Content Adaptation and Generation Principles for Heterogeneous Clients

Tayeb Lemlouma and Nabil Layaïda

OPERA Project, INRIA Rhône Alpes
E-Mail: Tayeb.Lemlouma@inrialpes.fr, Nabil.Layaida@inrialpes.fr

Abstract

In this paper, we propose a general framework for device independent authoring and presentation. Our approach relies on negotiated adaptation and generation techniques. These techniques allow the creation of customized presentations for different clients starting from a single and more abstract content representation. We focus on some key aspects of this framework through the learned lessons from an experimental system called NAC (Negotiation and Adaptation Core) under development in our project. A particular attention is given to the document model, the document transformation and media adaptation process. The role of the proxy in such a framework is also discussed.

1. Introduction

Providing a suitable content and presentation for different clients in heterogeneous environments is becoming increasingly important today. There is already a plethora of exotic electronic devices such as pagers, PDAs, color cellular phones and there is no sign that the diversity of their characteristics will diminish anytime soon. Making the web content useful on such a range of devices and user agents is challenging and still very hard to achieve. At the lower layers, one of the basic requirements of a successful framework is the provision of a minimal knowledge about clients, servers and network contexts. Starting from such knowledge, efficient mechanisms are needed to deliver the best service in the best possible manner.

The design of complete frameworks with sufficient performance depends widely on to the content representation and model initially used. This has a direct impact on the flexibility exposed during the content manipulation (transformation, adaptation, etc.).

The initial content representation (i.e. the content creation) is not the only factor of primary importance in the process of content delivery. The transformations are the other factor as they can be modularized and re-used in different situations and contexts. The negotiation and adaptation core of the NAC system, uses such a scheme and includes an adaptation layer that operates on the original content sent by the server in a proxy (or server) based architecture. The adaptation process is controlled using a negotiation strategy that allows to reconcile client limitations to servers adaptation capabilities following the notion of profiles (document profile, client profile, etc.). The adaptation layer includes media adaptations such as image transcoding and structural (or tree) transformation using XSLT for instance.
This paper presents some device independent authoring principles from the perspective of content generation using adaptation techniques. Along the presentation of the NAC system, we highlight some key issues and difficulties encountered during the development of the system.

2. Choosing a Document Model

Choosing a general purpose and suitable device independent content model is one of the most important steps in designing an efficient content delivery. The document model influences future scenarios and adaptation strategies applied by the content server. The flexibility and the efficiency of the adaptation process depends widely on the potential left to the original content model. Depending and that model, the achievement of a successful content delivery for different contexts can be easy, difficult or may become sometimes impossible.

This problem can be illustrated clearly when trying to apply an adaptation architecture for the actual Web content. In fact, it is very difficult to process the Web content since it does not respect rigorously a clean syntax at a first place, and thus can not be processed correctly. One possible solution is to "clean" the content first, before its processing, using intermediary processors. Tidy utility [1] represents a good example of a simple, but very useful utility, for cleaning HTML content by fixing syntactic errors automatically. For instance, by adding the missing mark '/' in end tags for anchors, recovering from mixed up tags, etc. Moreover, it allows obtaining a well formed and a valid XML structure of the input document when setting the configuration option 'output-xml' to yes, which helps XML transformation methods to operate correctly.

Unfortunately the task of cleaning the content or making legacy content well structured is not trivial. This requires inferring missing data (structural and formatting), restructuring non structured data and sometimes creating new structures missing initially. In some cases, the conversion requires to correct some mistakes introduced when authoring the original document, the correction -when it is possible- can result sometimes in new content which does not necessarily correspond to the intent of the original authors.

Several domain specific document models are available today and are used in different applications. These models have different functionalities and characteristics and do not have the same expressive power. The presentation description of such models (i.e. the way used to describe and write the presentation in a document) has many sides which differentiate these models and allow comparing them. The presentation is generally related in some way to the logical structure of the document, thus a natural way is to use structure driven transformations such as XSLT.

Some models are more suitable for particular applications. For example SMIL and XHTML + SMIL [5][15] can be used more efficiently than HTML for multimedia representations where the timing aspect is central. In heterogeneous environments the clients reside on a wide range of devices ranging from desktop to ubiquitous information appliances such as personal device assistants, mobile phones, digital televisions, etc. These devices differ in terms of hardware and also in terms of software: two identical devices can use different operating system with different functionalities. This implies that identical clients in terms of hardware may not be capable to render the same presentation. Furthermore, errors can occur when the client try to display the server content and sometimes the client can not display the content at all. This kind of problems is due to:

a) The original content characteristics, and

b) The client capabilities (software user agent and hardware capabilities).

