Une archietecture orientée services pour la fourniture de documents multimédia composés adaptables

Zakia Aoul Kazi Aoul

École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

TELECOM PARIS
école nationale supérieure des télécommunications

Thèse

Présentée en vue de l'obtention du grade de docteur de l'École Nationale Supérieure des Télécommunications
Au département Informatiques et Réseaux

Une architecture orientée services pour la fourniture de documents multimédia composés adaptables

Par

Zakia Imane Kazi-Aoul

Soutenue le 18 Janvier 2008 devant le jury composé de :

Françoise André	**Président du jury**
Hervé Guyennet	**Rapporteur**
Jean-Marc Pierson	**Rapporteur**
Vania Conan	**Membre du jury**
Isabelle Demeure	**Directrice de thèse**
Jean-Claude Moissinac	**Co-directeur de thèse**

REMERCIEMENTS

Je voudrais tout d'abord remercier ma directrice de thèse, Isabelle Demeure, pour ses conseils avisés, son soutien et sa patience. Je remercie également mon co-directeur de thèse, Jean-Claude Moissinac, sans qui ce travail n'aurait pas été possible.

Je tiens à remercier Hervé Guyennet et Jean-Marc Pierson, d'avoir consacré du temps pour la relecture et l'évaluation de mon rapport, Françoise André d'avoir accepté de présider le jury et Vania Conan d'avoir accepté d'en être membre.

Je remercie également Oscar et Ahmad d'avoir été des collègues exemplaires, ainsi que Bruno et Sonia d'avoir relu ma thèse.

Finalement, je remercie toutes les personnes qui m'ont soutenue, supportée, poussée à aller de l'avant. Je ne pourrai malheureusement pas citer tout le monde car je n'ai pas assez de place mais les personnes que je porte dans mon cœur se reconnaîtront. Merci à vous tous d'avoir été là pour moi.

À ma petite famille que j'aime …

TABLE DES MATIÈRES

TABLE DES MATIÈRES **1**

TABLE DES FIGURES **5**

TABLE DES TABLEAUX **7**

RÉSUMÉ **9**

ABSTRACT **12**

CHAPITRE 1 : INTRODUCTION **15**

1 MOTIVATIONS ET OBJECTIFS 15
2 PLAN DU RAPPORT 18

CHAPITRE 2 : ADAPTATION DE CONTENUS MULTIMÉDIA : ÉTAT DE L'ART **21**

1 INTRODUCTION 21
2 ADAPTATION : POURQUOI ET QUOI ? 21
2.1 Définition de l'adaptation *21*
2.2 Pourquoi adapter ? *21*
2.3 Qu'allons nous adapter ? *22*
3 CARACTÉRISTIQUES DES ARCHITECTURES DE FOURNITURE ET D'ADAPTATION DE CONTENUS MULTIMÉDIA 23
3.1 Gestion du contexte *24*
3.2 Gestion des contenus multimédia *25*
3.3 Gestion de la prise de décision *26*
3.4 Gestion de l'adaptation de contenus multimédia *27*
3.5 Adaptation des contenus multimédia *27*
3.5.1 Modèle architectural client / serveur et adaptation 27
3.5.1.1 Système d'adaptation au niveau du client 28
3.5.1.2 Système d'adaptation au niveau du serveur 29
3.5.2 Modèle architectural client / intermédiaire (s) / serveur et adaptation 29
3.5.3 Modèle architectural Pair-à-Pair et adaptation 31
3.5.4 Discussion : adaptation orientée client, serveur, intermédiaire(s) ou P2P ? 31
3.6 Techniques de consommations et d'adaptations de contenus multimédia *32*
3.6.1 Consommation d'un contenu multimédia 33
3.6.2 Catégorisation des techniques d'adaptation 33
4 ÉTAT DE L'ART SUR LES ARCHITECTURES DE FOURNITURE DE CONTENU MULTIMÉDIA 35
4.1 ADMITS *35*
4.1.1 Gestion du contexte dans ADMITS 37
4.1.2 Gestion de la prise de décision dans ADMITS 37
4.1.3 Gestion de l'adaptation du service dans ADMITS 37
4.2 APPAT *39*
4.2.1 Gestion du contexte dans APPAT 40
4.2.2 Gestion de la prise de décision dans APPAT 41
4.2.3 Gestion de l'adaptation dans APPAT 41
4.3 ISIS *42*
4.3.1 Gestion du contexte dans ISIS 43
4.3.2 Gestion de la prise de décision dans ISIS 43
4.3.3 Gestion de l'adaptation du service dans ISIS 44
4.4 UMA ou l'accès universel au multimédia *45*
4.4.1 Gestion du contexte et des contenus dans UMA 46
4.4.2 Gestion de la prise de décision dans UMA 46
4.4.3 Gestion de l'adaptation du service dans UMA 47
4.5 WAM et son architecture NAC *48*
4.5.1 Gestion du contexte dans NAC 49
4.5.2 Gestion de la prise de décision et d'adaptation du service dans NAC 49
4.6 DCAF *50*
4.6.1 Gestion du contexte dans DCAF 51
4.6.2 Gestion des contenus dans DCAF 51
4.6.3 Gestion de la prise de décision dans DCAF 51

4.6.4 Gestion de l'adaptation dans DCAF....52
4.7 Architecture SATO du projet Ambient....52
4.8 MAPS....54
4.9 M21....54
5 DISCUSSION ET ILLUSTRATIONS DU SCÉNARIO....54
5.1 Rappel du scénario « Suzy and her PDA »....54
5.2 Illustration du scénario....55
5.3 Discussion....55
5.3.1 Modèle architectural....56
5.3.2 Passage à l'échelle et l'extensibilité....56
5.3.3 Distribution et la description des ressources d'adaptation....56
5.3.4 Description des caractéristiques contextuelles spécifiques....56
6 OUTILS DE DESCRIPTION DE CONTEXTE....56
6.1 CC/PP....57
6.2 MPEG-21....58
7 OUTILS DE DESCRIPTION DE CONTENUS MULTIMÉDIA....62
7.1 MPEG-7....62
7.2 Dublin Core....64
7.3 SMIL....66
8 CONCLUSION : CONSTATATIONS ET OBJECTIFS....67

CHAPITRE 3 : PAAM : UNE ARCHITECTURE POUR LA FOURNITURE DE CONTENUS MULTIMÉDIA ADAPTABLES....69

1 INTRODUCTION....69
1.1 PAAM et la dimension économique....70
1.2 PAAM et la dimension technologique....70
2 ANALYSE DU SCÉNARIO « SUZY AND HER PDA »....71
2.1 Besoins fonctionnels pour réaliser le scénario....72
3 ARCHITECTURE FONCTIONNELLE DE PAAM....73
3.1 Déroulement d'une requête typique dans PAAM....75
3.2 Gestionnaire de contexte....77
3.2.1 Descripteur du contexte utilisateur....78
3.3 Gestionnaire du document multimédia composé....79
3.3.1 Descripteur du document composé :....80
3.4 Planificateur....81
3.4.1 Algorithme générique du processus de prise de décision et d'adaptation....82
3.5 Gestionnaire d'adaptation....83
3.5.1 Descripteur de l'adaptateur :....85
3.6 Couche de services....86
3.7 Plate-forme d'exécution....86
4 PASSAGE À L'ÉCHELLE DE PAAM....86
4.1 Entité de supervision....88
4.2 Interfaces d'accès à PAAM....88
4.2.1 Ajout d'un système d'adaptation PAAM....89
4.2.2 Ajout d'un adaptateur....89
4.2.3 Ajout d'un utilisateur....89
4.3 Discussion sur le passage à l'échelle....89
5 CONCLUSION....90

CHAPITRE 4 : PAAM ET L'UTILISATION DES SERVICES WEB....91

1 INTRODUCTION....91
2 OUTILS DES SERVICES WEB POUR ANNONCER, DÉCOUVRIR ET EXÉCUTER....92
2.1 Services Web et SOA....92
2.2 Définition des services Web....93
2.3 SOAP....94
2.4 WSDL....94
2.5 UDDI....96
3 COMPOSITION ET ORCHESTRATION DES SERVICES WEB AVEC BPEL/WS-BPEL....97
3.1 Utilisation de BPEL dans PAAM....98
3.2 Exemple générique d'un processus BPEL....99
3.3 Concepts de base de BPEL....100
3.3.1 Liens partenaires....100
3.3.2 Variables....101
3.3.3 Processus BPEL....101
3.3.4 Exemple....101
3.4 Serveurs BPEL....107

4 EXTENSIBILITÉ, MODULARITÉ ET PASSAGE À L'ÉCHELLE (SCALABILITÉ) AVEC LES SERVICES WEB 107
5 CONCLUSION 108

CHAPITRE 5 : GESTION DES ADAPTATEURS DISTRIBUÉS ET DE LA DYNAMICITÉ 109

1 INTRODUCTION 109
2 CATÉGORISATION ET IDENTIFICATION DES ADAPTATEURS 109
2.1 Catégories d'adaptateurs 110
2.2 Identification uniforme des noms des adaptateurs 110
2.3 Description sémantique d'un service Web d'adaptation 112
2.4 La liste temporaire des descriptions des adaptateurs 114
3 CHOIX D'UN ADAPTATEUR 115
3.1 Définition 115
3.2 Protocole de négociation et d'acceptation de PAAM 116
3.3 Politique d'acceptation d'un adaptateur 119
3.4 Exemples de politiques d'acceptation 120
3.4.1 Exemple 1 120
3.4.2 Exemple 2 : WESEMAC 120
4 DYNAMICITÉ ET GESTION DES DÉCONNEXIONS 121
4.1 Changements dans le contexte de l'utilisateur 121
4.2 Changements dans le contexte des adaptateurs 122
4.3 Tolérance aux disparitions des adaptateurs 122
4.3.1 Détection des disparitions des adaptateurs 124
4.3.2 Premier cas de la gestion des disparitions des adaptateurs 124
4.3.3 Deuxième cas de la gestion des disparitions des adaptateurs 125
4.3.4 Remarque 125
4.4 Reconstruction du graphe d'adaptation 125
5 CONCLUSION 126

