



NAC, une architecture pour l'adaptation multimédia sur le web

Nabil Layaïda, Tayeb Lemlouma, Vincent Quint

► To cite this version:

Nabil Layaïda, Tayeb Lemlouma, Vincent Quint. NAC, une architecture pour l'adaptation multimédia sur le web. *Revue des Sciences et Technologies de l'Information - Série TSI: Technique et Science Informatiques*, 2005, 24 (7), pp.789-813. inria-00423067

HAL Id: inria-00423067

<https://inria.hal.science/inria-00423067>

Submitted on 9 Oct 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

NAC, une architecture pour l'adaptation multimédia sur le web

Nabil Layaïda — Tayeb Lemlouma — Vincent Quint

INRIA Rhône-Alpes
655, avenue de l'Europe
F-38334 Montbonnot

Nabil.Layaïda, Tayeb.Lemlouma, Vincent.Quint@inria.fr

RÉSUMÉ. Le web évolue vers des contenus intégrant des médias de plus en plus riches et variés accédés à l'aide d'appareils très divers à travers de multiples réseaux. Cet environnement très hétérogène, mobile et changeant nécessite que les informations multimédias envoyées par les serveurs soient adaptées aux conditions précises de leur utilisation. Dans ce but, un ensemble de méthodes, langages, formats et protocoles sont développés, notamment par le W3C. L'architecture d'adaptation NAC présentée dans cet article a été définie et mise en œuvre sur la base de ces techniques, en mettant l'accent sur la répartition des traitements d'adaptation, sur les modèles de description de l'environnement, sur les protocoles de négociation et sur les transformations de contenu.

ABSTRACT. The web is evolving towards richer contents and diverse media that are accessed with different devices through multiple kinds of network. This heterogeneous, mobile and changing environment requires that multimedia information delivered by servers be adapted to the actual conditions of use. For that purpose, a number of methods, languages, formats and protocols are developed, especially by W3C. The NAC architecture presented in this article was designed and implemented based on these technologies, focusing on adaptation processing, on environment description models, on negotiation protocols, and on content transformations.

MOTS-CLÉS : world wide web, multimédia, terminaux mobiles, adaptation, description de contexte, transformation.

KEYWORDS: world wide web, multimedia, mobile devices, adaptation, context description, transformation.

1. Introduction

Le web a profondément changé depuis ses origines. L'époque est loin où on partageait de simples pages de texte et quelques images à partir de stations de travail et de PC connectés à l'internet. Des évolutions particulièrement marquantes ont eu lieu ces dernières années dans le domaine des contenus et dans celui des moyens d'accès réseau.

1.1. *Contenus multimédias*

Les contenus du web comprennent maintenant des médias continus, qu'il s'agisse de vidéo, de son ou d'animations. Ces contenus sont souvent utilisés isolément, comme un morceau de musique ou un clip vidéo. Il s'agit alors de ressources monomédia. Mais on trouve aussi de véritables documents multimédias, où plusieurs ressources de différents médias sont intégrées, se complètent mutuellement et sont mises en relation étroite les unes avec les autres.

Ces documents multimédias combinent des médias discrets, comme le texte ou les images fixes, avec des médias continus. Les différents fragments d'information sont reliés par des relations temporelles, spatiales et navigationnelles qui déterminent leur enchaînement dans le temps, leurs positions respectives dans l'espace d'affichage, les différents parcours que l'utilisateur peut effectuer, ainsi que les autres moyens d'interaction dont il dispose. Ce type de document est typiquement mis en œuvre dans le web à l'aide du langage SMIL (Synchronized Multimedia Integration Language) (Ayars, 2001), qui permet de décrire les multiples aspects des documents multimédias.

1.2. *Moyens d'accès*

Les moyens d'accès ont également beaucoup changé depuis les débuts du web, qu'il s'agisse des terminaux ou des supports de communication. Les réseaux sans fil se sont développés et se sont intégrés à l'internet. Ce sont à la fois des réseaux de téléphonie mobile et des réseaux de transmission de données du type WiFi. Tous font maintenant partie de l'infrastructure de communication du web.

Dans le même temps, des appareils de plus en plus variés se sont connectés sur ces réseaux, comme les téléphones cellulaires, les assistants personnels, les téléviseurs, les consoles de jeux et toutes sortes d'appareils embarqués. Tous sont des terminaux multimédias, dans le sens où il gèrent plusieurs médias, et notamment des médias continus. De plus, la plupart accèdent à des réseaux mobiles. On est donc en plein dans le monde du multimédia mobile.

1.3. *Adaptation des contenus*

Cette double évolution des contenus et des moyens d'accès ne va pas sans poser de problèmes, et le premier est sans doute l'adéquation entre contenus et moyens d'accès. En d'autres termes, il s'agit de permettre l'accès dans de bonnes conditions aux mêmes ressources d'information multimédias, à partir de divers appareils, à travers des réseaux aux caractéristiques différentes. C'est le problème de l'adaptation des contenus que nous abordons dans cet article.

L'approche pragmatique de l'adaptation consiste à développer des contenus spécifiquement ciblés vers un type d'appareil, éventuellement dans un format propre à cet appareil. C'est typiquement l'approche retenue pour les téléphones mobiles par le WAP : un format particulier, WML, permet d'organiser l'information d'une façon qui se prête bien à l'accès sur un petit écran et à travers un réseau à faible bande passante. Mais cette approche va à l'encontre des objectifs du web, qui se veut un espace d'information universel, accessible à tous. Le risque, en effet, est la fragmentation qui mènerait à un web pour les téléphones mobiles ne contenant que de l'information produite pour ces appareils, mais aussi à un web pour les téléviseurs, un web pour les assistants personnels, etc. Cela induit de fortes restrictions pour les utilisateurs de chaque type d'appareil : ils n'ont plus accès qu'à une partie de l'information. Cela induit aussi des coûts élevés pour les producteurs de contenu qui veulent toucher un large public : ils doivent créer plusieurs fois la même information sous des formats différents pour les différents types d'appareils.

Une telle évolution peut être évitée par une autre approche qui consiste à favoriser, au contraire, des formats aussi « universels » que possible. Selon cette approche, on produit chaque document une seule fois et on le stocke sous une forme unique. Ensuite, au moment où le document doit être envoyé à un client, on engendre à partir de cette source unique une autre forme qui tient compte des spécificités du *contexte* à cet instant. Le contexte comprend les ressources matérielles et logicielles de l'appareil client ainsi que les moyens de communication utilisés, mais aussi l'utilisateur, ses capacités, ses handicaps et ses préférences, voire les tâches qu'il effectue en parallèle (s'il est au volant, on évitera d'accaparer son regard sur un écran).

La section suivante présente brièvement quelques systèmes d'adaptation décrits dans la littérature et tente de comparer les approches. Puis nous analysons les différents aspects de l'adaptation tels qu'ils sont abordés aujourd'hui dans le web, en particulier à travers les travaux du W3C. La quatrième section aborde un travail plus focalisé mené dans notre équipe et qui se place dans ce cadre du web. Il s'agit d'une architecture d'adaptation répartie appelée NAC (Negotiation and Adaptation Core) qui fait intervenir des *proxies d'adaptation*, en plus des habituels clients et serveurs du web. La dernière section décrit brièvement les réalisations et évalue l'approche NAC à la lumière des expérimentations réalisées.

2. Etat de l'art

NAC s'inscrit dans une ligne de travaux très actifs sur l'adaptation (Lum *et al.*, 2002) (Mohan *et al.*, March 1999). Parmi ces travaux, quatre projets paraissent particulièrement pertinents pour situer NAC.

2.1. Le modèle AHM

Le modèle AHM (Amsterdam Hypermedia Model) (Hardman *et al.*, 1994) a été développé pour fournir un cadre à la combinaison des domaines du multimédia et de l'hypertexte. Dans le domaine de l'hypertexte, AHM s'appuie sur le modèle de Dexter (Halasz *et al.*, 1994) (Gronbaek *et al.*, 1994) et dans celui du multimédia sur le modèle CMIF (Bulterman *et al.*, 1991). Comme le modèle de Dexter, AHM permet l'usage d'un ensemble d'attributs de présentation spécifiques à des composants individuels du contenu. AHM étend cet ensemble d'attributs avec le concept des canaux qui sont des entités abstraites associées à la présentation d'un élément du contenu multimédia. Un canal regroupe un ensemble de caractéristiques qui concernent la présentation de l'élément.