The content model gathers two aspects: the structure of the content and the text media but also other media included by reference (images, audio, video). In many cases, the encoding of the

external media (images, video, audio, etc.) is complex and may cause serious problems for some clients especially those having limited processing capacities.

Therefore, original content such as external media can be useless for clients with limited capabilities. Authors who aim to broaden the access of their content for heterogeneous devices should not make too much assumptions of the capabilities client device.

Thanks to some recent efforts, authors can choose the appropriate document model for poor or limited devices. These models are called language profiles designed specifically for particular capabilities of the clients. Among these language profiles, we find the SMIL Basic language profile [4], which consists of a reduced subset of the full SMIL modules [5]. The defined subset can be supported by a wide variety of SMIL players even those running, for instance, on mobile devices presenting limited resources such as : small displays, few number of supported network transactions, limited input methods, etc. An example of the SMIL Basic applications is the one used by the 3GPP in the scene descriptions of the PSS clients and servers [16]. PSS SMIL collection includes the SMIL 2.0 basic language profile plus three additional modules.

Other languages profiles exist such as the SVG Tiny and SVG Basic [14], compact HTML [12], etc. These languages are useful and can be used either by limited or 'rich' clients. Authoring the content in a flexible model helps severs to provide their content for a variety of users by enabling rich adaptation mechanisms of the original content and making them easy to achieve either at the server side or the intermediary proxies.

3. The Modularization Principle and its Application in SMIL

Modularization represents an efficient approach to define the capabilities of the client. The main goal behind modularization is the define, on a module basis, what particular profile to use for every terminal. The modularization comes also with an extensibility framework which allows to scale the capability of a given language profile. This allows a smoother transition between minimal and full blown language profiles.

In addition to the benefits of the modularization at the server side (i.e. the content provider), the concept allows to describe the user capabilities in terms of supported modules and functionalities. This approach allows applying different selections and achieving custom adaptation according to the target context.

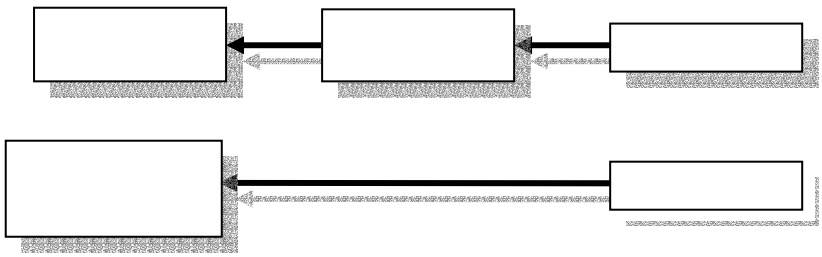
As we are targeting the generation of the content, the SMIL language [13] and its modularization framework represents a good approach to follow. Indeed, SMIL satisfies the need of ensuring definitions and descriptions of abstract functionalities collections either in the client or the server side. SMIL [5] offers extensibility for encoding several aspects of multimedia presentations, it specifies what media items will be presented, where and how. It also accounts for whom the media is played and, the most important in our context, how the presentation is adapted for different end users on different contexts. This last point can be ensured, partly, using the 'SMIL variants selection' achieved within the document without the need of the application of additional processing. The SMIL switch element and the test attributes [2] specify how to make the selections among a set of alternatives for inclusion at individual locations in the temporal hierarchy of the SMIL content [11]. Alternatives can be used, for example, to give different formats of a media object according to client preferences and capabilities.

SMIL selection mechanism represents a very simple model ideal for devices that present several limitations and constraints. Furthermore, the different objects used in SMIL

presentations (and which constitute basic elements of any multimedia document) can be stored anywhere on the Web and acceded by mean of URL's. Actually, the SMIL modules description knows an important use, such as by the third Generation Partnership Project (3GPP) [16], or in the Multimedia Messaging Service as the presentation language for minimal presentation descriptions [9].