CHAPITRE 6 : IMPLÉMENTATION D'UNE CHAÎNE D'ADAPTATION COMPLÈTE 128

1 INTRODUCTION 128
2 APERÇU DE LA CHAÎNE D'ADAPTATION COMPLÈTE DE PAAM 129
3 CHOIX TECHNOLOGIQUES 130
3.1 Projets J2EE, serveurs d'applications et pile de services Web 131
3.1.1 Apache 131
3.1.2 GlassFish 131
3.2 Choix d'implémentation 132
4 ÉLÉMENTS DE LA CHAÎNE D'ADAPTATION 133
4.1 Description du contexte utilisateur 133
4.1.1 Préférences de l'utilisateur 135
4.1.2 Capacités du terminal 135
4.2 Description du document multimédia composé 136
4.3 Implémentation de la prise de décision 137
4.3.1 Définition du graphe d'adaptation 137
4.3.2 Politiques d'adaptation 138
4.3.2.1 Politiques d'adaptation relatives au handicap 138
4.3.2.2 Politiques d'adaptations relatives aux médias 138
4.3.2.1 Remarque 139
4.3.3 Algorithme de prise de décision 139
4.4 Discussion sur l'algorithme de prise de décision 139
5 CONCLUSION 143

CHAPITRE 7 : ÉTUDE DES COÛTS ET TESTS 145

1 INTRODUCTION 145
2 PAAM VS ARCHITECTURE D'ADAPTATION CENTRALISÉE DE CONTENUS MULTIMÉDIA 145
2.1 Analyse et comparaison 146
2.2 Conclusion 147
3 ÉTUDES DES COÛTS DES FONCTIONNALITÉS ADDITIONNELLES DE PAAM 147
3.1 Conditions d'expérimentations et hypothèses 147
3.2 La constitution de la liste temporaire des adaptateurs disponibles 148
3.3 Implémentation du protocole de négociation et d'acceptation 150
3.4 Gestion des déconnexions des adaptateurs 150
3.5 Gestion des documents multimédia composés 151
3.6 Apport de la parallélisation des adaptations 152
4 CONCLUSION 156

CHAPITRE 8 : CONCLUSION ET PERSPECTIVES 157

1 RAPPEL DES OBJECTIFS 157
1.1 Adapter…oui mais où ? *157*
1.2 L'adaptation distribuée *157*
2 CONTRIBUTIONS ET BILAN SCIENTIFIQUE 158
3 PERSPECTIVES ET TRAVAUX FUTURS 159
3.1 Travaux futurs relatifs au planificateur *159*
3.2 Travaux futurs relatifs au gestionnaire d'adaptation *159*
3.3 Travaux futurs relatifs à un adaptateur *159*
3.4 Autres travaux envisagés *160*

LISTE DES PUBLICATIONS 161

BIBLIOGRAPHIE 162

ANNEXE 1 : WESEMAC : WEB SERVICES MULTIMÉDIA 168

1 OBJECTIF DU PROJET WESEMAC 168
2 FRAMEWORK D'EXÉCUTION 168
2.1 Environnement de développement et d'exécution *168*
2.2 Logiciels *168*
2.2.1 Tomcat 5.0.28 168
2.2.2 AXIS 2 (axis 2 0.92) 168
2.3 Librairies de conversion d'entités média *169*
2.3.1 JMF API (version 2.2.1) 169
2.3.1.1 Architecture RTP 170
2.3.2 JAI API (version 1.1.2_01) 170
2.3.3 Architecture fonctionnelle de Wesemac 170

ANNEXE 2 : CONTRAINTES LIÉES AUX ADAPTATEURS 172

1 RAPPEL DU CONTEXTE D'UTILISATION DES ADAPTATEURS 172
2 CONTRAINTES ET SPÉCIFICATIONS DES ADAPTATEURS PAAM 172
2.1 Entrées / sorties des adaptateurs *172*
2.2 Asynchronisme et gestion de suivi de session des adaptateurs *173*
2.3 Algorithme de négociation et d'acceptation *175*
2.3.1 Algorithme 1 175
2.3.2 Algorithme 2 176

TABLE DES FIGURES

FIGURE 1: SCÉNARIO ILLUSTRATIF 16
FIGURE 2 : FONCTIONNALITÉS DE BASE D'UNE ARCHITECTURE DE FOURNITURE DE CONTENUS MULTIMÉDIA ADAPTABLES AU CONTEXTE D'UN UTILISATEUR 23
FIGURE 3 : MODÈLE ARCHITECTURAL CLIENT / SERVEUR 28
FIGURE 4 : MODÈLE ARCHITECTURAL CLIENT / INTERMÉDIAIRE(S) / SERVEUR 30
FIGURE 5 : MODÈLE ARCHITECTURAL PAIR-À-PAIR (P2P) 31
FIGURE 6: L'ARCHITECTURE DU PROJET ADMITS 36
FIGURE 7: LES COMPOSANTS D'UN SERVEUR ADMS 36
FIGURE 8: ARCHITECTURE DE LA PLATE-FORME APPAT 39
FIGURE 9: L'ARCHITECTURE DE BOUT EN BOUT DE ISIS 43
FIGURE 10: ARCHITECTURE DE LA PLATE-FORME UMA 45
FIGURE 11: ORGANISATION GÉNÉRALE DE L'ARCHITECTURE NAC 48
FIGURE 12 : ARCHITECTURE DCAF 51
FIGURE 13 : CC/PP EST UNE APPLICATION RDF QUI REPOSE SUR XML 57
FIGURE 14 : ORGANISATION HIÉRARCHIQUE D'UN PROFIL CC/PP 58
FIGURE 15: EXEMPLE D'UN MODÈLE DE DÉCLARATION D'UN ÉLÉMENT NUMÉRIQUE ([BVH03]) 59
FIGURE 16: ADAPTATION D'UN ÉLÉMENT NUMÉRIQUE (DIA) 60
FIGURE 17: APERÇU ET ORGANISATION D'OUTILS D'ADAPTATION D'ÉLÉMENT DIGITAL 60
FIGURE 18 : EXEMPLE DE DESCRIPTION DES CARACTÉRISTIQUES D'UN TERMINAL ET DES PRÉFÉRENCES D'UN UTILISATEUR EN MPEG-21 62
FIGURE 19: EXEMPLE D'UNE DESCRIPTION MPEG-7 64
FIGURE 20 : EXEMPLE DU DOCUMENT SMIL SHAKIRA.SMIL 66
FIGURE 21 : REPRÉSENTATION AVEC REALONE DU DOCUMENT SMIL SHAKIRA.SMIL 67
FIGURE 22 : DOCUMENT SOURCE DU SCÉNARIO *SUZY AND HER PDA* 71
FIGURE 23 : LE DOCUMENT ADAPTÉ AU CONTEXTE DE SUZY 72
FIGURE 24 : ADAPTATIONS NÉCESSAIRES POUR LE SCÉNARIO *« SUZY AND HER PDA »* 72
FIGURE 25 : VUE GLOBALE DE LA LOGIQUE PAAM 73
FIGURE 26 : L'ARCHITECTURE FONCTIONNELLE DE PAAM 74
FIGURE 27 : DIAGRAMME DE SÉQUENCE D'UN SCÉNARIO TYPIQUE 76
FIGURE 28 : ÉLÉMENTS DU GESTIONNAIRE DE CONTEXTE 77
FIGURE 29 : ÉLÉMENTS DU GESTIONNAIRE DU DOCUMENT MULTIMÉDIA COMPOSÉ 79
FIGURE 30 : LES FONCTIONS DU PLANIFICATEUR 81
FIGURE 31 : LES ÉLÉMENTS DU GESTIONNAIRE D'ADAPTATION 84
FIGURE 32 : LA PASSAGE À L'ÉCHELLE DE PAAM 87
FIGURE 33 : LES DIFFÉRENTES INTERFACES D'ACCÈS 88
FIGURE 34 : L'ARCHITECTURE ORIENTÉE SERVICES DE BASE 93
FIGURE 35 : LE CONCEPT DE BASE DES SERVICES WEB 93
FIGURE 36 : EXEMPLE DE DESCRIPTION WSDL 96
FIGURE 37 : UDDI ET LE PROTOCOLE D'ANNONCE ET DE DÉCOUVERTE DE DESCRIPTIONS WSDL EN UTILISANT DES MESSAGES SOAP 97
FIGURE 38 : INTÉGRATION DU MOTEUR D'EXÉCUTION DES PROCESSUS BPEL AU SEIN DU GESTIONNAIRE D'ADAPTATION DE PAAM 98
FIGURE 39 : EXEMPLE GÉNÉRIQUE D'UN PROCESSUS BPEL 100
FIGURE 40 : INTERACTIONS ENTRE LE PROCESSUS BPEL DE COMPOSITION D'ADAPTATEURS ET SES LIENS PARTENAIRES 102
FIGURE 41: DESCRIPTION PAAM DU SERVICE D'ADAPTATION AUDIOEXTRACTOR 114
FIGURE 42 : PROTOCOLE DE NÉGOCIATION ET D'ACCEPTATION 116
FIGURE 43 : CAS 1 : CHERCHER UN AUTRE ADAPTATEUR POUR REMPLACER L'ADAPTATEUR DISPARU 125
FIGURE 44 : CAS 2 : DEUX ADAPTATEURS POUR LE MÊME MÉDIA SONT INSTANCIÉS EN MÊME TEMPS 125
FIGURE 45 : LES ÉLÉMENTS DE BASE À IMPLÉMENTER POUR RÉALISER UNE PLATE-FORME D'ADAPTATION BASÉE SUR PAAM 128
FIGURE 46 : UNE CHAÎNE D'ADAPTATION COMPLÈTE 129
FIGURE 47 : DESCRIPTION DU CONTEXTE D'UN UTILISATEUR DU SYSTÈME D'ADAPTATION PAAM 134
FIGURE 48 : BALISE METADATA D'UN DOCUMENT SMIL CORRESPONDANT AUX SPÉCIFICATIONS DE PAAM 137
FIGURE 49 : UN GRAPHE D'ADAPTATION DANS LE CADRE DE PAAM 138
FIGURE 50 : ALGORITHME DE PRISE DE DÉCISION 143