Les canaux du modèle AHM permettent d'adapter une présentation multimédia à différents contextes. Cependant, le modèle ne définit pas de stratégie explicite pour la gestion et la sélection des canaux. Les attributs des canaux, qui décrivent les caractéristiques du contexte, restent limités et ne peuvent pas assurer une négociation dynamique. Le modèle néglige aussi la préférence entre différentes variantes telle qu'elle existe dans HTTP/1.1 avec les variables de qualité. Ce paramétrage simple des présentations hypermédia a été repris et amélioré par quelques autres modèles.

2.2. Le modèle ZYX

ZYX a été développé sur la base de modèles de documents existants, comme SMIL 2.0 et AHM, et à partir des résultats obtenus dans le domaine de la présentation multimédia dans les systèmes de gestion de bases de données (Boll *et al.*, 1999a) (Boll, 2001). En définissant un nouveau modèle de document, ZYX s'est fixé comme objectifs la réutilisation des ressources, l'adaptation et la neutralité vis-à-vis de la présentation.

Pour assurer une bonne réutilisation du contenu, ZYX définit plusieurs granularités de composants : les objets média, les fragments de document et les documents entiers. Les fragments, de taille arbitraire, peuvent être réutilisés dans n'importe quel autre document. Ce principe est similaire à l'approche adoptée par Madeus (Layaïda, 1997). Le modèle offre la possibilité de réutiliser la structure du document en y ajoutant des informations de disposition pour utiliser les différentes granularités d'entités avec les informations ajoutées. Pour sélectionner et identifier les différentes granularités des

entités, le modèle utilise des annotations et ajoute aux entités des métadonnées de description du contenu.

Au contraire de SMIL 1.0 (Hoschka, 1998), ZYX évite de dépendre d'un ensemble prédéfini d'attributs pour guider les mécanismes d'adaptation. L'approche de ZYX consiste à rendre cet ensemble d'attributs extensible et permet ainsi de prendre en charge un contexte riche et une gamme étendue de clients (Boll *et al.*, 1999b). Tirant parti des avantages de SMIL 2.0 et de AHM, ZYX propose une modélisation statique des alternatives de présentation. Le modèle définit également des primitives pour instancier les alternatives au moment où le contenu est demandé.

ZYX ne dispose pas de stratégie de négociation de contenu. L'adaptation est fondée sur les fonctionnalités du contenu lui-même, qui doit être organisé sous forme d'alternatives décrites de manière statique. Ces deux facteurs interdisent toute possibilité de déléguer l'adaptation à des entités externes, comme les serveurs ou les proxies utilisés dans les systèmes permettant une négociation avec adaptation dynamique.

2.3. Le modèle InfoPyramid

Smith *et al.* (Smith *et al.*, 1999) définissent un système extensible de transmission du contenu multimédia qui utilise un nouveau modèle de données, appelé InfoPyramid. Le modèle représente un cadre pour la gestion et la manipulation du contenu multimédia composé de ressources vidéo, image, audio et texte. InfoPyramid manipule les variantes de ces ressources avec différentes modalités et niveaux de fidélité et il sélectionne les alternatives pour adapter le contenu source à différents terminaux. L'approche d'InfoPyramid se fonde sur une description hiérarchique des médias afin de faciliter la recherche et l'extraction du contenu pertinent.

Le modèle comprend un ensemble de classes pour le traitement des quatre modalités (vidéo, image, audio et texte) et des méthodes pour le transcodage/transmodage des ressources. Ces méthodes sont la conversion de texte en audio, le transcodage des images et de la vidéo, l'extraction d'image à partir d'une vidéo et le résumé et la traduction de texte.. Le modèle se limite au traitement des objets média qui peuvent être utilisés dans une présentation, mais n'aborde pas les documents multimédias qui organisent ces objets. En effet, il offre des possibilités de substitution et de transcodage de quelques types de médias indépendamment des relations qui peuvent exister entre les ressources elles-mêmes ou entre les ressources et le scénario de la présentation. Le modèle définit quelques règles pour respecter trois types simples de contraintes, qui sont relatives à la taille des ressources, au temps de chargement et à l'espace d'affichage du terminal.

2.4. MPEG-21

MPEG-21 (Burnett *et al.*, 2003) est une norme élaborée par le groupe MPEG (Moving Picture Experts Group) qui vise à permettre l'utilisation transparente de res-

sources multimédias par toutes les catégories d'utilisateurs à travers une vaste gamme de réseaux et d'appareils. MPEG-21 couvre l'ensemble des processus de création, manipulation, utilisation, gestion et diffusion de documents multimédias et constitue un vaste cadre d'interopérabilité où les éléments d'une application multimédia s'intègrent et interagissent.

Ce cadre inclut les problèmes d'adaptation de contenu, mais ne se limite pas à ceux-ci. C'est toutefois sur ces aspects que cette brève présentation se concentre. Parmi la douzaine de parties qui composent MPEG-21, une est spécifiquement dédiée à l'adaptation : DIA (Digital Item Adaptation) (Vetro, 2004). D'autres contiennent également des éléments d'adaptation, comme par exemple les éléments *choice* de DID.

L'élément de base dans MPEG-21 est le « Digital Item » (DI), qui est un document multimédia avec toutes ses ressources, auquel sont attachés un certain nombre de descripteurs, chaque descripteur couvrant un aspect différent du DI. L'adaptation concerne à la fois les ressources et les descriptions associées. MPEG-21 DIA définit des outils de description pour représenter les capacités des terminaux, les caractéristiques du réseau, celles de l'utilisateur et l'environnement dans lequel celui-ci se trouve (ambiance sonore, luminosité, lieu, date, heure, etc.). Ajoutées à la description du DI lui-même, ces informations permettent de produire un DI adapté aux conditions de son utilisation. Il est important de noter que la norme ne décrit que les outils qui aident à l'adaptation, mais pas les opérations effectives d'adaptation. On peut toutefois noter que, dans la pratique, les descripteurs peuvent être adaptés par des moteurs XSLT, puisqu'ils utilisent une syntaxe XML.

Les ambitions de MPEG-21 sont plus larges que celles de NAC. MPEG-21 couvre par exemple les questions de propriété intellectuelle et de droits, non abordées dans NAC. MPEG-21 est donc très général, alors que nos travaux se focalisent sur le web et tentent de s'intégrer dans les technologies qui y sont déjà déployées. Enfin, MPEG-21 vise à standardiser tout un environnement, alors que NAC cherche au contraire à s'inscrire autant que possible dans l'environnement existant du web.

3. L'adaptation sur le web

Les travaux résumés ci-dessus se placent dans le cadre général du multimédia. Certains prennent en compte l'environnement spécifique du web, d'autres ne le considèrent pas particulièrement, même s'ils peuvent s'y appliquer, au moins en partie. Dans cette section, nous analysons les travaux récents et en cours menés pour l'adaptation des contenus multimédias dans le web. Pour l'essentiel, il s'agit d'un effort collectif entrepris dans le cadre du W3C par des dizaines de chercheurs et d'ingénieurs à travers le monde et auquel les auteurs de cet article participent.

Dans cette approche centrée sur le web, quatre éléments principaux interviennent dans l'adaptation : les formats de documents, la description du contexte, la négociation et la transformation des contenus.

3.1. *Formats de documents*

Pour faciliter l'adaptation, il est clair que le format dans lequel un document est décrit doit faire intervenir le moins possible de caractéristiques liées à la présentation ou à l'appareil de restitution. C'est la direction que prend l'approche XML, développée pour représenter les documents et les données sur le web. Elle sépare le contenu et sa structure logique d'un côté et la présentation de l'autre. Contenu et structure sont représentés dans la ressource XML elle-même, alors que la présentation est décrite dans des ressources séparées, les feuilles de style. Plusieurs feuilles de style différentes peuvent être appliquées à la même ressource XML, qui prend ainsi des aspects différents. Cette propriété de XML offre ainsi une possibilité efficace d'adaptation : on choisit le jeu de feuilles de style qui convient le mieux à un contexte donné.