4. Intermediary Proxy Solution: NAC application

The negotiation and adaptation core, called NAC, is an architecture developed in order to provide a solution for the delivery of multimedia content in heterogeneous environments. The goal of the architecture is to enable the delivery of content for a wide range of clients that can include devices from desktops to ubiquitous appliances. For content generation, NAC uses dynamic and static adaptation of the server content. The adaptation is controlled using an adaptation and negotiation module (called ANM), and an optional module on the user side (user context module or UCM). It allows to enrich the ANM knowledge about the client description in terms of profiles. NAC default organization belongs to the proxy-based architectures category, but the proxy entity can be omitted by using (installing) ANM at the original server side which causes some differences in the global behavior (Figure 1). These differences are directly related to the fact that in a server based architecture; ANM can have a total control of the server content including existing variants and their characteristics.



One of the important difficulties encountered in NAC is the handling and processing of the server content encoded according to a non adaptable model (example: a non markup language) or one which does not validate against its model (example: a non valid HTML page). Unfortunately, this corresponds to a huge amount of legacy content available on most of the web servers (see section 2).

4.1 Proxy Role

The proxy architecture which consists to add a third entity between the server(s) and the client(s) represents a good approach to address the heterogeneity of clients and servers. Indeed, in a proxy-based architecture the network platform is not modified and all the environment characteristics that already there are taken into account.

In the context of content generation by adaptation, the proxy is the entity responsible of retrieving client requests and contexts and performing possible adaptation on the content received from the server. The generated content is then sent to the client with respect to its characteristics. The proxy can transform existing multimedia content and thus existing content does not have to be produced in multiple versions. All the proxy tasks are designed to behave transparently to clients and content servers.

4.1 The Versioning Principle: a static content adaptation approach

In some situations, the content server maintains different variants of the same original content in order to provide them in different contexts. For example, we find some content providers that store several versions of original videos with different sizes and according to various connection speeds (using modems, LAN, etc.).

Authoring the content in different variants can be useful in some particular cases where no transformation methods of the original content are available or when existing methods generate a non understandable content by the target context. A good example of this is the adaptation of images for different display sizes. In [7] we have introduced an XSLT transformation of HTML to WML that uses the image versioning in order to substitute the original images by their equivalent in wbmp format (the format which is generally used in WML documents).

Maintaining variants of each kind of document and media is costly and has various disadvantages:

• It requires a lot of memory storage space on the content server, especially with large amounts of documents (a great data base for example).

• It processing intensive for document authoring and updates.

Furthermore, it is impossible to predict beforehand all of the different types of devices and new ones may appear later requiring an additional burden on the entire process. This shows that both the client and the server must cooperate in the global architecture in order to offer the best possible scheme.

4.2 Adaptation Process Control: a dynamic content adaptation approach

Unfortunately, many actual content servers do not consider client characteristics and contexts when delivering their content. We carried out a simple test using a personal device assistant, and we requested an image resources from different servers. The requested images, which are encoded in different sizes and formats on the servers, were received on the PDA without any adaptation that takes into account the device display limitations. The received resources are for large display areas and on the PDA screen they cannot be displayed entirely without the use of the player scrollbars.
Here, the useful information about the client context and which was not considered by the content server concerns the displaying capabilities. They were conveyed inside the HTTP request using the two HTTP header fields: "UA-color: color16" and "UA-pixels: 240x320" but did not appear have any effect.

4.2.1 Strategy

The way in which clients use the server content is different from a client to another. A client requests the content its device (PDA, laptop, phone, etc.) that has its specific characteristics and capabilities. Consequently, the presentation provided by the content server, which has its own semantics, should not be the same for all these devices. The content negotiation concept aims to guide content servers to deliver the appropriate content according to the user context, i.e. the client capabilities and the user preferences. For example, we can have a client that uses a PDA to access to HTML pages with French as a preferred language. In this case, a good negotiation strategy must end to the delivery of small HTML pages (which takes into account the client capabilities) with a content written in French.

Generally, in an architecture composed by at least a client and a content server, a content negotiation solution requires the following basic elements (Figure 2):

a) A description tool of the context in which the content is used: such as the description of the client context, the server capabilities, the document profile, etc.