FIGURE 51 : PAAM VS ARCHITECTURE D'ADAPTATION CENTRALISÉE 146
FIGURE 52 : VARIATION DU TEMPS DE PARSING DES DESCRIPTIONS PAAM DES ADAPTATEURS SUIVANT LEUR NOMBRE 149
FIGURE 53 : ILLUSTRATION DES DURÉES DES DIFFÉRENTES ÉTAPES DU PROCESSUS D'ADAPTATION DANS PAAM ET DANS L'ARCHITECTURE DE RÉFÉRENCE 155
FIGURE 54 : ARCHITECTURE FONCTIONNELLE DE WESEMAC 171

TABLE DES TABLEAUX

TABLEAU 1 : CLASSIFICATION DE QUELQUES TECHNIQUES D'ADAPTATION 34
TABLEAU 2 : FICHE DESCRIPTIVE DE ADMITS 38
TABLEAU 3 : FICHE DESCRIPTIVE DE APPAT 42
TABLEAU 4 : FICHE DESCRIPTIVE DE ISIS 44
TABLEAU 5 : FICHE DESCRIPTIVE DE UMA 47
TABLEAU 6 : FICHE DESCRIPTIVE DE NAC 50
TABLEAU 7 : FICHE DESCRIPTIVE DE L'ARCHITECTURE DCAF 52
TABLEAU 8 : ÉLÉMENTS DE MÉTA-DONNÉES DU DUBLIN CORE, VERSION 1.1 [DUBLIN] 65
TABLEAU 9 : TABLEAU REPRÉSENTANT LES IDENTIFICATIONS URN DES ADAPTATEURS 112
TABLEAU 10 : TEMPS D'ENVOI DE REQUÊTES SOAP 148
TABLEAU 11:DURÉE DE TEMPS DE TRAITEMENT D'UN DOCUMENT MULTIMÉDIA COMPOSÉ 152
TABLEAU 12 : RÉCAPITULATIF DU NOMBRE DE MESSAGES ÉCHANGÉS LORS DE LA GESTION DES ADAPTATEURS ... 153
TABLEAU 13 : CARACTÉRISTIQUES DU DOCUMENT MULTIMÉDIA COMPOSÉ SERVANT À ILLUSTRER L'APPORT DE LA PARALLÉLISATION 153
TABLEAU 14 : LE NOMBRE DE MESSAGES ÉCHANGÉS LORS DU PROTOCOLE DE NÉGOCIATION ET DE LA GESTION DES ÉTATS DES ADAPTATEURS 154

RÉSUMÉ

L'échange de documents multimédia composés de plusieurs médias élémentaires tels que des vidéos, des images ou du texte, est l'une des applications les plus populaires d'Internet. Idéalement, tout usager d'Internet devrait pouvoir accéder à ces contenus et les recevoir dans un format adapté au contexte dans lequel il travaille.

Un contexte utilisateur peut être défini par les caractéristiques personnelles de l'utilisateur (ex : sa langue parlée, son handicap et ses centres d'intérêt), ses préférences de présentation des contenus multimédia (ex : son lecteur multimédia préféré ou la taille d'image souhaitée), les capacités de son terminal (ex : la taille de l'écran du terminal ou les lecteurs multimédia présents) et les caractéristiques de son réseau d'accès (ex : la bande passante). Compte tenu de la combinatoire des éléments de contexte, il n'est pas envisageable de fournir autant de versions des documents multimédia que de contextes possibles : l'adaptation des contenus est donc nécessaire.

L'accroissement des utilisateurs des terminaux à capacités réduites tels que les assistants personnels (par ex. PDA) exclut une adaptation côté client (ou utilisateur final). L'adaptation, côté source du document multimédia, nécessite l'implémentation de modules supplémentaires qui n'est pas toujours possible et qui peut créer une charge supplémentaire indésirable. L'adaptation par un ou plusieurs intermédiaires répond le mieux aux besoins de passage à l'échelle et d'extensibilité. Une machine intermédiaire est un nœud inséré entre le client et le serveur et dédié, par exemple, à la découverte ou à l'adaptation de services (ex : réduction de la taille d'une image ou traduction et insertion de sous-titres au sein d'une vidéo). L'intermédiation ainsi réalisée apporte une valeur ajoutée en évitant de charger l'utilisateur final et la source du document de tâches spécifiques consommatrices de ressources sans rapport direct avec le service final offert.

Cette approche est celle qui est prise dans la plupart des solutions existantes. Celles-ci utilisent des intermédiaires dédiés. Il en résulte une configuration d'adaptation figée ne garantissant pas la gestion de nouvelles techniques d'adaptation (ex : les adaptations relatives à l'handicap) et ne passant pas à l'échelle. Certaines solutions, basées sur ce même modèle, intègrent l'adaptation distribuée en répartissant la charge entre les intermédiaires qui réalisent l'adaptation. Elles ne traitent cependant pas la gestion dynamique des adaptateurs qui consiste à aller chercher des adaptateurs dans le réseau, les composer et les recomposer dynamiquement en cas de disparition. Elles ne traitent pas non plus l'adaptation des documents multimédia composés qui demande un effort supplémentaire d'analyse du document et de synchronisation des médias élémentaires le composant.

La première contribution de cette thèse est la conception d'une architecture appelée PAAM (pour *Architecture for the **P**rovision of **A**d**A**ptable **M**ultimedia composed documents*) qui a pour but d'adapter des documents multimédia composés au contexte des usagers. L'une des originalités de cette architecture est de mettre en place une adaptation distribuée sur différents nœuds du réseau en évitant de confier l'adaptation à un serveur ou à un intermédiaire dédié. La plate-forme d'adaptation de PAAM intègre aussi bien des fournisseurs de services d'adaptation que des particuliers qui se porteraient volontaires pour exécuter des fonctions d'adaptation en donnant un peu de leurs ressources matérielles et logicielles. Les principaux éléments fonctionnels de PAAM sont : le gestionnaire du contexte utilisateur, le gestionnaire des documents multimédia composés, le planificateur et le gestionnaire d'adaptation. Le gestionnaire du contexte utilisateur et le gestionnaire des documents multimédia composés récupèrent, analysent et agrégent respectivement les informations contextuelles de l'utilisateur et les informations descriptives des documents multimédia. Le planificateur implémente un algorithme de prise de décision reposant sur des politiques d'adaptation. Ce planificateur produit un graphe d'adaptation, c'est-à-dire un ensemble d'adaptateurs organisés en parallèle ou en séquence. Ce graphe est utilisé en entrée du gestionnaire d'adaptation qui recherche ces adaptateurs là où ils se trouvent, les instancie, les compose, si nécessaire, et les recompose si un ou plusieurs adaptateurs disparaissent.

Nous avons choisi d'utiliser les services Web pour implémenter PAAM afin qu'elle soit distribué, extensible, modulable, tolérante aux fautes et passant à l'échelle, répondant ainsi aux limitations des autres architectures d'adaptation. Cette solution technologique permet à PAAM de décrire des ressources d'adaptation, de les publier, de les rechercher et les instancier. Dans le cadre de la composition et de l'orchestration des services Web, nous présentons BPEL (Business Process Execution Language) et son éventuelle intégration au sein d'un gestionnaire d'adaptation pour gérer l'exécution d'un graphe d'adaptation.

La seconde contribution de cette thèse est la gestion des adaptateurs (description, recherche et instanciation). Nous proposons, pour cela, une nomenclature incluant un grand nombre d'adaptateurs. Nous proposons aussi une description d'adaptateurs qui étend WSDL, et qui facilite la recherche, l'instanciation et la composition de ces ressources d'adaptation. Nous exposons par la suite le protocole de négociation et d'acceptation établi entre un gestionnaire d'adaptation et un adaptateur permettant de déterminer si cet adaptateur peut réaliser l'adaptation ou non.

PAAM gérant l'adaptation distribuée sur différents nœuds du réseau, susceptibles de se déconnecter à chaque instant, nous proposons des solutions pour gérer les déconnexions dans PAAM afin de lui procurer un aspect dynamique.

Afin de démontrer la faisabilité de notre architecture, nous implémentons une chaîne d'adaptation complète incluant les principales fonctionnalités de PAAM : le gestionnaire du contexte utilisateur, le gestionnaire des documents multimédia composés, le planificateur et le gestionnaire d'adaptation.

Nous présentons, par la suite, une étude des coûts induits par notre implémentation de PAAM et des tests de performances qui montrent que l'utilisation des services Web n'introduit pas de surcoûts significatifs par rapport au gain obtenu en distribuant l'adaptation sur différents nœuds.

Pour conclure, parce qu'elle permet de gérer une grande variété d'adaptateurs de manière distribuée, l'architecture PAAM répond bien aux limitations des architectures d'adaptation basées sur une configuration client/serveur. L'intérêt de cette approche est la possibilité d'étendre et d'enrichir le système d'adaptation et de le déployer à large échelle tout en garantissant sa robustesse.

ABSTRACT

The exchange of multimedia documents composed of many elementary medias such as videos, images or text is one of the most popular application over the Internet. Ideally, every user should be able to access these contents and consume them in a suitable format adapted to his working context.

A user context can be defined by the personal preferences of the user (e.g. his spoken language, his handicap and his centers of interest), his multimedia contents display preferences (e.g. his preferred multimedia player or the preferred image size), his terminal capabilities (e.g. his terminal screen size or resolution or the present multimedia players) and his access network characteristics (e.g. the related bandwidth). Taking into account the multiplicity of the context elements, it is not conceivable to provide as many multimedia contents versions as possible contexts; thus the adaptation of these contents is necessary.