La notion de feuille de style a été introduite d'abord pour les documents textuels, mais elle s'étend très bien aux documents multimédias. Des formats XML multimédias développés pour le web comme SMIL (Ayars, 2001) et SVG (Ferraiolo *et al.*, 2003) s'appuient sur le langage de style CSS (Bos, n.d.), pour déterminer la présentation effective. CSS de son côté ne se limite pas à définir l'aspect graphique des documents, puisqu'il comporte un module qui permet de déterminer très précisément l'aspect sonore que doit prendre un document textuel restitué par un synthétiseur vocal. Cela permet, simplement en choisissant la bonne feuille de style, d'adapter un document à un appareil graphique ou à un appareil sonore, comme un téléphone, par exemple.

CSS offre également un mécanisme appelé `@media` qui permet de fournir, dans la même feuille de style, des règles destinées à différentes catégories d'appareils : écrans graphiques, projecteurs, imprimantes, téléviseurs ou terminaux Braille. Le logiciel client applique les règles pertinentes pour l'appareil sur lequel il travaille.

En plus des feuilles de style, de nombreux formats XML multimédias disposent de mécanismes favorisant l'adaptation. XHTML propose l'élément `object` pour inclure des ressources externes (des objets média) dans une page web. Cet élément permet de spécifier une hiérarchie de ressources, chacune étant considérée comme une solution de repli si la précédente n'est pas utilisable dans un contexte donné. On peut ainsi inclure une image animée accompagnée d'une version statique de l'image et d'un texte qui décrit l'image, chacun pouvant être présenté en remplacement des précédents. SVG et SMIL disposent d'un mécanisme analogue avec l'élément `switch` qui fournit une liste d'alternatives adaptées à des contextes différents.

Les langages XHTML, SMIL et SVG sont organisés sous forme d'un ensemble de modules décrivant chacun une fonction particulière du langage. Cette modularisation permet d'identifier pour chaque contexte le sous-ensemble de modules supporté par le client. Dans le cadre de l'adaptation, cette connaissance est précieuse, pour la négociation comme pour l'identification précise du langage cible.

L'expérience montre que beaucoup de producteurs d'information n'utilisent pas pleinement l'ensemble de ces mécanismes. C'est pourquoi il est important de déve-

lopper des techniques d'écriture (Hanrahan *et al.*, 2004) et des environnements auteur qui aident et incitent les producteurs à utiliser au mieux les capacités d'adaptation des nouveaux formats XML.. Ces environnements, comme Amaya (Amaya, 2005) ou LimSee2 (LimSee, 2005), mettent l'accent sur la structure logique des documents, fournissent des moyens efficaces pour manipuler le style séparément du document, et donnent accès aux différents éléments qui favorisent l'adaptation.

3.2. Description du contexte

Une fois que le document est dans un format favorable, il faut encore, avant de l'adapter, connaître les conditions précises de son utilisation. C'est le rôle de la description du contexte. Cette description concerne au moins le terminal et certaines préférences de l'utilisateur. Elle peut comprendre aussi les caractéristiques du réseau, les fonctionnalités du contenu et les traitements d'adaptation disponibles. Pour que l'ensemble des applications concernées à travers le web puissent les partager, les descriptions de contexte doivent être exprimées dans un langage largement accepté et doivent être manipulables par toutes sortes de machines et de logiciels. Les formats standard de description de contexte jouent donc un rôle clé.

CC/PP (Composite Capabilities/Preference Profiles) (Klyne, 2004) est un standard de ce type. Bien qu'il soit récent, il semble être le dénominateur commun des applications web mettant en jeu l'adaptation. C'est en fait un cadre de travail qui s'appuie à la fois sur XML et sur RDF (Resource Description Framework), un modèle pour les métadonnées sur le web. CC/PP permet de décrire les caractéristiques logicielles et matérielles d'un terminal, mais aussi les préférences de l'utilisateur. Dans sa version courante, il se limite à ces aspects du contexte.

3.3. Négociation

La description du contexte sert de base à une négociation entre un client et un serveur. Grâce à cette description le serveur peut connaître les contraintes qui lui sont imposées. Il se peut qu'il soit en mesure de les satisfaire toutes et, dans ce cas, la négociation est vite conclue. Mais il se peut aussi qu'un dialogue doive s'engager pour établir un compromis acceptable entre ce que le client demande et ce que le serveur peut offrir, ou simplement pour échanger davantage d'informations de contexte, lorsque celles-ci sont trop volumineuses pour être transmises systématiquement dans leur totalité. C'est le rôle du protocole de négociation de mener ce dialogue.

Dans le web, le protocole le plus utilisé est HTTP. Son rôle essentiel est de transmettre les contenus entre serveur et client. Dans sa première version, il offrait très peu de possibilités de négociation. Tout au plus permettait-il au client d'énoncer quelques préférences (entête `Accept`) sur les jeux de caractères, les langues ou les médias acceptés (types MIME). Il laissait alors au serveur le soin de faire au mieux à partir de cette maigre information. La dernière version du protocole, HTTP 1.1, va plus loin

et autorise un dialogue de négociation. Cela permet aussi d'impliquer le client et le serveur et ainsi de répartir la charge de la négociation.

Même si HTTP 1.1 apporte des progrès significatifs, il reste limité à une approche fondée sur des versions multiples préexistantes des contenus et ne permet pas vraiment une adaptation dynamique fondée sur des transformations d'une ressource unique.

3.4. Transformation

La transformation est la dernière phase du processus d'adaptation. Elle consiste à produire une version bien adaptée au contexte à partir d'une ressource unique, ou d'un nombre limité de variantes. Elle peut intervenir à des niveaux différents : sur les objets média, sur la structure du document et sur certains aspects de sa sémantique.

Le premier niveau de transformation concerne les objets média. On peut ainsi transformer les images et la vidéo en appliquant une réduction de couleurs ou de niveaux de gris, en réduisant leurs dimensions, ou en changeant l'encodage. Une transformation plus drastique est offerte par le transmodage, qui consiste à remplacer une vidéo par une image fixe qui en est extraite, ou une image animée par sa version statique, ou encore une image fixe ou animée par un texte la décrivant.

On peut aussi intervenir au niveau de la structure logique, de l'organisation globale du document. La structure relativement riche d'un document HTML peut être ramenée à une structure XHTML Basic (Baker *et al.*, 2000), une version XMLisée et épurée de HTML conçue pour les terminaux mobiles. Un autre exemple est la transformation d'un format XML abstrait vers un format plus proche de la présentation, comme XHTML ou SVG. Cette transformation de structure peut évidemment s'accompagner d'une transformation des objets média qu'elle utilise. L'outil de choix des transformations de structures est le langage XSLT (Clark, 1999), spécialement adapté aux transformations d'un langage XML vers un autre langage XML, ou vers une autre syntaxe textuelle.

Le dernier niveau de transformation considéré ici est celui de l'adaptation sémantique (Euzenat *et al.*, 2003) (Lemlouma *et al.*, 19-22 janvier 2004). Celle-ci consiste à prendre en compte le sens associé aux éléments structuraux d'un document et aux relations qui les lient. L'adaptation est alors plus ciblée que par de simples transformations structurales, puisqu'elle tire parti du sens associé à la structure. À titre d'exemple, dans SMIL, les éléments *par* et *seq* ont des structures très proches du strict point de vue XML, mais ils portent des sémantiques temporelles fort différentes, celles d'objets présentés en parallèle (*par*) ou en séquence (*seq*) dans le temps. La prise en compte de cette sémantique permet de les traiter différemment lors de l'adaptation.

4. NAC, une architecture pour l'adaptation

On a vu les principes de base de l'adaptation des documents multimédias sur le web, ainsi que les principaux formats, protocoles et langages disponibles aujourd'hui dans ce domaine. Certains de ceux-ci présentent des limitations qu'il faut dépasser pour parvenir à une adaptation de qualité. Par ailleurs, il s'agit d'un ensemble de technologies qui fournissent un certain nombre de moyens d'adaptation, mais il faut encore les mettre en œuvre ensemble.

C'est dans ce but que nous avons conçu et réalisé l'architecture répartie NAC (Lemlouma *et al.*, 2003) qui intègre l'ensemble de ces technologies et réalise un cadre d'adaptation pour le multimédia sur le web. Ce projet nous a permis également d'explorer leurs possibilités et leurs limites.