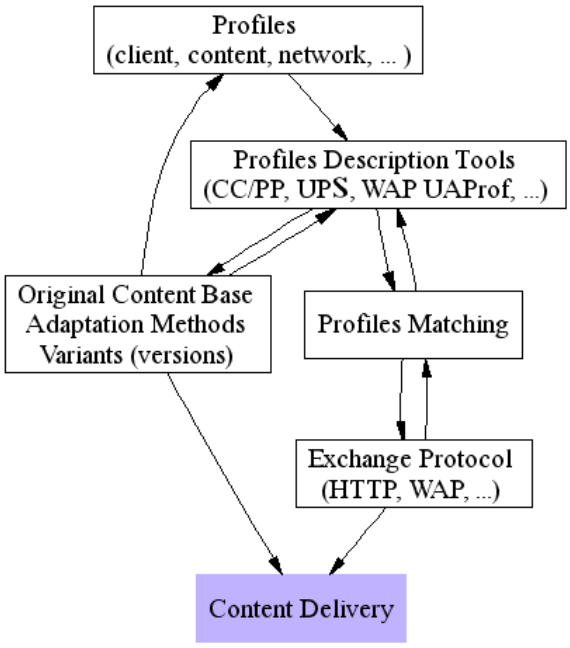
b) An exchange protocol: a well determined dialogue and request format used in the exchange of control messages and the communication of the user context to the server or other entities.

c) Adaptation methods and content versioning: used to adapt or substitute the content with the appropriate variant.

d) A matching strategy: an algorithm which is applied generally at the server side and which aims to match the different profiles (clients, document, server, etc.) in order to determine the best service context and adaptation methods.

Content negotiation techniques are applied mainly following two ways:

A- Variant selection: consists of choosing the best variant on the content server on behalf the user agent. The selection is applied on the available variant list and based on variants description and the user requirements. Selection parameters include the language, the media type, the char-set, etc. The decision of the selection can be determined using an algorithm that covers the different possibilities or simply using a formulae that combines different selection factors and returns the presentation level of a given variant [10][3].



B- Content adaptation: in many situations the available content can not be sent directly to the client because of the content nature or the client characteristics. In such case, the content server can be met only after applying some more complex transformation. The adaptation process can be in the form of a program, a script, a XSLT style sheet, etc. Adaptation techniques belong to two categories:

1) ***Media resources transformation category***: in this category, we find transformation methods that concern the media adaptation like image and video adaptation (color reduction, resizing, etc.), media transcoding, and other methods that operates directly at the encoding level.

2) ***Structural transformation category***: concerns transformations that are applied on the global document organization or logical tree. An example of such applications: transforming HTML to WML, filtering HTML documents, transforming XML to SVG, etc. A structural transformation can either keep the same media resource used by the original document, filter it or use an external media transformation to adapt the media for the target context.

The set of the user context requirements can be seen as a set of *constraints* that the content provider must satisfy in order to find an agreement between what the client demands and what the server can provide. In our approach, the constraints resolution strategy is achieved by adding progressively the constraints to the original content. Finally, it ends with a representation which corresponds to the content to be delivered.
In order to achieve an efficient content adaptation control, the environment constraints must expressed in a manner to cover a wide range of contexts, possibly all of them. Constraints must:

1- Describe enough flexibility, in order to avoid empty solutions.
2- Ease the resolution strategies.
3- Avoid ambiguity: an constraint expression must end to a unique and clear solution.

The universal profiling schema (UPS) [8] was defined to have a central a role in the generation of adapted content. UPS identifies three main categories of contexts: the client category, the server category and the network category. From the content side, the server category includes the *document instance profile* that describes the document characteristics and functionalities. It includes also the *resource profile* that describes a used media resource and the adaptation method profile that describes an available adaptation method that exist in the server or the proxy side.

In our approach, the client constraints are extracted directly from the HardwarePlatform, SoftwarePlatform and BrowserUA components that exist inside the UPS client profiles. In the RDF bags: OnlySupportedResources, PreferredSupportedResource and NonSupportedResources, we extract additional constraints in terms of capabilities and preferences. During the negotiation matching, the client profile and the document instance profile of the requested content are parsed and the set of included constraints are stored in memory according to their types. The server makes the reference to the document instance profile. According to its content, the server can retrieve -using the exchange protocol- the client resource profile [8] that corresponds to the resource used by the requested content. For example, the server retrieves the client resource profile of the WBMP images if the original requested document uses WBMP images.
The server checks then if the resource (media or document) is supported by the client or not. In the positive case the resource is sent directly to the client without any modifications. In the negative case the server checks if there is any existing version of the resource that can meet the client requirements. Links to the list of versions related to the resource are included in the document instance profile or the client resource profile [8]. If the server succeeds to find a variant that responds to the client requirements, the original resource is substituted by this variant; otherwise the server tries to adapt the original resource. To achieve this operation, the server compares the original resource description (using its profile) and the set of the input requirements of each available adaptation methods included in the RDF bag: InputRequirements of the adaptation method profile [8]. If the resource description matches the input requirements of an adaptation method, the server checks if the output description of this method (included in the RDF bag OutputDescription of the adaptation method profile) matches the client requirements. If yes, the server applies this adaptation method on the

original resource and delivers the created resource to the client. In the negative case, i.e. no adaptation method can be applied, the server sends a negative reply.