The increase of users provided with terminals with reduced capabilities such as personal assistants (e.g. PDA) excludes a client (end-user) side adaptation. A server or the multimedia document source side adaptation requires the implementation of additional modules which is not always possible and could create undesirable additional charge. The adaptation within one or more intermediaries meets the needs of scalability and extensibility. An intermediary machine is a node inserted between a client and a server and is for example dedicated to services look up or adaptation (e.g. the reduction of an image size, or the translation and the insertion within a video of subtitles). This kind of intermediation brings then a real value-added by avoiding the charge of the end-user and the document provider of specific tasks with no direct relationship of the final offered service.

This approach is the one followed in the majority of the existing solutions. These solutions are based on dedicated intermediaries; the result is a fixed adaptation configuration that does not support new adaptation techniques (e.g. handicap based adaptations) and scalability. Other intermediaries-based solutions manage the distributed adaptation by dividing the charge between the adaptation intermediaries. However, these solutions do not carry out the dynamic management of the adaptors that consist of searching the adaptors in the network, compose and dynamically recompose them in case of disappearance. They also do not treat the adaptation of multimedia composed document that brings an additional effort in analysing the document and synchronizing the elementary medias composing this document.

The first contribution of this thesis is the design of an architecture called PAAM (for Architecture for the Provision of Multimedia AdAptable composed documents). The purpose of PAAM is to adapt to the users context of the users composed multimedia documents. One of the originalities of this architecture

is to set up a distributed adaptation through various nodes of the network avoiding thus to dedicate the adaptation to a server or an intermediary. PAAM takes advantages from adaptation service providers as well as from private individuals who would voluntary carry out adaptation functions by sharing some of their hardware and software resources. The main functional elements of PAAM are: the user context manager, the multimedia composed document manager, the planner and the adaptation manager. The user context manager and the composed multimedia manager recover and analyze respectively the user contextual information and descriptive information of the multimedia document. The planner implements a decision making algorithm resting on adaptation policies and heuristics. This planner produces an adaptation which is a set of adaptation processes, in parallel or in sequence. This graph is used as an input of the adaptation manager which seeks these adaptors where they are, instantiates them, composes them, if necessary, and recomposes them if one or more adaptors disappear.

We decide to use Web services to implement PAAM allowing it to be distributed, extensible, flexible, fault tolerant, and scalable, thus answering the limitations of other adaptation architectures. This technological choice allows PAAM to describe, announce, search and instantiate adaptation resources. Concerning the composition and the orchestration of adaptation Web services, we present BPEL (Business Process Execution Language) and his possible integration within the adaptation manager in order to supervise the adaptation graph execution.

The second contribution of this thesis is the adaptors management (description, research and instantiation). We propose, for that, a nomenclature including a great number of adaptors. We propose also an adaptor description which extends a WSDL service description, and which facilitates the research, the instantiation and the composition of these adaptation resources. We expose a negotiation and acceptance protocol established between the adaptation manager and a given adapter in order to determine if this adapter can deal with the adaptation.

In order to prove the feasibility of our architecture, we implement a complete adaptation chain including the principal functionalities of PAAM: the user context manager, the multimedia documents manager, the planner and the adaptation manager.

We present, thereafter, a study of the costs that are induced by our PAAM implementation. We also present some tests which show that the use of the Web services does not introduce significant overhead compared to the gain obtained by distributing the adaptation on various nodes and by managing the load balancing between the adaptors.

To conclude, because it makes it possible to manage a large variety of adaptors in a distributed way, The PAAM architecture answers the limitations client/server base adaptation architectures. The interest of this approach is the possibility of extending and of enriching the system by adaptation and of deploying it in a scalable way while guaranteeing its robustness.

Chapitre 1

Introduction

L'échange de documents multimédia composés de plusieurs médias élémentaires tels que des vidéos, des images ou du texte, est l'une des applications les plus populaires d'Internet. Idéalement, tout usager d'Internet devrait pouvoir accéder à ces contenus et les recevoir dans un format adapté au contexte dans lequel il travaille.

Par contexte utilisateur on entend par les caractéristiques personnelles de l'utilisateur (ex : sa langue, son handicap et ses centres d'intérêt), ses préférences de présentation des contenus multimédia (ex : son lecteur multimédia préféré ou la taille d'image souhaitée), les capacités de son terminal (ex : la taille de l'écran du terminal ou les lecteurs multimédia présents) et les caractéristiques de son réseau d'accès (ex : la bande passante). Compte tenu de la combinatoire des éléments de contexte, il n'est pas envisageable de fournir autant de versions de documents multimédia que de contextes possibles : l'adaptation des contenus est donc nécessaire.

1 Motivations et objectifs

Afin de mieux présenter nos motivations et de cerner nos objectifs, considérons une communauté d'utilisateurs géographiquement distants les uns des autres (voir Figure 1). Chaque utilisateur possède ses propres préférences, se connecte grâce à un terminal ayant certaines capacités et utilise son propre réseau d'accès. Ces usagers souhaitent s'échanger des documents multimédia qui peuvent contenir de la vidéo, de l'audio, de l'image et du texte ou un mélange de ces médias élémentaires.

Suzy est l'une de ces internautes. Elle est anglaise. Elle se connecte souvent aux forums de discussion, spécialement ceux qui parlent de musique. Compte tenu de son travail, Suzy est souvent en déplacement et se connecte depuis un assistant personnel (PDA). L'abonnement à son fournisseur d'accès Internet/Téléphonie lui permet de se connecter à tout moment, via différents réseaux, suivant l'endroit où elle se trouve : Wifi-ADSL depuis chez elle, UMTS dans la rue, le train ou la voiture. Suzy souhaite utiliser son PDA avec une configuration minimale afin d'économiser la batterie de son PDA. Ainsi Suzy ne souhaite récupérer que les documents contenant des images, des bandes sons ou du texte.

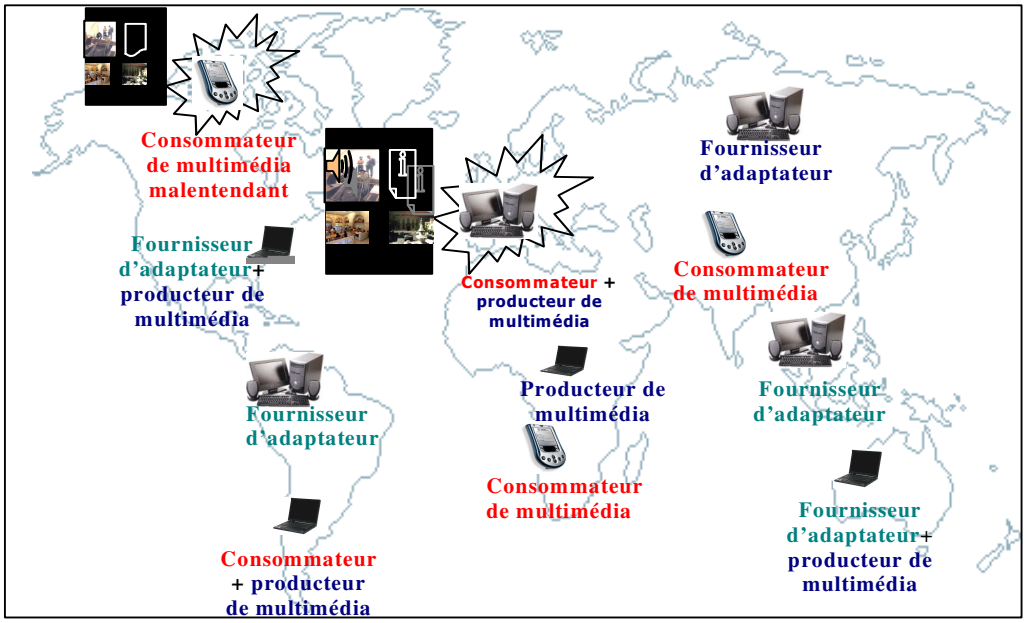

Figure 1: Scénario illustratif

En quittant son bureau, Suzy veut récupérer un document multimédia relatif à sa chanteuse préférée. Il est composé d'un contenu audio-vidéo du dernier clip de « Shakira », d'un texte sur sa bibliographie, écrit en français, et de l'image de la pochette de son dernier album. Ce document multimédia doit être adapté aux préférences de Suzy (langue parlée et restriction sur les contenus vidéo) et aux capacités de son PDA (taille réduite de l'écran).

Certaines fonctions d'adaptation telles que la traduction de texte du français vers l'anglais, la conversion du format (mimetype) de l'image et le transmodage[1] de la vidéo vers l'audio peuvent être exécutées en parallèle par des adaptateurs correspondants, éventuellement offerts par des nœuds dans le réseau. La réduction de l'image ne se fera qu'après la conversion de son format.

Les médias adaptés doivent être re-synchronisés et réassemblés pour former un document multimédia composé adapté aux préférences de Suzy ainsi qu'aux capacités de son contexte.

Si le pair réalisant la traduction du texte disparaît, ou que Suzy décide de s'affranchir de l'audio, le système qui gère l'adaptation doit s'adapter à ces changements.

Nous déduisons de ce scénario, les besoins suivants :

- Fournir un système d'adaptation évolutif et pouvant être déployé à large échelle, compte tenu du nombre croissant d'utilisateurs des applications d'échange de contenus multimédia et l'évolutivité de ces contenus.

1. Le changement de la modalité d'un média. Par exemple, passer d'une vidéo à une image.

- Prendre en compte les caractéristiques personnelles ou les besoins spécifiques de l'utilisateur (Suzy ne veut pas de vidéo) : peu de solutions répondent aujourd'hui à ce besoin.
- Réaliser une adaptation au niveau des intermédiaires : l'accroissement des utilisateurs des terminaux à capacités réduites telles que les assistants personnels (par ex. PDA ou blackberry) exclut une adaptation côté client (ou utilisateur final). L'adaptation côté serveur (c'est-à-dire à la source du document multimédia), nécessite l'implémentation de modules supplémentaires, ce qui n'est pas toujours possible et ce qui peut créer une charge supplémentaire indésirable
- Gérer les disparitions des ressources d'adaptation : ce besoin est une conséquence de l'adaptation distribuée au niveau des intermédiaires.