4.1. Architecture

NAC vise à assurer l'accès adapté aux contenus multimédias, où qu'ils se trouvent sur le web et quel que soit le protocole utilisé pour les transférer. Mais le serveur détenant l'information peut ne disposer d'aucun moyen d'adaptation et utiliser un protocole insuffisant en matière de négociation de contenu. Le client peut également souffrir des mêmes limitations. La solution consiste à interposer un *proxy* entre le client et le serveur. Ce proxy prend en charge l'essentiel de la négociation et de l'adaptation. En contact à la fois avec le client et avec le serveur, il a une vision globale du contexte et est ainsi bien placé pour effectuer une adaptation pertinente. Il peut fonctionner sur une machine dédiée pour offrir des services d'adaptation à une large population de clients et de serveurs qui n'ont ainsi pas besoin d'être modifiés pour bénéficier de possibilités d'adaptation. Il peut aussi résider sur un serveur, soit pour optimiser les performances, soit pour offrir des services plus spécifiques à ce serveur.

Pour permettre une bonne adaptation, il faut un protocole de négociation. On l'a vu, les protocoles de transfert disponibles couramment, comme HTTP, sont généralement insuffisants dans ce domaine, voire inexistantes pour les versions les plus anciennes. L'approche de NAC consiste à séparer la négociation et le transfert de contenu dans deux protocoles différents. Ainsi, même si un client ne supporte pas la négociation, le proxy peut s'engager pour lui, à partir de son profil, dans une négociation avec un serveur et lui retourner ensuite le contenu négocié. Pour cela, le client n'a besoin que du protocole de transfert habituel dans ses échanges avec le proxy.

La description des contextes est une ressource importante que NAC stocke dans une *base de profils*. Celle-ci décrit les terminaux clients, les préférences des utilisateurs, l'état du réseau, les capacités des serveurs, etc. En la plaçant au plus près du proxy qui effectue l'adaptation, on rend celle-ci plus efficace. Le client n'a pas besoin d'envoyer une description complète de ses caractéristiques, puisqu'à partir de la seule indication du modèle d'appareil ou du nom et de la version du logiciel utilisé, on peut retrouver le détail de son profil dans la base. De même, les préférences des utilisateurs

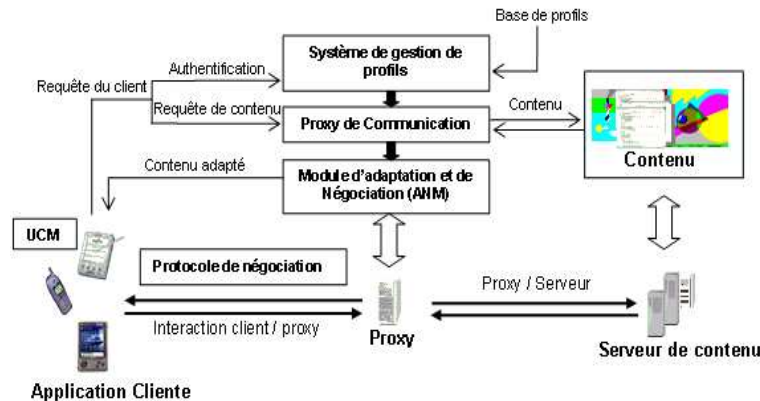


Figure 1. Organisation générale de l'architecture NAC

teurs peuvent être stockées dans cette base, pour éviter de les transmettre à chaque connexion. Cette base doit évidemment être mise à jour régulièrement pour intégrer les nouveaux terminaux et logiciels, pour changer les préférences des utilisateurs ou tenir compte des fluctuations du réseau.

Sur la base de ces principes, NAC est constitué comme le montre la figure 1. L'architecture comporte cinq composants :

Le proxy de communication assure l'accès au contenu et la communication qui se déroulent classiquement entre le client et le serveur. Il reçoit les requêtes des applications clientes et les transmet aux serveurs de contenu. Il reçoit en retour les réponses des serveurs. Il adapte ou modifie ces réponses avant de les délivrer à leur destination finale, les clients qui ont émis les requêtes initiales. En réservant un port additionnel, le proxy assure aussi la communication orientée négociation avec un module résidant sur le client, le module UCM.

Le module d'adaptation et de négociation (ANM) assure l'adaptation de contenu et la négociation. Cela est effectué grâce à l'application d'un ensemble de méthodes de transformation structurale et d'adaptation de contenu. L'adaptation est dynamique dans la majorité des cas et dépend des valeurs que prennent les caractéristiques du contexte, telles que l'application cliente utilisée, les formats acceptés par le client, la taille d'écran du terminal, etc. Le module ANM coopère avec les autres entités de l'architecture afin de prendre la meilleure décision de négociation : choix de version de contenu, choix de méthode d'adaptation, choix de méthode de transmission du contenu final, etc.

Le module du contexte de l'utilisateur (UCM) est utilisé côté client pour permettre une négociation avancée du contenu. Grâce à l'UCM, le terminal peut effectuer une configuration du proxy et du port d'autorisation utilisés dans la négociation. L'UCM participe à la négociation de contenu d'une manière complètement indépen-

dante de l'application cliente utilisée. Ainsi, le terminal peut changer d'application cliente ou utiliser plusieurs applications clientes en même temps tout en gardant la même configuration de l'UCM. Ce module permet aussi de sélectionner le proxy client et d'en changer à n'importe quel moment. L'UCM optimise la négociation afin d'éviter la dégradation des performances des terminaux, dont les capacités sont souvent limitées.

Le protocole de négociation permet de mettre en œuvre une stratégie de négociation avancée. Il définit un mode d'interaction entre le module UCM côté client et le processus de négociation du module ANM. Cette interaction s'effectue par l'échange de requêtes et de réponses. Par exemple, le processus de négociation peut interroger le terminal sur ses capacités et ses préférences courantes ou il peut vérifier si le profil client a subi un changement depuis la dernière interrogation.

Le système de gestion de profils assure l'analyse et la gestion des descriptions de contexte (la description du client, du contenu, des méthodes d'adaptation, etc.) au profit de l'adaptation de contenu. Les contraintes indiquées par les profils sont exploitées afin que les méthodes d'adaptation puissent être sélectionnées et exécutées au mieux. Le système comprend aussi l'exploitation des services qui assurent l'extraction des profils, des parties de profils ou des caractéristiques élémentaires d'un profil donné. A cette fin, il prend en charge l'interaction avec les bases de profils.

Un terminal qui fait partie de l'environnement hétérogène utilise une application cliente pour accéder et utiliser un contenu disponible sur un serveur. Le contenu peut être simple, par exemple sous la forme d'un contenu textuel ou d'une page HTML, ou complexe, par exemple sous la forme d'une présentation multimédia qui utilise des ressources comme des vidéos de haute résolution, des images, etc. Lors de l'accès au contenu, le proxy de communication reçoit la requête de l'application cliente. Il tente d'identifier le profil courant du terminal et son contexte grâce au protocole de négociation qui peut interroger directement le module UCM en cas de changement de profil ou s'il a besoin d'une propriété particulière du profil de l'utilisateur. La connaissance du contexte client peut aussi faire appel au système de gestion de profils pour exploiter les services des bases de profils disponibles.

Le contexte de l'application cliente qui est relatif à la requête courante représente un ensemble de contraintes que le module de négociation et d'adaptation ANM doit prendre en compte avant la transmission du contenu final. Cet ensemble de contraintes est construit à l'aide du système de gestion de profils qui se base sur le contenu des profils disponibles. Le traitement du module ANM peut varier selon la nature des contraintes, les méthodes d'adaptation disponibles, les versions disponibles du contenu demandé et les caractéristiques du contenu source ou du serveur qui détient ce contenu.

Pour faire la correspondance entre les caractéristiques du contenu source, les capacités du système d'adaptation et les exigences du client, le module ANM peut à tout moment enrichir son ensemble de contraintes à l'aide du protocole de négociation et des bases de profils. Il tente d'établir un graphe d'adaptation qui liste toutes les

étapes et techniques d'adaptation qui permettent d'obtenir, à partir du contenu source non adapté, un contenu respectant l'ensemble des contraintes, par l'application des techniques d'adaptation.

4.2. Représentation des profils

Le contexte d'utilisation des documents joue un rôle clé dans l'architecture NAC. Celle-ci comporte :

- un modèle de description du contexte,
- un mécanisme de stockage des descriptions,
- le traitement distribué de l'ensemble des descriptions disponibles.