5. Content Creation Genericity using XSLT

XSLT (eXtensible Stylesheet Language Transformations) is used to transform an XML document into another XML document. The XSL language (eXtensible Stylesheet Language) specifies the styling of XML content by using XSLT processor to describe how the document is transformed into another document that uses formatting vocabulary.

The problem with XSLT transformations is that a single XSLT style sheet performs a single transformation according to a specific content model. If we require a transformation with little changes in the generated content we need, generally, to rewrite the style sheet. Providing general XSLT style sheets for the various user contexts (or profile) is very interesting because once this is achieved, the corresponding server has just to apply the generated style sheet and provides the adapted content to the user. A solution of this problem can be to concatenate, each time, the original service with the user agent profile and then to apply a style sheet that operates according to the profile parts of the input tree. This solution is very complicated to achieve using XSLT templates. Indeed, in addition to the concatenation task of the two XML documents (the profile and the original document) into one valid XML document and which we must achieved *each time* we have a content to deliver; it requires very complicated processing of the initial tree because the two parts of the input tree (the constraints profile and the original content) are separated. This requires even more processing to achieve the cooperation between the two parties.
The scheme proposed in [6] defines a generic style sheet that admits as input the client profile and generates as output a style sheet. The style sheet allows adapting he concerned content according to the input set of constraint included in the client profile. This approach works reasonably well but unfortunately it is bound to a unique document model (SMIL in this case).

7. Conclusions

In this paper, we have discussed some basic ideas related to the device independent authoring and adaptation framework. We explored some key issues gathered thanks to the implementation of the negotiation and adaptation core (NAC) and we reported the difficulties encountered in the manipulation of content that does not follow appropriate models.
As we have seen, adopting more abstract models helps not only the authoring task but also allows an efficient adaptation control of the content in order to meet different contexts. Other aspects such as the selection of multiple variants should also be considered. Selection mechanisms are generally faster and may complement transformation in the adaptation process.

References

[1] Ragget D. Tidy. http://www.w3.org/People/Ragget/tidy/ , W3C working group, August 2000.

[2] Bulterman D. and Ayars J. *The SMIL 2.0 Content Control Modules*, http://www.w3.org/TR/smil20/smil-content.html

[3] Holtman K., TUE and Mutz A. Transparent Content Negotiation in HTTP. RFC 2295, Network Working Group, March 1998.

[4] Kubota K., Cohen A. and Kim M. SMIL 2.0 Basic Profile and Scalability Framework. http://www.w3.org/TR/smil20/smil-basic.html.

[5] Layaïda N. and Van Ossenbruggen J. SMIL 2.0 Language Profile. http://www.w3.org/TR/smil20/smil20-profile.html.

ent Delivery for Different Contexts. Submitted to the SAINT 2003 Conference.

[7] Lemloumaew ork for Media Resource Manipulation in an A]aptation and Negotiation Architecture. Opera Project, INRIA, August 2001.

[8] Lemlouma Profiling for Content Negotiation and Adaptation in Heterogeneous Environments. http://opera.inrialpes.fr/people/Tayeb.Lemlouma/NegotiationSchema/index.htm, January 2002.

[9] Multimedia Messaging Service (MMS) C 6th 2002. http://www.nokia.com.

[10] Network Working Group. Hypertext Transfer Protocol – HTTP/1.0. RFC 1945: http://www.ietf.org/ rfc/rfc1945.txt, May 1996.

Research Currently Applied on a Global Scale. CWI (Centrum voor Wiskunde en Informatica), NL-1090 GB Amesterdam, The Netherlands.

r Small Information Appliances. W3C Note, 09 February 1998.

[13] W3C. Synchronized Multimedia Integration Language (SMIL 2.0), W3C Recommendation 07 August 2001.

[14] W3C. Mobile SVG Profiles: S Recommendation 30 April 2002. http://www.w3.org/TR/SVGMobile/.

[15] W3C. XHTML: The Extensible Hyper Text Markup Language, A Reformulation of HTML 4 in XML 1.0, http://www.w3.org/TR/xhtml/

[16] 3GPP, Technical Specification Group Services and System Aspects, Transparent end-to-end PSS, protocols and codecs (Release 4), 3GPP TS 26.234 v1.5.1, March 2001.