Il existe plusieurs solutions architecturales qui adaptent des contenus multimédia au contexte d'un utilisateur. Citons, par exemple, les plates-formes ADMITS [BHK03], APPAT [LRe05], ISIS [ISIS02], UMA [Mag02] et NAC [Lem04], DCAF [GBP05], M21 [RBu04], MAPS [LHC03] et SATO [MKS06]. L'analyse de ces architectures (que nous présentons dans le chapitre 2) montre qu'elles ne répondent pas à tous les besoins cités ci-dessus.

En réponse à ces besoins, nous apportons, dans cette thèse, deux contributions : la réalisation d'une architecture d'adaptation de documents multimédia composés par rapport au contexte des utilisateurs, et la gestion des adaptateurs distribués.

La première contribution est la conception et l'implémentation d'une architecture appelée PAAM (pour *Architecture for the **P**rovision of **A**d**A**ptable **M**ultimedia composed documents*). Cette architecture a pour but d'adapter les documents multimédia composés au contexte des usagers. L'une des originalités de cette architecture est de mettre en place une adaptation distribuée sur différents nœuds du réseau en évitant de dédier l'adaptation à un serveur ou à un intermédiaire. PAAM permet d'intégrer aussi bien des fournisseurs de services d'adaptation que des particuliers qui se porteraient volontaires pour exécuter des fonctions d'adaptation en donnant un peu de leurs ressources matérielles et logicielles. Les principaux éléments fonctionnels de PAAM sont : le gestionnaire du contexte utilisateur, le gestionnaire des documents multimédia composés, le planificateur et le gestionnaire d'adaptation. Le gestionnaire du contexte utilisateur récupère et analyse les informations pertinentes relatives au contexte de l'utilisateur. Le gestionnaire des documents multimédia composés prend en entrée un document multimédia composé et analyse les informations descriptives de ce document. Le planificateur récupère les informations analysées depuis le gestionnaire de contexte et le gestionnaire des documents et décide si un document a besoin d'être adapté ou non. Le planificateur implémente un algorithme de prise de décision reposant sur des politiques d'adaptation et produit un graphe d'adaptation. Un graphe d'adaptation est un ensemble d'adaptateurs assemblés en parallèle ou en séquence. Ce graphe est utilisé en entrée du gestionnaire d'adaptation. Celui-ci recherche les adaptateurs, les instancie, les compose, et si nécessaire, les recompose si un ou plusieurs adaptateurs n'existent pas. Il gère également la disparition des adaptateurs instanciés.

Notre choix d'implémentation s'est porté sur les services Web qui, grâce à leurs propriétés, permettent à PAAM de passer à l'échelle, ainsi que d'être extensible et flexible. PAAM permet également la tolérance aux disparitions.

La gestion des adaptateurs (description, recherche et instanciation) est la seconde contribution de cette thèse. Nous proposons une nomenclature incluant un grand nombre d'adaptateurs. Nous proposons aussi une description d'adaptateurs qui étend WSDL, et qui facilite la recherche, l'instanciation et la composition de ces ressources d'adaptation. Afin de gérer l'équilibrage de charge, le gestionnaire d'adaptation choisit les adaptateurs en appliquant un protocole de négociation et d'acceptation.

Afin de démontrer la faisabilité de l'architecture PAAM, nous implémentons une chaîne d'adaptation complète incluant les principales fonctionnalités de PAAM : le gestionnaire du contexte utilisateur, le gestionnaire des documents multimédia composés, le planificateur et le gestionnaire d'adaptation.

Nous présentons, par la suite, une étude des coûts induits par notre implémentation de PAAM et des tests de performances qui montrent que l'utilisation des services Web n'introduit pas de surcoûts significatifs par rapport au gain obtenu en distribuant et en parallélisant l'adaptation sur différents nœuds.

2 Plan du rapport

Cette thèse est organisée comme suit :

Chapitre 2 : Adaptation de contenus multimédia : état de l'art

Ce chapitre présente un état de l'art sur les architectures de fourniture de contenus médias ou multimédia adaptables au contexte d'un utilisateur. Nous utilisons le scénario d'adaptation introduit au Chapitre 1 et nous l'illustrons lors de la description des architectures afin de les comparer. Ce chapitre présente également les outils les plus souvent utilisés pour décrire le contexte d'un utilisateur et les documents multimédia composés.

Chapitre 3 : PAAM : Une architecture pour la fourniture de contenus multimédia adaptables

Ce chapitre présente notre première contribution : PAAM, une architecture qui fournit, à des utilisateurs, des documents multimédia composés adaptés aux contextes de ces utilisateurs. Après avoir situé PAAM sur un plan architectural, économique et technologique, nous détaillons chaque bloc de cette architecture en illustrant les entrées/sorties. Nous discutons également du passage à l'échelle de PAAM.

Chapitre 4 : PAAM et l'utilisation des services Web

Ce chapitre justifie notre choix pour les services Web utilisés pour implémenter PAAM. Nous présentons les principaux standards des services Web : WSDL, SOAP et UDDI, en partie utilisés dans l'implémentation de PAAM. Nous présentons également BPEL, un outil qui peut être utilisé dans un gestionnaire d'adaptation afin de composer et d'orchestrer les services Web d'adaptation.

Chapitre 5 : Gestion des adaptateurs distribués et de la dynamicité

Ce chapitre présente notre deuxième contribution : l'identification unique et la description des services Web d'adaptation ainsi que la gestion des adaptateurs. Nous exposons également un protocole de négociation et d'acceptation qui permet à un gestionnaire d'adaptation de choisir le bon adaptateur.

Chapitre 6 : Implémentation d'une chaîne d'adaptation complète

Ce chapitre illustre une chaîne d'adaptation complète réalisant les fonctionnalités de base de PAAM. Après avoir exposé et justifié nos choix technologiques ainsi que nos solutions pour décrire le contexte utilisateur et les documents multimédia composés, nous présentons un exemple d'algorithme de prise de décision basé sur des politiques d'adaptation.

Chapitre 7 : Étude des coûts et tests

Ce chapitre a pour objectif d'évaluer le coût des adaptations réalisées avec PAAM. Nous procédons par comparaison de PAAM avec une architecture d'adaptation de documents multimédia composés par rapport au contexte d'un utilisateur, et qui, contrairement à PAAM, ne distribue pas l'adaptation. Cette architecture de référence nous sert à identifier les fonctionnalités additionnelles de PAAM (telles que la gestion de l'adaptation distribuée) qui sont susceptibles d'induire des coûts supplémentaires. Nous mesurons par ailleurs ces coûts.

Chapitre 8 : Perspectives et conclusion

Ce dernier chapitre conclut notre rapport et présente les perspectives de nos travaux.

Chapitre 2

Adaptation de contenus multimédia : état de l'art

1 Introduction

Dans ce chapitre, nous commençons d'abord par définir l'adaptation et par déterminer le type de contenus multimédia ainsi que les applications auxquels nous nous intéressons. Nous décrivons ensuite les fonctionnalités de base de toute architecture de fourniture de contenus multimédia adaptés aux contextes des usagers ainsi que les types d'adaptation qu'elles peuvent réaliser. Ceci nous sert, par la suite, à analyser un ensemble d'architectures d'adaptation de contenus multimédia par rapport aux contextes des utilisateurs. Nous rappelons alors le scénario présenté dans le chapitre précédent qui nous permet de comparer ces architectures. Enfin, avant de conclure, nous citons les principaux standards et outils permettant de décrire le contexte utilisateur et les contenus multimédia afin de justifier nos choix.

2 Adaptation : pourquoi et quoi ?

2.1 Définition de l'adaptation

L'adaptation signifie un changement dans le système afin d'accommoder un changement dans son environnement [SCh01].

Cette dernière définition introduit la notion d'environnement. Nous l'associons à la notion de contexte que nous définissons dans ce qui suit.

2.2 Pourquoi adapter ?

Considérant les adeptes des vidéos et images à la demande, et les inconditionnels des échanges de fichiers, nous constatons que le nombre d'utilisateurs d'applications multimédia via le Web ne cesse de croître.

Chaque utilisateur peut être caractérisé par son profil personnel (par exemple l'âge, les centres d'intérêts ou encore le handicap), ses préférences de présentation de contenu (par exemple, un utilisateur peut préférer les images en noir et blanc ou les vidéos avec un format bien particulier), les

caractéristiques de son terminal (par exemple la taille ou la résolution de l'écran et les codecs disponibles) et les caractéristiques de son réseau d'accès (par exemple la bande passante). Toutes ces informations sont fréquemment regroupées sous l'appellation de « *contexte utilisateur* ».

Les contenus multimédia ont également des caractéristiques variées. Tout d'abord, il existe un certain nombre de modalités (médias ou « mimetypes ») différentes : image, audio, vidéo, texte, texte animé, image 3D, vidéo 3D, etc. Chaque média est caractérisé par un certain nombre d'éléments tels que le format (par exemple WMV ou AVI pour les vidéos ou JPG ou GIF pour les photos), la taille, le taux de compression ou les dimensions. Les types de contenus multimédia sont donc variés.