Le **modèle de description**, appelé UPS (Universal Profiling Schemata)(Lemlouma *et al.*, 4-5 mars 2002), est fondé sur des *profils*, qui sont des ensembles cohérents de caractéristiques décrites dans un langage de marquage déclaratif. Le modèle prend en compte aussi bien les caractéristiques statiques, comme les dimensions d'un écran, que les caractéristiques dynamiques, comme la bande passante instantanée du réseau. Les profils peuvent être créés directement par un auteur, mais ils peuvent aussi être engendrés automatiquement à partir de l'environnement.

Le modèle de description est mis en œuvre comme un ensemble d'extensions de CC/PP (Klyne, 2004). MPEG-21, en particulier sa partie DIA, pourrait être considéré comme une alternative, mais, il y a quatre ans, lorsque ce travail a commencé, CC/PP offrait déjà un cadre de travail adéquat, il présentait les fonctionnalités de base nécessaires, il était spécifiquement conçu pour le web et son déploiement était déjà en cours. MPEG-21 était moins avancé et ne s'appliquait pas directement aux formats des documents considérés (SMIL, notamment).

Le modèle UPS vise à ajouter à CC/PP les caractéristiques qui lui manquent. Il utilise donc RDF qui peut être défini par :

- un ensemble de ressources,
- un ensemble de littéraux,
- un sous-ensemble de ressources appelé propriétés,
- un ensemble de déclarations sous la forme de triplets (p, s, o) où p représente une propriété, s est une ressource et o est une ressource ou un littéral.

Un triplet (p, s, o) peut être interprété comme suit : o est la valeur de la propriété p pour la ressource s . Un ensemble de telles déclarations peut être représenté par un graphe étiqueté dont les ressources et les littéraux constituent les noeuds et les propriétés les arcs. Un triplet (p, s, o) est représenté par les deux noeuds s et o reliés par un arc étiqueté p , dont l'origine est s et l'extrémité o . La Figure 2 donne une représentation graphique de la déclaration `HardwarePlatform` extraite d'un profil écrit en UPS (voir le profil `ups-client-profile9.rdf` de la base de profils de

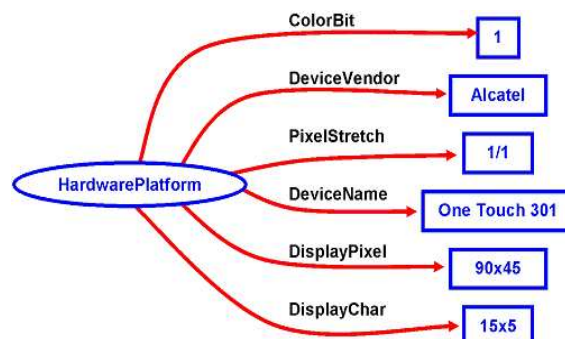


Figure 2. Représentation graphique d'un composant UPS

NAC (Lemlouma, 2003). On peut y lire notamment que la taille en pixels du dispositif d'affichage de la plate-forme matérielle est 90x45.

Un profil UPS exprimé par un ensemble de triplets peut être représenté dans plusieurs formats, notamment XML et N-Triples (Jan *et al.*, 2003). En utilisant le format N-Triples, qui est une représentation textuelle linéaire des graphes RDF, la déclaration (DisplayPixel, HardwarePlatform, "90x45") s'exprime :

```
<HardwarePlatform> <DisplayPixel> "90x45".
```

Le modèle de description UPS comprend six profils différents, chacun étant décrit par un schéma RDF :

- client : caractéristiques de la plate-forme matérielle et logicielle,
- ressource client : formats supportés par le client,
- instance de document : caractéristiques d'un document (format, taille, langue, etc.),
- ressource de document : caractéristiques des ressources (images, vidéos, ...) utilisées par un document,
- méthode d'adaptation : caractéristiques d'une feuille de transformation XSLT ou d'un programme de transcodage (formats d'entrée et de sortie, notamment),
- réseau : type, bande passante, protocole.

UPS décrit ces différents aspects du contexte dans des composants séparés. Un composant comporte différents types de caractéristiques qui sont représentées chacune par un triplet RDF. La Figure 3 montre une partie d'un profil client écrit en UPS avec la syntaxe XML. Cette figure représente le composant HardwarePlatform (plate-forme matérielle) d'un profil client.

Le **stockage des descriptions** est possible à trois niveaux différents : client, proxy et base de profils. Quel que soit le niveau de stockage, les caractéristiques des profils doivent être communiquées aux processus d'adaptation. Cette communication néces-

```

1: <ccpp:component>
2:   <rdf:Descriptionrdf:ID="HardwarePlatform">
3:     <neg:DeviceName>One Touch 301</neg:DeviceName>
4:     <neg:DisplayPixel>90x45</neg:DisplayPixel>
5:     <neg:DisplayChar>15x5</neg:DisplayChar>
6:     <neg:ColorBit>1</neg:ColorBit>
7:     <neg:DeviceVendor>Alcatel</neg:DeviceVendor>
8:     <neg:PixelStretch>1/1</neg:PixelStretch>
9:   </rdf:Description>
10: </ccpp:component>

```

Figure 3. Exemple de représentation de profil en UPS

site un échange de messages à travers le réseau dans le cas où les profils sont détenus par le client (grâce au module UCM) ou par une base de profils. Le premier type d'échange de message fait appel au protocole de négociation appliqué entre le module ANM et le module UCM, alors que le deuxième fait appel à des services web offerts par la base de profils et qu'il utilise le protocole SOAP (Mitra, 2003). Les échanges réseau sont optimisés de manière à véhiculer uniquement l'information nécessaire quand il le faut.

Le **traitement des descriptions** consiste à extraire les informations appropriées et à faire la correspondance entre ces informations. L'extraction des informations peut être statique ou dynamique. L'extraction statique fait appel à une interrogation paramétrée de la base de profils. Cette interrogation est effectuée d'une manière locale ou distante selon le niveau de stockage adopté. Les paramètres de l'interrogation dépendent du contexte courant de la requête émise par l'application cliente. En effet, les caractéristiques du contexte de la requête (type de requête, format du contenu demandé, application cliente utilisée, etc.) déterminent le type des informations nécessaires.

L'extraction dynamique fait appel à l'application de certaines méthodes déjà disponibles pour le système d'adaptation. Ces méthodes permettent de calculer les valeurs de certaines caractéristiques du contexte qui sont généralement de nature dynamique, telles que la bande passante disponible au moment de la réception de la requête, la charge instantanée du serveur, ou la taille de la mémoire disponible sur le terminal client.

4.3. Protocoles

Les différentes entités d'un environnement hétérogène subissent souvent des changements de contexte dus à la localisation des terminaux mobiles, aux préférences des utilisateurs, à l'utilisation de différentes applications clientes, etc. Afin de prendre

en charge ces changements, l'architecture NAC comporte un protocole de négociation pour l'acquisition du contexte client. Ce protocole représente une alternative aux mises à jour du contexte effectuées en passant par la base de profils.

Les protocoles de communication destinés à la transmission du contenu aux terminaux, comme HTTP, sont généralement des protocoles sans états. Ces protocoles sont appliqués entre le navigateur du client et le serveur de contenu. Leur principe est simple : le client envoie une requête et le serveur retourne une réponse. Dans le cas général, le serveur n'a pas besoin de garder d'informations sur le client ou sur la session courante. En effet, une fois la réponse du serveur envoyée (le contenu demandé, un message d'erreur, un message de non modification, etc.), la communication entre le serveur et le client est terminée.

L'objectif du protocole de négociation pour l'acquisition du contexte est différent de celui d'un protocole de communication. Le protocole de négociation assure l'échange d'information de contexte entre le client et le module d'adaptation et de négociation. NAC sépare les deux protocoles pour rendre le système indépendant d'un navigateur particulier et couvrir ainsi une plus large diversité de clients.

Le protocole de négociation de NAC est conçu de manière à optimiser les ressources du client, du serveur et du réseau. L'échange de messages de négociation est minimisé et véhicule uniquement l'information utile. Le protocole prend en compte les changements de contexte et assure l'extraction des éventuels changements. Afin de maintenir les contextes des différents terminaux simultanément, le protocole applique une stratégie de cache temporaire fondée sur les adresses IP des terminaux.

