third-party service provider specialised in "Compute Server" services.

### **The Mobile Aware Server (MAS)**

Located under the same administrative authority as the content server, the MAS performs an informed decision of which adaptation to perform for a given situation, and where this adaptation should occur (at which Compute Servers). The type of content adaptation to be performed include which proxy modules to use and in what order they should be executed. Another important question is where they should be placed. The answer to this question depends on the type of content adaptation being performed: for protocol enhancement functions, most proxy functionalities require low-level operating system support [2], this makes them unlikely to be downloaded on a per-session basis. For content adaptation proxies, the metrics on which Compute Server to choose for a particular session depends on the network condition e.g. bandwidth, delay between different Compute Servers, the content server and the client. The choice of which Compute Servers to use is not easily resolved and the issue has been partially addressed in proposals such as [9]. The actual details of Compute Server discovery mechanisms and choosing Compute Servers are left for future research.

The MAS contains the logic necessary to make an informed decision about what actions should be taken to tailor the media content to the situation. The input to this process is the information that the client provides at the session request: Terminal capabilities, network characteristics and user

preferences. The MAS adds its specific knowledge about the media and the server architecture (e.g. information about a distributed server replicated in different countries) and makes the decision. The properties of this decision process are currently under investigation.

A way of realizing this decision process is to use a static configuration determined by human decision. The rationale is that the content provider only can decide which particular situations it is willing to cater for and take specific actions for these. The server administrator can be provided with an graphical user interface, through which s/he can pre-determine actions to be taken given the input parameters. Each particular situation that the Content Provider is willing to cater for would be described by using this gui, and stored in a database at the MAS. This way the feasibility of implementing and successfully administrating the service can be greatly improved. We believe that the human input of what type of adaptation to perform and how to compose the services makes our framework unique, and greatly increase its feasibility. Although as presented here it can arguably limit the granularity of the service the server can provide to different clients, we are currently investigating methods to aggregate and generalise the proxy configurations defined by the MAS operator.

### **The MARCH client Entity (MCE)**

The MCE is located on the client device. It is responsible for capturing the terminal capabilities, the access network characteristics and the user preferences. Additionally, it plays an important role in stream handling as it allows the application traffic to be handled by the correct proxy. In some cases, it may be required that the client also runs a light Compute Server to house proxy functionalities (e.g. decryption or decompression)

### **The proxy repositories**

These store the proxy modules and can be located at the content server, the access network or at any other administrative domain. Existing web content distribution procedures can be used to reduce the access latency to the proxy modules.

The location of the Compute Servers and proxy repositories are flexible. MARCH only provides the

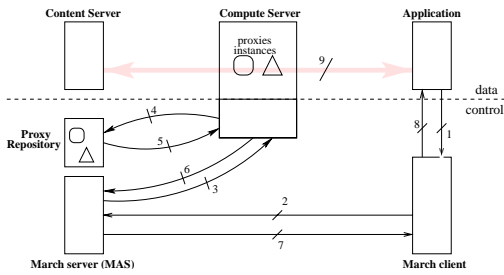


Figure 2: Session establishment operation

framework to interconnect the components. This means that different parties can independently provide performance enhancing proxies. For example, an access Provider may provide some computational power as a service and some proxies to adapt content or protocols to suit its own access network technology, a content provider may provide some proxies to cater for proprietary content types, etc.

### 3.3 Operation

The sequence of events for the establishment of a session within the MARCH framework is shown in figure 2. More precisely, this figure shows the case of a situation for which two proxies need to be instantiated within a single Compute Server.

Applications can either explicitly use the MARCH framework through an API, or transparently if the media semantics and/or session protocol is known to the MARCH client. The API is not specified at this time. When an application requests a service from the network (1), the MARCH client first looks up the corresponding content server, and determines if it is equipped with a MAS. If this is the case, the MARCH client requests a session (2), transmitting the user's preferences, terminal and network characteristics along with the request. The MARCH Server, upon reception of this request, performs a simple lookup to decide if it can handle the request. Depending on the situation, one or more proxy-modules may be required to be instantiated in one or more Compute Servers, or the request can be directed through appropriate previously instantiated proxy(ies). In the first case, as shown in figure 2, a signaling phase takes place between the MAS and the elected Compute Server. Firstly, the MARCH Server requests the Compute Server to instantiate the cho-

sen proxy-modules (3). The Compute Server then fetches the modules (4,5). Once they are instantiated and configured (6), the MARCH server returns to the MARCH client the first-hop proxy information (7). Finally, the MARCH client forwards this information to the application (8). The applications can then receive/request the data from the content servers, transparently through the proxy-modules (9).

The terminal capabilities transmitted in the first request (2) can either be entered manually on each terminal by the user, or can be automatically detected by the server based on the application request as described in [12]. The access network characteristics can either be detected using various operating system specific methods in the client host, or using network monitoring techniques.

### 3.4 Discussion

Using MARCH has a number of advantages for both end-users and content providers. Compared to existing solutions, MARCH enables support for a much more diverse set of device, network and media combinations and the granularity of the service quality improvement is considerably higher. For the content providers, the main advantage is that MARCH broadens the base of devices capable of using the content without any effort from the providers. In section 2, we listed some of the problems with using static proxy solutions. Here, we discuss how MARCH will alleviate these problems and provide a better service for the end-user.

First the useability of the system is far superior to that of the fixed proxy architecture. Since there is no intermediate proxy with a fixed location in MARCH and since the application will know the through user input, there is no need for the users to obtain address information from a network administrator prior to starting a session. Furthermore, users do not have to manually configure settings for the proxies, nor do they have to configure any applications with the proxy address. Finally, when the user changes access network, there is no need to reconfigure any parameters to reflect the proxy location.