Face à ce constat, et compte tenu du contexte propre à chaque utilisateur, deux solutions sont envisageables :

1. Avoir autant de versions du document que de contextes utilisateurs différents. Bien qu'en théorie assez facile à réaliser, cette solution génère une explosion combinatoire des configurations possibles d'un document donné, si nous décidons, par exemple, de tenir compte de tous les types de terminaux. WURFL recense plus de 5000 différents terminaux mobiles en prenant en considération un ensemble de capacités [WURFL]. En considérant également d'autres types de terminaux tels que les ordinateurs personnels (PC), cette solution parait donc inenvisageable du point de vue des producteurs de contenus multimédia qui souhaitent produire des documents multimédia sans se préoccuper des contextes des consommateurs.
2. Adapter ces contenus par rapport aux contextes des utilisateurs. Il s'agit de rendre possible cette consommation, tout en adaptant, au besoin, les contenus d'origine au lieu de les produire, au préalable, sous différentes versions.

Ainsi, afin de ne pas surcharger les producteurs de contenus avec des traitements supplémentaires, il est logique d'adapter des contenus multimédia par rapport aux contextes des utilisateurs plutôt que de les produire en autant de versions que de contextes utilisateurs.

2.3 Qu'allons nous adapter ?

[Her94] décrit deux aspects du multimédia :

- Mélange et intégration dans un même document d'éléments de natures différentes : images fixes ou animées, sons, textes, programmes informatiques, données diverses.
- Possibilité pour l'utilisateur de naviguer à sa guise d'une information à l'autre.

L'avènement de nouvelles technologies dans le monde du multimédia et du Web est à l'origine de la multiplicité des contenus multimédia échangés à travers Internet et de la diversité d'outils et de fonctionnalités permettant d'y accéder. Dans le cadre de nos travaux, nous nous intéressons aux applications d'échange de contenus multimédia riches. Nous considérons plus particulièrement les documents multimédia composés. Nous adoptons la définition avancée par [Kim05a] [Kim05b] qui présente un ***document multimédia composé*** *comme un document où des médias élémentaires (vidéo,*

image, audio, texte,…etc.) sont spatialement organisés et temporellement synchronisés. Par exemple, une description d'un document multimédia composé de deux vidéos peut suggérer qu'une des deux vidéos commence après la fin de la seconde et soit placée à sa droite.

3 Caractéristiques des architectures de fourniture et d'adaptation de contenus multimédia

Nous décrivons, dans les prochains paragraphes, les plates-formes et architectures dont le but est de fournir des contenus multimédia adaptés au contexte de l'utilisateur final de la manière la plus transparente possible.

Afin de bien classifier les architectures existantes, nous devons répondre à la question suivante :

Qu'est ce qui caractérise une architecture de fourniture de contenus multimédia adaptables ?

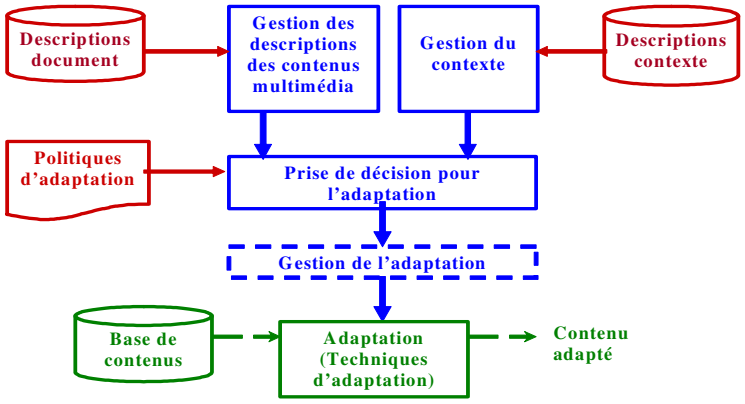

Figure 2 : Fonctionnalités de base d'une architecture de fourniture de contenus multimédia adaptables au contexte d'un utilisateur

Nous distinguons un ensemble de fonctionnalités de base communes à presque toutes les architectures de fourniture et d'adaptation de contenus multimédia et qui sont (cf. Figure 2) :

- La gestion du contexte utilisateur et de sa description,
- la gestion des contenus multimédia et de leurs descriptions,
- la gestion de la prise de décision pour l'adaptation,
- la gestion de l'adaptation - éventuellement distribuée-, et
- éventuellement la gestion des changements dynamiques dans le contexte utilisateur.

Les sections qui suivent détaillent chaque fonctionnalité. Ces descriptions servent de base dans la comparaison entre les architectures de fourniture de contenus multimédia présentées dans la section 4.

3.1 Gestion du contexte

Le mot contexte tel que trouvé dans le dictionnaire Le Petit Larousse est définit comme suit :« un *ensemble de circonstances dans lesquelles s'insère un fait, une situation globale où se situe un événement* ».

D'une manière plus générale, [Dey00] voit le contexte comme :

« Toute information qui peut être utilisée pour caractériser la situation d'une entité. Une entité est une personne, un endroit ou un objet considérés comme pertinents lors d'une interaction entre l'utilisateur et les applications elles-mêmes ».

D'après [Dey00], un système est dit **sensible au contexte** s'il utilise le contexte afin de fournir l'information et/ou les services pertinents à l'utilisateur, où la pertinence dépend de la tâche de l'utilisateur.

Nous nous inspirons de la définition du contexte de Dey en le définissant comme toute caractéristique ou capacité d'une entité de l'environnement (utilisateur, terminal, réseau, environnement naturel de l'utilisateur, état et caractéristiques des modules réalisant l'adaptation) qui peut influer sur la manière de consommer un service (contenu multimédia) donné, et qui est susceptible de changer dans le temps suite à des événements.

La gestion du contexte s'appuie d'une part sur un langage de description des éléments du contexte d'un utilisateur et d'autre part sur les outils pour collecter ces données, les comprendre et les analyser.

En général, ces langages décrivent tout ou partie des informations contextuelles telles que :

- les informations du profil personnel de l'utilisateur (par exemple le nom, le prénom, l'age et les centres d'intérêt),
- les préférences de l'utilisateur en terme de présentation des contenus multimédia (par exemple la taille et résolution souhaitée d'une image, la couleur d'affichage et la police de caractère d'un texte et les formats préférés d'un contenu vidéo),
- les capacités de son terminal (par exemple la taille de l'écran, les types des agents utilisateurs, les codecs disponibles et la résolution), et
- les caractéristiques de son réseau d'accès et de son environnement naturel (par exemple la valeur de la bande passante, l'indice de luminosité et de bruit de l'environnement de l'utilisateur).

Le système d'adaptation des contenus multimédia doit être muni des bons outils lui permettant de décrire, de stocker, de récupérer et d'analyser ces informations contextuelles, qui, agrégées, constituent un des éléments clés dans le processus de prise de décision pour l'adaptation.

La gestion du contexte peut se faire de manière distribuée : au niveau de l'utilisateur et au niveau du gestionnaire de contexte qui est un intermédiaire. Cette configuration peut se produire dans plusieurs

cas comme lorsque le client ne souhaite pas mettre à disposition d'une entité tiers tout ou une partie de ses informations personnelles.

Il existe des standards dont le but est d'offrir les bons mécanismes pour décrire le contexte de l'utilisateur tels que CC/PP [CCPP] et MPEG-21 [MPEG21].

La gestion du contexte n'étant pas une de nos contributions, nous décrivons brièvement ces langages de description du contexte dans la section 6.

3.2 Gestion des contenus multimédia

Cette fonctionnalité, bien qu'ignorée dans les premiers systèmes d'adaptation, trouve tout son intérêt lorsqu'il s'agit d'adapter des contenus multimédia riches. Ainsi, en plus des données descriptives relatives au contexte, un système d'adaptation de contenus multimédia a également besoin des descriptions de ces contenus afin de mieux prendre des décisions pour l'adapter. Ces données descriptives sont souvent appelées méta-données (*metadata* en anglais).

Une méta-donnée (du grec *meta* "après" et du latin *data* "informations") est une donnée servant à définir ou décrire une autre donnée quel que soit son support (papier ou électronique).

L'intérêt de l'utilisation des méta-données dans une page Web ou un site est de faciliter la recherche d'information en décrivant le contenu et les relations entre les fichiers d'un site, en classifiant le contenu suivant un degré de difficulté ou un public cible, ou encore en référençant au mieux un site ou une page sur Internet.

Sur un autre plan, les méta-données permettent de partager et d'échanger des informations (interopérabilité), facilitent la gestion et l'archivage des ressources électroniques, gèrent les droits d'accès, protègent les droits de propriété intellectuelle et offrent la possibilité d'authentifier un texte en encodant par exemple une signature électronique.

Concernant la gestion des contenus multimédia, plusieurs raisons peuvent pousser un producteur de contenus multimédia à fournir une description, soit accompagnée du document multimédia qu'il souhaite partager, soit carrément incorporée au sein même du document multimédia lorsqu'il s'agit d'un contenu auto-descriptif comme un contenu SMIL [SMIL]. Ainsi un producteur de contenus multimédia (ou serveur de contenus multimédia) peut souhaiter :

- décrire une vidéo scalable à l'aide de MPEG-7 [MKP02, DMa02] en vue d'une reconstruction/adaptation de la vidéo,
- décrire l'enchaînement temporel et l'emplacement spatial relatifs à plusieurs médias composant un document multimédia composé (ex. deux vidéos et une bande son) à l'aide de SMIL [SMIL] en vue d'une reconstruction/adaptation de document,
- décrire les dimensions et le format d'une image en vue d'un re-dimensionnement ou d'un changement de format de l'image.

Nous décrivons quelques outils de description de contenus multimédia dans la section 7.

3.3 Gestion de la prise de décision

La gestion de prise de décision pour l'adaptation est le cœur même de toute architecture de fourniture de contenus multimédia adaptables. Le module concerné est souvent appelé moteur de prise de décision.

Le but de cette fonctionnalité est de déterminer si le contenu demandé par l'utilisateur nécessite une adaptation ou non. Pour ce faire, il est nécessaire d'analyser, d'une part, les informations contextuelles de l'utilisateur, et d'autre part, les informations relatives aux contenus multimédia demandés. La prise de décision peut être guidée par des *politiques d'adaptation*.