Le protocole définit cinq types de messages :

- GET GLOBAL PROFILE est utilisé par le module ANM pour demander le contexte courant du client,
- OK SENDING PROFILE est utilisé par le module UCM lors de l'envoi de son contexte,
- OK SENDING CHANGE est utilisé par le module UCM lors de l'envoi de son contexte dans le cas où ce dernier a subi un changement depuis le dernier envoi de contexte,
- NO PROFILE CHANGE est utilisé par le module UCM si le contexte du client n'a subi aucun changement depuis le dernier envoi de contexte,
- NO PROFILE ACQUISITION est utilisé par le module UCM s'il est impossible d'extraire le contexte du client (cas d'erreur au niveau du client ou absence de méthode d'extraction de contexte).

Le protocole de négociation est appliqué entre le module de contexte client UCM et le module d'adaptation et de négociation ANM. Le module ANM peut à tout moment vérifier le contexte des différents clients grâce à un fonctionnement multi-tâches de NAC. Le contexte d'un client est décrit selon le modèle UPS et sauvegardé dans le format XML sur le terminal. Le changement de contexte est matérialisé par le chan-

gement que subit le profil UPS. À l'aide du module UCM, le client peut changer son profil à tout moment.

Les changements locaux de profils ne sont pas transmis au module ANM tant que cela n'est pas explicitement demandé. C'est uniquement lorsque le client envoie une requête pour demander un contenu que le module ANM vérifie un éventuel changement de contexte à l'aide de la requête GET GLOBAL PROFILE. Dans le cas où le contexte subit un changement, le module UCM répond par un message OK SENDING CHANGE avec le nouveau contexte, sinon le client répond par un message NO PROFILE CHANGE. Dès qu'un changement de contexte est détecté par le module ANM, le module applique un nouveau processus de négociation sur le contenu demandé.

4.4. *Adaptation*

Le nombre et la diversité des paramètres qui interviennent dans le contenu, les caractéristiques du terminal et les autres éléments du contexte nécessitent une grande variété de méthodes d'adaptation. Toutes les méthodes envisagées dans la section 3.4 sont utilisées dans NAC, en plus des méthodes plus traditionnelles comme le choix d'une variante existante ou l'exploitation des alternatives offertes par les formats de documents (voir section 3.1).

Ces méthodes restent néanmoins d'un nombre raisonnable grâce à deux approches qui permettent l'utilisation de chaque méthode dans plusieurs contextes différents. La première approche est fondée sur des paramètres qui déterminent le comportement précis de la méthode. La valeur de ces paramètres est prise directement dans le contexte, ou calculée à partir du contexte. C'est de cette façon par exemple que les images peuvent être adaptées à la largeur de l'écran du terminal, la méthode s'appuyant sur un paramètre qui donne la taille de l'écran. La deuxième approche utilise deux phases. Elle engendre d'abord des règles à partir du profil du client, puis elle applique ces règles sur le contenu à adapter (Lemlouma *et al.*, 4-5 mars 2002).

L'architecture NAC distingue deux types d'adaptation selon qu'on prend en compte ou non les variations des différentes caractéristiques du contexte : l'adaptation statique et l'adaptation dynamique.

4.4.1. *Adaptation statique*

L'adaptation statique consiste à préparer une collection de méthodes pour les différents contextes. Chaque méthode considère un unique couple (contexte initial, contexte final) et ne peut pas être réappliquée si le contexte change. Lorsqu'il y a une correspondance entre les caractéristiques du contexte courant et les caractéristiques du couple de contextes de la méthode d'adaptation, la méthode est appliquée et le résultat est utilisé pour transmettre finalement un contenu adapté.

Les méthodes d'adaptation peuvent être spécifiées en utilisant un langage de programmation classique, comme elles peuvent être spécifiées en utilisant des langages

spécifiques, notamment ceux dédiés à la transformation de documents. De façon générale, les méthodes basées sur l'utilisation des langages de programmation généralistes concernent les adaptations appliquées aux objets média, alors que les langages de transformation sont utilisés pour adapter la structure dans laquelle ces objets média sont assemblés pour réaliser une présentation multimédia.

En suivant une approche d'adaptation statique, la variabilité des caractéristiques du contenu ainsi que les changements du contexte nécessitent une grande variété de méthodes d'adaptation pour chaque couple (contexte initial, contexte final). Cela rend nécessaire le développement d'une approche dynamique.

4.4.2. *Adaptation dynamique*

Une méthode d'adaptation est applicable à un document ou à un objet média si la description UPS de cette ressource correspond aux conditions d'entrée de la méthode et si la description UPS de sortie de la méthode correspond aux contraintes du client cible. Plutôt que de spécifier plusieurs méthodes d'adaptation statiques pour chaque type de ressource, il est plus efficace de définir des méthodes d'adaptation qui peuvent chacune engendrer en sortie une variété de ressources de caractéristiques différentes et qui peuvent être appliquées dans des contextes différents. C'est le principe de l'adaptation dynamique.

NAC prend en compte l'adaptation dynamique en suivant deux approches :

- les méthodes d'adaptation paramétrées,
- la génération automatique d'adaptation.

La première approche est assurée par des méthodes d'adaptation qui supportent chacune plusieurs contextes, par exemple une méthode de conversion de ressources de différents formats d'entrée, une compression variable d'images, une extraction de contextes de profils guidée par des expressions XPath, etc.

Pour prendre en compte le contexte, des variables sont associées à chaque méthode d'adaptation dynamique (par exemple, le format d'entrée, le niveau de compression, une expression XPath de sélection, etc.). Lors de la réception d'une requête et après l'évaluation du contexte et de ses différentes caractéristiques, le processus de négociation applique la méthode d'adaptation correspondante en affectant à ses variables les valeurs des caractéristiques qui correspondent à la nature de l'adaptation. Une adaptation dynamique de conversion peut ainsi être exécutée avec la valeur de format de sortie 'GIF' si le client cible ne supporte pas le format d'origine de la ressource image ; une adaptation de compression peut être lancée avec un niveau de compression élevé si le contexte courant indique que la bande passante du réseau est très faible.

Dans NAC, l'adaptation dynamique est aussi utilisée pour l'extraction de contexte en s'appuyant sur la base de profils UPS. En effet, le processus d'extraction de contexte est une méthode unique qui accepte différentes variables en entrée. Ces variables guident la sélection des parties de profils UPS, qui sont ensuite envoyées au proxy.

Le choix entre les méthodes d'adaptation dynamiques disponibles prend en compte

- la qualité du contenu : le contenu adapté doit respecter les caractéristiques et la sémantique du contenu source,
- l'efficacité des méthodes : on cherche des méthodes d'adaptation économes en temps d'exécution,
- la satisfaction du contexte courant : l'instanciation des différents paramètres doit respecter le contexte cible.

La génération automatique d'adaptation, elle, s'appuie sur le langage de transformation XSLT, largement utilisé dans NAC pour assurer la transformation structurale des documents. Pour ces transformations, il est nécessaire de lier les feuilles de transformation aux différents profils des clients cibles. Afin d'éviter le développement de feuilles de transformation pour chaque profil client et pour chaque type de document, une approche consiste à concaténer le document source et le profil du client et à appliquer ensuite une feuille de transformation sur la structure résultante. La concaténation permet aux règles de la feuille de transformation d'effectuer des traitements et des sélections qui dépendent du profil cible. Mais dans la pratique, le développement de telles feuilles de transformation est très compliqué. Cela est dû au fait que les deux parties (contraintes de profil et document source) de la structure résultante sont en fait indépendantes, ce qui nécessite un traitement lourd dans l'utilisation des contraintes pour la transformation du document cible.

Pour résoudre ce problème, NAC adopte une approche différente. On définit une feuille de transformation générique qui s'applique à un document de type profil et qui génère en sortie une nouvelle feuille de transformation. La génération se fait automatiquement sur le profil courant, et la transformation engendrée est appliquée à l'ensemble des documents demandés par la requête. Ce processus permet de prendre en compte les contraintes du client déclarées dans son profil.

5. Mise en œuvre, applications et évaluation

Les différentes techniques d'adaptation présentées dans les sections précédentes ont été mises en œuvre dans l'architecture NAC, expérimentées et évaluées.