MARCH uses general-purpose proxy modules that are concatenated to build up the desired service. Thus, it is very easy to add new functionality by simply adding new modules. Such an

open system allows third party vendors to develop these modules and distribute them when, for example, new media types or better algorithms are introduced. Device manufacturers can also develop proxy modules to perform specific tasks optimised for new devices that are introduced to the market. The problem of end-to-end security and proxies can be solved in MARCH by the MAS being able to place the proxy functionality at the server and the client.

The MAS is able to make an informed decision about ordering of proxy modules to preserve system integrity as mentioned in section 3.1. Since MARCH allows concatenation of proxies, it is possible for end users to enjoy multiple functions simultaneously. For example, it is possible to use a filter that strips advertisements of web pages, a filter that reduces the size of images and transcodes them from colour to greyscale while at the same time using end-to-end encryption.

#### Advantage of the server-centric approach

We argue that the server centric control approach of MARCH, one of its distinctive features, has many advantages over other schemes.

- If we consider the copyright issues surrounding content transformation in the Internet, it is obvious that a server-oriented decision is superior since the transformation is done under the control of an authorised distributor for this content [12].
- When dealing with proprietary media data formats, the content server might be the only entity able to understand the semantics of the data being transmitted, it is therefore the most suitable node to control content transformation
- The MAS has access to all information about the media before configuring the session. An example of additional information not available to clients is sub-encoding. It is impossible for a client to determine the actual coded bit rate of an mp3 file, only by looking at the mime type or file extension. The server however, has got easy access to such information.
- The server has got knowledge of its own architecture/state. Clients that want to contact

a server do not have any information about whether or not the server is distributed over several different physical locations. Since the server has this knowledge, it can much better optimise network paths and therefore critical parameters such as response times.

- The security model is improved: for sensitive content, the server is in control of what type of transformation is being performed, and where this transformation occurs. For example a banking application could use encryption along public network paths, perform decryption and content transformation in a secure enclave, e.g. one of the bank branches, and re-encrypting the data before leaving the secure enclave. A more comprehensive study of the security matters related to proxy architectures can be found in [13].

## 4 Conclusion and future work

In this paper we have introduced MARCH, a framework for adapting media content to suit the operational environment for heterogeneous devices and networks. MARCH exhibits several advantages over traditional static proxy server solutions, for both content providers and end-users. We believe that the framework can play an important role in enhancing the quality of Internet services experienced by mobile users.

Some functionalities of the framework remain for future research. The Compute Server discovery mechanisms, the security model between {Compute Servers, MAS, clients, and proxy repository}, the decision algorithm for how to choose the most suitable Compute Server and proxy modules remain unsolved at this time.

## References

- [1] Opes - open pluggable edge services. <http://www.ietf-opes.org/>.
- [2] Performance enhancing proxies. <http://search.ietf.org/internet-drafts/draft-ietf-pilc-peg-05.txt>.
- [3] E. Amir, Steven McCanne, and Randy Katz. An active service framework and its applica-

- tion to real-time multimedia transcoding. In *proc. of SIGCOMM 98*, 1998.
- [4] Hari Balakrishnan, Srinivasan Seshan, Elan Amir, and Randy H. Katz. Improving TCP/IP performance over wireless networks. In *Proc. of 1st ACM Conference on Mobile Computing and Networking*, Berkeley, California, November 1995.
- [5] A. Cohen, S. Rangarajan, and N. Singh. Supporting transparent caching with standard proxy caches. In *Proc. of the 4th International Web Caching Workshop*, March 1999.
- [6] The ICAP forum. Internet content adaptation protocol. <http://www.i-cap.org/>.
- [7] A. Fox, S. D. Gribble, Yatin Chawathe, and E. A. Brewer. Adapting to network and client variation using active proxies: Lessons and perspectives. *IEEE Personal Communications*, August 1998.
- [8] Michael Fry and Atanu Ghosh. Application level active networking. *Computer Networks*, 7:655–667, 1999.
- [9] Atanu Ghosh, Michael Fry, and Jon Crowcroft. An architecture for application layer routing. In *IWAN 2000*.
- [10] Per Gunninberg and Aruna Seneviratne. Services and architectures in the next generation internet using dynamic proxies. In *FTF99*, Beijing, China, December 1999.
- [11] R. Keller, S. Choi, M. Dasen, D. Decasper, G. Fankhauser, and B. Plattner. An active router architecture for multicast video distribution. In *IEEE Infocom 2000*, March 2000.
- [12] Wei-Ying Ma, Ilja Bedner, Grace Chang, Allan Kuchinsky, and HongJiang Zhang. A framework for adaptive content delivery in heterogeneous network environments. Hewlett-Packard Laboratories, 2000.
- [13] M. Portmann and A. Seneviratne. The problem of end-to-end security for proxy-based systems. In *Proc. of Protocols for Multimedia Systems, PROMS2000*, Cracow, Poland, oct 2000.
- [14] Ranil De Silva. *PNUT: Protocols configured for Network and User Transmission*. PhD thesis, School of Electrical Engineering, University of Technology, Sydney, 1998.
- [15] David L. Tennenhouse and David J. Wetherall. Towards an active networks architecture. In *proc. of Multimedia Computing and Networking*, San Jose CA, 1996. Network Systems group, MIT.
- [16] Mark Yarvis, An-I A. Wang, Alexey Rudenko, Peter Reiher, and Gerald J. Popek. Conductor: Distributed adaptation for complex networks. Technical Report CSD-TR-990042, UCLA, August 1999.