Dans [MLS04], les politiques d'adaptation sont des règles déclaratives gouvernant les choix dans le comportement d'un système. Ces auteurs distinguent deux types de politiques : les politiques d'adaptation d'autorisation et les politiques d'adaptation d'obligation. La première famille de politiques est utilisée pour définir les ressources et les services auxquels un objet (par exemple un gestionnaire ou un utilisateur) peut accéder. La deuxième famille concerne des règles « condition / action » à base d'événements, utilisées pour, par exemple, définir les conditions de réservation des ressources réseau.

Une application qui fonctionne grâce à des politiques d'adaptation doit interpréter ces politiques afin de s'adapter aux besoins d'utilisateurs spécifiques. Un exemple de politique est de déterminer quelle information doit être filtrée lorsque la bande passante ou les capacités d'un terminal deviennent limitées.

Les règles d'obligation correspondent le mieux à la problématique de la prise de décision pour adapter un contenu multimédia. En effet, les actions choisies par le module de prise de décision sont précédées de conditions. Il s'agit d'étudier les contraintes résultant de l'analyse des données du contexte de l'utilisateur et des descriptions des contenus et de prendre la meilleure décision d'adaptation grâce à l'implémentation d'algorithmes d'adaptation. Ainsi, chaque politique est analysée et une seule ou plusieurs sont appliquées. Il est également possible que la prise de décision repose sur ces politiques, le moment venu, afin de choisir entre deux ou plusieurs décisions possibles.

La prise de décision peut concerner des médias simples (élémentaires) et des adaptations simples telles que la réduction de taille d'une image ou le changement d'encodage d'une bande son. Cependant, la prise de décision peut devenir assez complexe dans le cas d'adaptation d'un document multimédia composé de plusieurs médias tels qu'une vidéo, une image ou du texte. Dans ce cas, nous pouvons, par exemple, tenir compte de chaque média élémentaire et décider individuellement quelle adaptation ou ensemble d'adaptations sont nécessaires, ou encore quel média supprimer du document final. Ainsi s'établit une liste d'adaptations à réaliser en séquence ou en parallèle. Certains utilisent le terme de plan d'adaptation, d'autres parlent de graphe d'adaptation. Le but est de rendre les adaptations aussi indépendantes les unes des autres dans l'optique de paralléliser le maximum de traitements adaptatifs. Il

est nécessaire de reconstruire, par la suite, le nouveau document multimédia qui sera composé des médias d'origine et/ou des média adaptés.

3.4 Gestion de l'adaptation de contenus multimédia

Cette fonctionnalité n'est pas commune à toutes les architectures et est souvent intégrée à la fonction de gestion de la prise de décision. Nous avons décidé de la considérer comme une fonctionnalité à part entière car elle peut s'avérer assez complexe dans certains cas. En effet, une fois la phase d'identification des adaptations à réaliser finie (gestion de la prise de décision), il s'agit de déterminer le ou les endroits où sera réalisée l'adaptation.

Le module responsable de la gestion de l'adaptation doit fournir des outils pour :

- identifier le ou les module(s) (ressource(s)) capable(s) de réaliser les différents processus d'adaptation,
- les localiser,
- les instancier,
- et gérer le bon déroulement des adaptations.

En d'autres termes, si nous reprenons la terminologie citée ci-dessus, il faut instancier la suite des adaptations à réaliser en trouvant ou en choisissant les bons adaptateurs et assurer un bon déroulement de leur exécution.

Dans la section qui suit, nous présentons les endroits où peut être réalisée l'adaptation, suivant les différentes configurations architecturales existantes.

3.5 Adaptation des contenus multimédia

Nous nous intéressons, dans cette section, à l'identification des différentes solutions existantes relatives à la localisation des modules qui réalisent l'adaptation. Ceci peut justifier ou non, la nécessité de la fonctionnalité de gestion de l'adaptation. Nous présentons par la suite les différentes techniques d'adaptation et types d'adaptation pris en compte par une architecture de fourniture de contenus multimédia adaptables.

3.5.1 Modèle architectural client / serveur et adaptation

L'architecture **client/serveur** désigne un mode de communication entre plusieurs ordinateurs d'un réseau qui distingue un ou plusieurs postes clients du serveur : chaque application cliente peut envoyer des requêtes à un serveur. Un serveur peut être spécialisé en serveur d'applications, de fichiers, de terminaux, ou encore de messagerie électronique.

Un **serveur** est passif (ou esclave), à l'écoute, prêt à répondre aux requêtes envoyées par des clients. Dès qu'une requête lui parvient, il la traite et envoie une réponse.

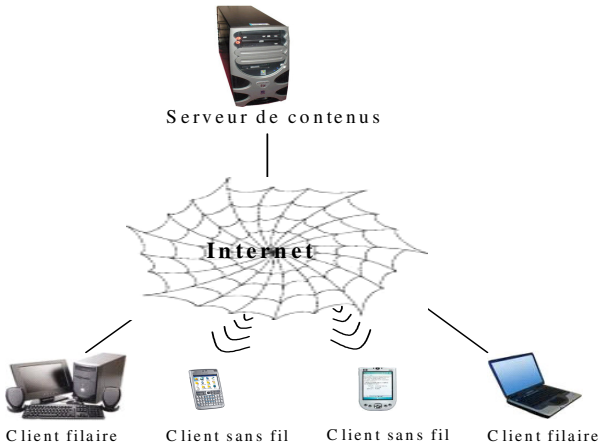

Figure 3 : Modèle architectural client / serveur

Un **client** est, contrairement au serveur, actif (ou maître). Il envoie des requêtes au serveur, attend et reçoit les réponses du serveur.

Le client et le serveur doivent bien sûr utiliser le même protocole de communication. Un serveur est généralement capable de servir plusieurs clients simultanément.

Les architectures de fourniture de contenus adaptables basées sur le modèle client / serveur implémentent leur logique d'adaptation soit au niveau du client soit au niveau du serveur.

Dans ce qui suit, nous décrivons comment est réalisée l'adaptation au niveau des clients et des serveurs.

3.5.1.1 Système d'adaptation au niveau du client

La logique d'adaptation dans ces architectures se situe au niveau du client. Dans ce cas, le contexte de l'utilisateur, défini le plus souvent par ses préférences et les capacités de son terminal, est stocké et analysé au niveau du client. Le serveur est ainsi déchargé de tout traitement sur les médias qu'il envoie tels quels au consommateur.

Plusieurs techniques d'adaptation peuvent être réalisées au niveau du terminal client si les capacités logicielles et matérielles le permettent.

La plus simple et également la plus courante des adaptations orientées client est la transformation à la volée d'un contenu. Un exemple est l'utilisation des feuilles de style telles que XSLT (Extensible Style Sheet Language) ou CSS (Cascading Style Sheets) pour transformer des documents ou définir le *look and feel* des langages à balises comme XML.

Les travaux présentés dans [Parth06] proposent de nouveaux sous-systèmes d'affichage dans les terminaux légers qui garantissent une moindre consommation d'énergie. Les auteurs introduisent des changements aussi bien matériels, en introduisant par exemple une technologie d'affichage (par ex. OLED *Organic Light-Emitting Diode*) qui crée des régions d'affichage indépendantes les unes des

autres en terme de consommation d'énergie, que logiciels en concevant des interfaces utilisateur sensibles à l'énergie.

L'adaptation étant réalisée au sein d'une entité bien déterminée (le terminal client), la fonctionnalité de gestion de l'adaptation n'est pas indispensable.

3.5.1.2 Système d'adaptation au niveau du serveur

Ces systèmes implémentent la logique d'adaptation à la source, c'est-à-dire au niveau du serveur de contenus. Il est intéressant de considérer cette solution lorsque l'adaptation ne peut être réalisée au niveau du client et cela à cause de ses limitations logicielles et/ou matérielles, ou lorsque l'utilisateur ne souhaite pas utiliser ses ressources pour autre chose que la consommation du contenu demandé adapté à son contexte.

L'adaptation côté serveur trouve tout son intérêt lorsque le producteur de contenus multimédia (serveur) est désireux de préserver ses droits d'auteur. Rajoutons à cela la possibilité pour un producteur de sélectionner parmi ses versions de contenus ou de les adapter. Le producteur est assuré ainsi d'offrir un contenu intègre à ses utilisateurs ce qui n'est pas forcément le cas si l'adaptation se fait au sein d'une entité intermédiaire.

Le serveur dans ce cas possède deux fonctions : produire le document d'origine et en réaliser plusieurs versions correspondant aux différents contextes utilisateur. Adapter au niveau du serveur suggère ainsi une très grande capacité de stockage et de traitement compte tenu de la diversité des contextes utilisateurs.

L'adaptation étant réalisée au sein d'une entité bien définie (le serveur), la fonctionnalité de gestion de l'adaptation n'est pas indispensable.

3.5.2 Modèle architectural client / intermédiaire (s) / serveur et adaptation

Les architectures client / intermédiaire(s) (*proxies* en anglais) / serveur sont de plus en plus adoptées au dépens du modèle classique client / serveur. Un module intermédiaire est un nœud inséré entre le client et le serveur et dédié, par exemple, à la découverte, à l'annonce ou encore à l'adaptation d'un service, d'un contenu ou d'une ressource (ex : la recherche d'une référence dans une base de données, la gestion d'un annuaire distribué, le transcodage[2] d'une vidéo ou la réduction d'une image). L'intermédiation ainsi réalisée apporte une vraie valeur ajoutée en évitant de charger le client et le serveur de tâches spécifiques consommatrices de ressources sans rapport avec le service final offert. La Figure 4 illustre ce modèle architectural.