5.1. Adaptation des documents et objets média

Les méthodes de transformation d'objets média disponibles dans NAC concernent principalement les images statiques et la vidéo. Il s'agit de méthodes de compression, pour générer des objets plus compacts, de méthodes de changement de taille, pour produire des objets affichables sur différents écrans, et de méthodes de conversion de type (transmodage), pour produire par exemple une image représentant une vidéo. Toutes ces méthodes ont été réalisées en embedded C++ ou en Java.

D'autres méthodes prennent en charge l'adaptation des structures de documents. Puisque, dans le contexte qui nous intéresse, les structures sont principalement représentées en XML, ces méthodes ont été réalisées avec le langage de transformation XSLT. Parmi les méthodes disponibles dans NAC, on peut citer notamment les méthodes de filtrage, pour supprimer sélectivement certains éléments de structure, les conversions de langages, comme le passage de XHTML vers WML pour le support des terminaux WAP, ou encore le passage de certains langages XML vers SVG pour produire une forme affichable à partir d'une représentation plus abstraite. Un cas particulier est la transformation de MathML vers SVG.

Les méthodes d'adaptation dynamiques ont également été réalisées dans l'architecture NAC. Un bon exemple est le filtrage dynamique de documents SMIL. Supposons que le client auquel on veut envoyer un tel document indique dans son profil qu'il ne supporte pas les objets média de type audio ni les alternatives proposées par SMIL (élément `switch`). L'application d'une feuille de transformation XSLT générique sur le profil permet d'effectuer une sélection et une instanciation des règles de transformation nécessaires. Elle engendre une autre feuille de transformation XSLT qui filtre les éléments `audio` et remplace chaque élément `switch` par l'une de ses alternatives.

Les méthodes dynamiques de NAC couvrent également l'adaptation des objets média eux-mêmes. Un cas intéressant est celui du changement automatique du taux de compression des images en fonction de la bande passante disponible pour leur transmission. Les expérimentations ont montré les gains importants que l'on peut obtenir sur les temps de chargement, en adaptant dynamiquement les taux de compression (voir Tableau 1).

	Image	Adapt. 1	Adapt. 2	Adapt. 3
Compression JPEG	0 %	50 %	80 %	90 %
Taille (octets)	13131	10094	6576	5062
Gain de taille (octets)	0	3037	6555	8069
Temps de chargement de l'image par le proxy (ms)	25			
Temps d'adaptation (ms)	0	95	95	95
Temps de réception (ms) bande passante 128 Kbit/s	820,69	725,87	506	411,37
Temps de réception (ms) bande passante 256 Kbit/s	410,34	410,43	300,5	253,19
Temps de réception (ms) bande passante 512 Kbit/s	205,17	252,72	197,75	174,10

Tableau 1. Mesures d'adaptation dynamique

5.2. Adaptation aux terminaux à capacités limitées

L'architecture NAC a été employée en particulier pour adapter des documents multimédias complexes à des terminaux mobiles légers, aux capacités limitées. On a par

exemple abordé le cas d'un terminal qui sait jouer un flux vidéo MPEG-1, mais ne supporte pas le langage SMIL et ne propose aucune possibilité d'adaptation. Pour permettre aux utilisateurs d'un tel terminal d'accéder à des contenus SMIL quelconques, l'architecture d'adaptation doit opérer une transformation radicale.

On tire parti de plusieurs éléments de l'architecture NAC pour réaliser cette adaptation. Le module ANM exploite les profils UPS pour connaître les caractéristiques exactes de l'appareil. Il évalue aussi le contexte instantané pour prendre en compte notamment la bande passante disponible ou la langue de l'utilisateur. Enfin, il effectue une transformation qui concerne à la fois la structure SMIL et les objets média. La connaissance de la langue lui permet, dans les éléments *switch* de SMIL, de ne retenir que les textes en français et d'ignorer les alternatives dans les autres langues. Il analyse la totalité de la structure SMIL pour savoir comment les différents objets média se placent dans le temps et sur la surface de l'écran. Il décode chaque objet média et les combine pour produire un unique objet média, une vidéo dans le format supporté par l'appareil. La compression de cet objet média final est déterminée par la bande passante disponible à cet instant.

Les temps de traitement ne sont pas négligeables dans une adaptation de ce type. Pour plus d'efficacité, la transformation est faite en deux temps. Le premier temps consiste à transformer le document SMIL et ses objets média en un seul fichier vidéo. Avec un document de 32 secondes qui intègre une vidéo, plusieurs sous-titres (texte), des commentaires vocaux et un logo fixe (image) présent pendant toute la durée de la présentation, il faut 36 secondes de calcul pour produire et stocker une vidéo de 352x288 pixels à 25 images/seconde. Cette vidéo est enregistrée dans un fichier puis adaptée au vol lors de l'envoi. C'est alors que les dimensions, l'encodage ou la cadence sont adaptés au contexte courant. Des mesures détaillées de ces adaptations ont été effectuées. Elles sont présentées dans (Hagimont *et al.*, 2002).

5.3. Bases de profils

Une base de profils UPS a été créée. Elle utilise pour la négociation et l'interrogation des profils l'implémentation Apache de SOAP et l'implémentation Tamino du langage d'interrogation XQuery. Elle comporte cent dix-huit descriptions de terminaux mobiles en UPS, qui sont accessibles en ligne (Lemlouma, 2003).

Des mesures ont été menées pour évaluer les bénéfices de l'approche qui consiste à ne pas transmettre des profils entiers, mais seulement les parties qui sont pertinentes pour une adaptation donnée. Les recherches dans la base sont plus rapides lorsque chaque profil est traité comme un tout indivisible, mais les temps de transferts de tels profils deviennent alors plus longs. À l'inverse, des requêtes XQuery ciblées pour obtenir seulement les fragments de profils utiles demandent plus de temps d'exécution, mais les résultats sont transmis plus rapidement. L'expérimentation a été menée sur une plate-forme constituée d'un réseau sans fil IEEE 802.11b et d'un Ethernet filaire à 100 Mbits/s avec des terminaux Pocket PC sous Windows CE, PC portables et PC

de bureau. Les mesures montrent un réel bénéfice à utiliser des requêtes précises renvoyant des fragments de profils.

6. Conclusion

De cette présentation des principales caractéristiques de NAC et de ses applications, il ressort quelques points sur lesquels cette architecture apporte une contribution. Le premier point est sans doute une approche globale du problème de l'adaptation sur le web, qui prend en compte l'ensemble des caractéristiques des environnements hétérogènes et la diversité des clients, avec toutes leurs contraintes. Cette approche englobe les formats de documents et d'objets média, les modèles et langages de description de contexte, les protocoles de négociation et les méthodes de transformation des structures et des contenus.

De plus, sur chacun de ces aspects, NAC tente d'étendre les résultats apportés par les systèmes précédents. Avec UPS, la description du contexte prend en compte toutes les entités participantes et capture toutes les caractéristiques qui peuvent influencer le processus de transmission d'un contenu adapté. L'extensibilité apportée par les schémas RDF assure l'évolutivité du modèle.

Le protocole de négociation est un autre aspect important de NAC. L'absence d'un tel protocole dans les systèmes d'adaptation disponibles sur le web réduit considérablement leurs possibilités, puisqu'il est alors impossible d'avoir une vue précise et actualisée de tous les aspects du contexte.

Les méthodes d'adaptation paramétrées et la génération dynamique de transformations apportent essentiellement la dynamique. Elles permettent une grande évolutivité de l'adaptation ainsi qu'une bonne flexibilité qui permet de réagir instantanément aux changements du contexte.

Enfin, la mise en œuvre de l'architecture NAC dans une application distribuée s'appuyant sur des proxies et des bases de profils partagées permet un déploiement effectif qui englobe les systèmes existants, sans nécessiter de modifications de ces systèmes, simplement en y ajoutant les nouveaux composants que sont les proxies et les bases de profils.

Sans atteindre la généralité de MPEG-21, l'architecture NAC répond aux besoins d'adaptation du web multimédia et est utilisable immédiatement, sans remise en cause de l'existant. Elle ne couvre certes pas certains aspects de MPEG-21, comme notamment la gestion des droits de propriété intellectuelle, mais de ce point de vue, un document SMIL, même adaptable, n'est pas différent d'une page HTML utilisant quelques applets.

Dans cet article, nous avons pu constater que l'adaptation des ressources multimédias du web à l'environnement hétérogène apporté par les appareils et réseaux mobiles est un problème vaste aux facettes multiples. Plusieurs techniques convergent à sa résolution, qui vont des formats de document indépendants des terminaux aux

transformations de structures et de contenu, en passant par les langages de description de contexte et les protocoles de négociation.

Sur chacune de ces techniques des progrès ont été faits récemment et d'autres sont en cours, mais en même temps, il est nécessaire d'aborder les questions d'architecture : le modèle très simple du web des débuts, où un client dialogue directement avec un serveur à l'aide d'un unique protocole de transfert, ne suffit plus. Des architectures réparties plus élaborées sont nécessaires pour permettre de nouveaux progrès dans ce domaine.

7. Bibliographie

- Amaya, *Amaya at W3C*, <http://www.w3.org/Amaya/>, 2005.
- Ayars J., *Synchronized Multimedia Integration Language (SMIL 2.0)*, W3C Recommendation, <http://www.w3.org/TR/smil20/>, 2001.
- Baker M., Ishikawa M., Matsui S., Stark P., Wugofski T., Yamakami T., *XHTML Basic*, W3C Recommendation, <http://www.w3.org/TR/xhtml-basic/>, 2000.
- Boll S., ZYX : Towards Flexible Multimedia Document Models for Reuse and Adaptation, Thèse de doctorat, Université de Vienne, Autriche, 2001.
- Boll S., Klas W., « ZYX - A Semantic Model for Multimedia Documents and Presentations », *Proceedings of the 8th IFIP Conference on Data Semantics (DS-8)*, Rotorua, Nouvelle-Zélande, 1999a.
- Boll S., Klas W., Wandel J., « A Cross-Media Adaptation Strategy for Multimedia Presentations », *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, ACM, Orlando, Floride, USA, p. 37-46, 1999b.
- Bos B., *Cascading Style Sheets home page*, W3C, <http://www.w3.org/Style/CSS/>, n.d.
- Bulterman D., Rossum G. V., van Liere R., « A structure for transportable, dynamic multimedia documents », *Proceedings of the Summer 1991 USENIX Conference*, Nashville, TN, p. 137-155, 1991.
- Burnett I., de Walle R. V., Hill K., Bormans J., « MPEG-21 : Goals and Achievements », *IEEE Multimedia*, vol. 10, n° 4, p. 60-70, 2003.
- Clark J., *XSL Transformations (XSLT) Version 1.0*, W3C Recommendation, <http://www.w3.org/TR/xslt>, 1999.
- Euzenat J., Layaïda N., Dias V., « A semantic framework for multimedia document adaptation », *Proceedings of the 18th International Joint Conference on Artificial Intelligence IJCAI'2003*, Morgan Kaufman, San-Mateo, p. 31-36, 2003.
- Ferraiolo J., Fijisawa J., Jackson D., *Scalable Vector Graphics (SVG) 1.1 Specification*, W3C Recommendation, <http://www.w3.org/TR/SVG/>, 2003.
- Gronbaek K., Randall H., « Design issues for a dexter-based hypermedia system », *Communications of the ACM*, vol. 37, n° 2, p. 41-49, 1994.
- Hagimont D., Layaïda N., « Adaptation d'une application multimédia par un code mobile », *Technique et science informatiques*, vol. 21, n° 6, p. 877-897, 2002.
- Halasz F., Schwartz M., « The Dexter hypertext reference model », *Communications of the ACM*, vol. 37, n° 2, p. 30-39, 1994.

- Hanrahan R., Merrick R., *Authoring Techniques for Device Independence*, W3C Working Note, <http://www.w3.org/TR/di-atdi/>, 2004.
- Hardman L., Bulterman D., Rossum G. V., « The Amsterdam hypermedia model : Adding time and context to the dexter model », *Communications of the ACM*, vol. 37, n° 2, p. 50-63, 1994.
- Hoschka P., *Synchronized Multimedia Integration Language (SMIL 1.0)*, W3C Recommendation, <http://www.w3.org/TR/REC-smil/>, 1998.
- Jan G., Dave B., *RDF Test Cases*, W3C Proposed Recommendation, <http://www.w3.org/TR/rdf-testcases/>, 2003.
- Klyne G., *Composite Capability/Preference Profiles (CC/PP) : Structure and Vocabularies 1.0*, W3C Recommendation, <http://www.w3.org/TR/CCPP-struct-vocab/>, 2004.
- Layaïda N., *Madeus : Système d'édition et de présentation de documents structurés multimédia*, Thèse de doctorat en informatique, Université Joseph Fourier, Grenoble, 1997.
- Lemlouma T., *UPS Client Repository*, INRIA, <http://wam.inrialpes.fr/people/lemlouma/MULTIMEDIA/UPS-Client-Repository/ups-client-repository.html>, 2003.
- Lemlouma T., Layaïda N., « Context-Aware Adaptation for Mobile Devices », *IEEE Int. Conf. on Mobile Data Management*, Berkeley, California, USA, p. 106-111, 19-22 janvier 2004.
- Lemlouma T., Layaïda N., « Adapted Content Delivery for Different Contexts », *Proc. International Symposium on Applications and the Internet (SAINT 2003)*, IEEE Computer Society, p. 190-197, 2003.
- Lemlouma T., Layaïda N., « Universal Profiling for Content Negotiation and Adaptation in Heterogeneous Environments », *W3C Workshop on Delivery Context*, Sophia-Antipolis, France, <http://www.w3.org/2002/02/DIWS/>, 4-5 mars 2002.
- LimSee, *LimSee2*, <http://wam.inrialpes.fr/software/limsee2/>, 2005.
- Lum W., Lau C., « A Context-Aware Decision Engine for Content Adaptation », *Pervasive Computing*, vol. 1, n° 3, p. 41-49, 2002.
- Mitra N., *Simple Object Access Protocol (SOAP) Version 1.2 Part 0 : Primer*, W3C Recommendation, <http://www.w3.org/TR/soap12-part0/>, 2003.
- Mohan R., Smith J., Li C., « Adapting Multimedia Internet Content For Universal Access », *IEEE Transactions on Multimedia*, vol. 1, n° 1, p. 104-114, March 1999.
- Smith J., Mohan R., Li C., « Scalable Multimedia Delivery for Pervasive Computing », *Proceedings of Seventh ACM International Conference on Multimedia (Part 1)*, Orlando, Floride, USA, p. 131-140, 1999.
- Vetro A., « MPEG-21 Digital Item Adaptation : Enabling Universal Multimedia Access », *IEEE Multimedia*, vol. 10, n° 4, p. 84-87, 2004.

Article reçu le 20 avril 2004
Version révisée le 3 février 2005

Nabil Layaïda a soutenu en 1997 une thèse de doctorat à l'Université Joseph Fourier, Grenoble. Il est actuellement chargé de recherches à l'INRIA Rhône-Alpes dans le projet WAM. Nabil est membre du groupe de travail SYMM (Synchronized Multimedia) du World Wide Web

Consortium (W3C) et co-auteur des langages SMIL 1.0, SMIL 2.0 et SMIL 2.1 (Synchronized Multimedia Integration Language). Ses thèmes de recherche portent sur le traitement des documents multimédias structurés, les transformations XML et l'adaption automatique de contenu.

Tayeb Lemlouma a soutenu en 2004 une thèse de doctorat à l'Institut National Polytechnique de Grenoble (INPG). Il est actuellement post-doctorant à l'INRIA Rhône-Alpes. Il est membre du groupe de travail Device Independence du World Wide Web Consortium (W3C) et un des auteurs d'une spécification produite par ce groupe. Ses recherches portent sur l'adaptation et la négociation de contenu pour les appareils hétérogènes (assistants personnels, téléphones mobiles, etc.).

Vincent Quint est directeur de recherche à l'INRIA Rhône-Alpes. Après avoir travaillé sur les réseaux, il s'est tourné vers les documents électroniques puis le web. Il est entré au World Wide Web Consortium (W3C) en 1996 et y a occupé les fonctions de directeur adjoint et de responsable du domaine Formats de documents. Il a lancé en 2003 le projet WAM (Web, Adaptation et Multimédia) à l'INRIA. Depuis février 2005, il est co-animateur du Technical Architecture Group (TAG) du W3C.