2. La conversion d'un objet ou d'un fichier média d'un format à un autre

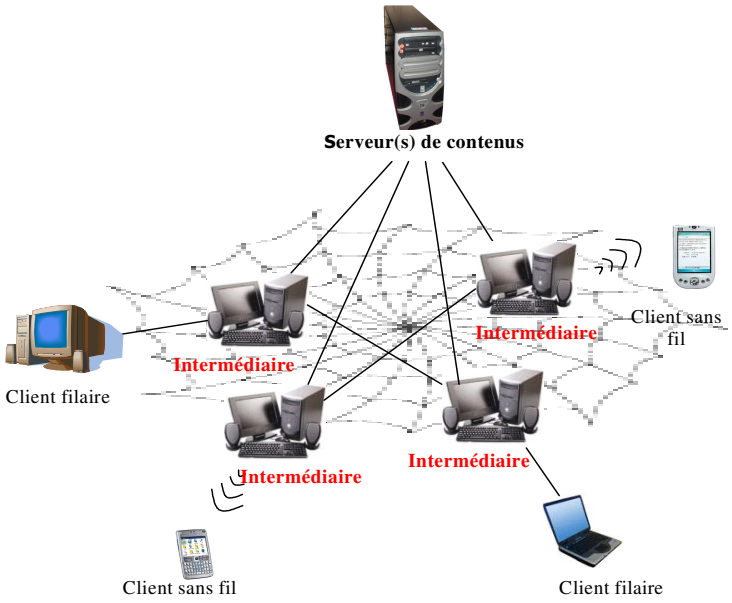

Figure 4 : Modèle architectural client / intermédiaire(s) / serveur

Le nombre d'intermédiaires d'adaptation pouvant varier entre 1 et N, l'emplacement de la logique d'adaptation ainsi que l'adaptation des contenus peut varier d'une solution à une autre. Nous distinguons les deux cas suivants :

- *La prise de décision et l'adaptation se font sur une même machine intermédiaire* : Il s'agit des architectures *client* / ***intermédiaire*** / *serveur*. L'intermédiaire interprète les requêtes de l'utilisateur, récupère et analyse le contexte de cet usager et détermine la ou les adaptations nécessaires, en tenant compte éventuellement des descriptions du contenu demandé. Ce même intermédiaire récupère le contenu depuis la source, l'adapte et le retransmet au consommateur.

- La prise de décision se fait sur une machine intermédiaire et l'adaptation se fait de manière distribuée sur plusieurs intermédiaires : Il s'agit des architectures *client* / ***intermédiaires*** / *serveur*. La prise de décision est souvent réalisée au sein d'un intermédiaire. Ce dernier comme dans le premier cas, utilise les informations contextuelles de l'utilisateur et les informations relatives au contenu multimédia et identifie les adaptations à réaliser. Les types de contenus multimédia visés par ces architectures concernent souvent les documents multimédia composés de plusieurs médias élémentaires (par exemple un document composé d'une vidéo qu'il faut réduire et d'un texte qu'il faut traduire) ou les médias subissant plusieurs étapes d'adaptation (par exemple l'extraction de l'audio depuis un contenu audio-visuel suivie par le transcodage de l'audio extrait). L'apport de ce modèle architectural à N intermédiaires est de réaliser l'adaptation sur plusieurs sites de manière distribuée. Un effort supplémentaire de supervision et de synchronisation est nécessaire. La fonctionnalité de gestion de l'adaptation est dans ce cas indispensable.

3.5.3 Modèle architectural Pair-à-Pair et adaptation

Le modèle architectural Pair-à-Pair (P2P) est un modèle où chaque nœud (pair) est à la fois client et serveur. Les ressources sont ainsi annoncées par des noeuds et consommées par d'autres nœuds, qui peuvent également jouer le rôle d'annonceur. Les ressources offertes sont généralement des médias, nous proposons par la suite qu'elles soient aussi des adaptateurs. L'adaptation peut être alors réalisée sur un ensemble du réseau de pairs comme c'est le cas dans une adaptation au niveau des intermédiaires (cf. section précédente) en tirant partie des outils offerts par un tel modèle architectural tels que les outils d'annonce, de recherche et de découverte des ressources. Dans ce cas, la fonctionnalité de gestion de l'adaptation est indispensable car il faut chercher et trouver les bonnes ressources d'adaptation, les instancier, les orchestrer et gérer leur disparition.

La Figure 5 illustre ce modèle architectural.

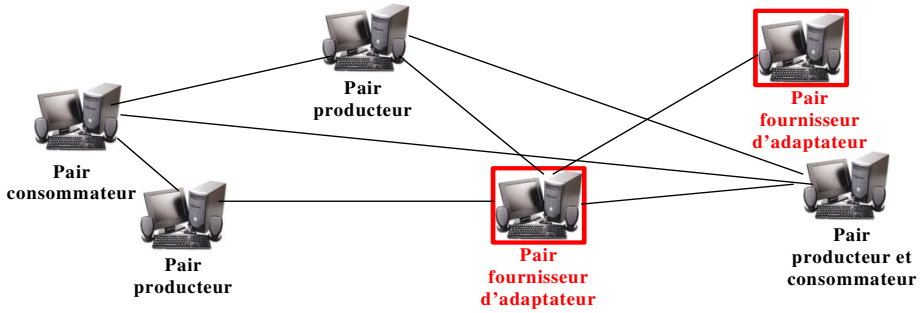

Figure 5 : Modèle architectural Pair-à-Pair (P2P)

3.5.4 Discussion : adaptation orientée client, serveur, intermédiaire(s) ou P2P ?

Comme nous venons de le voir, il existe plusieurs solutions architecturales répondant à la problématique d'adaptation de contenus multimédia. Cette constatation est due au fait que les applications relatives au multimédia, les types de contenus multimédia ainsi que les besoins d'adaptation varient énormément : à chaque problème une solution.

La gestion de l'adaptation au niveau du client a pour avantage de garantir une gestion locale du contexte de l'utilisateur. Ce dernier n'a pas à envoyer son contexte qui est susceptible de contenir des données personnelles de l'utilisateur. L'adaptation au sein du client peut également décharger le serveur de tout traitement supplémentaire. En revanche, les capacités logicielles et matérielles réduites de certains terminaux tels que les PDAs ou les téléphones portables (ressources d'adaptation limitées, charge mémoire, CPU et bande passante limités, etc.) rendent l'adaptation possible que pour certaines applications ou sur certains terminaux seulement. D'autres limitations venant s'ajouter à celle-ci sont relatives au fait que l'adaptation au niveau du client ne tienne pas compte des caractéristiques du réseau d'accès et ne permette pas d'adapter un contenu par rapport à la bande passante disponible étant donné que l'adaptation est réalisée une fois tout le contenu téléchargé.

L'adaptation orientée serveur offre, quant à elle, la possibilité de gérer les droits d'auteurs. En effet, qui mieux que le producteur (ou serveur) peut adapter un contenu tout en respectant le rendu final sans perte ni déformation ? Le serveur peut également proposer un certain nombre de versions pour un même contenu et cela par rapport aux systèmes d'exploitation (ex. Windows XP ou Windows 2000), à la bande passante (ex. 100K ou 300K), à la taille de la fenêtre de navigation (ex. petit ou grand format) ou au lecteur disponible sur le terminal destination (ex. RealOne ou QuickTime). Bien que très intéressante, cette dernière solution est inenvisageable compte tenu du nombre croissant de document et à la diversité des contextes utilisateurs. L'adaptation au niveau du serveur a ainsi comme inconvénient d'être gourmande en espace de stockage et en traitement CPU. Les architectures telles que ISIS, [ISIS] UMA [UMA] ou encore WAM offrent des systèmes de prise de décision et d'adaptation au niveau du serveur, même si cela doit impliquer d'ajouter des modules au serveur Web ou au serveur d'application.

L'architecture orientée intermédiaires tente de pallier aux limitations des solutions ci-dessus. En effet, adapter au niveau d'un ou de plusieurs intermédiaires évite d'utiliser les ressources du client (respectivement du serveur) autrement que pour la consommation (respectivement la production et le stockage) du contenu. Cette solution se prête également parfaitement à la mise à l'échelle et à l'évolutivité du système d'adaptation. Cependant, un choix architectural tel que celui-ci ne garantit pas forcément la préservation des droits d'auteur, la gestion des droits d'accès de l'utilisateur, ou l'accès sécurisé voire chiffré aux données. Cela dit, il n'est pas exclu de greffer, aux services d'adaptation, des systèmes offrant des services supplémentaires tels que des services de chiffrage, de paiement et facturation (*charge and billing*), de sécurité, et de transaction. [ADMITS, UMA, WAM] sont des exemples d'architectures qui réalisent l'adaptation au niveau d'un intermédiaire tandis que [APPAT, NINJA, MAPS et DCAF] réalisent l'adaptation au niveau de plusieurs intermédiaires.

Le modèle architectural P2P peut également pallier aux problèmes liés à l'adaptation au niveau du client et au niveau du serveur. Il apporte une souplesse et une évolutivité dans l'adaptation car chaque nœud peut être à la fois adaptateur, client et producteur de contenus multimédia. Ce modèle introduit néanmoins une gestion des adaptateurs plus complexe compte tenu de la volatilité des nœuds. Ce modèle architectural se prête également au passage à l'échelle à condition d'avoir une plate-forme d'exécution qui le permet. Par exemple, les plates-formes P2P implémentées avec JXTA [JXTA] ne passent pas à l'échelle. M21 [RBu04], MAPS [LHC03] et SATO [SATO06] intègrent une logique d'adaptation basée sur le modèle architectural P2P.

3.6 Techniques de consommations et d'adaptations de contenus multimédia

Nous venons de décrire les principales fonctionnalités que doit avoir toute plate-forme d'adaptation de contenus multimédia ainsi que les différents modèles architecturaux qu'elle peut adopter. Cette section présente tout d'abord les différentes manières de consommer un contenu multimédia. Elle illustre par la suite les différentes techniques d'adaptation et types d'adaptation pris en compte par une architecture de fourniture de contenus multimédia adaptables. Cela nous permet de situer notre travail par rapport aux techniques de consommation et d'adaptation de contenus multimédia existantes.

3.6.1 Consommation d'un contenu multimédia

Consommer un contenu multimédia pour un utilisateur peut se faire à l'aide de deux techniques :

- Télécharger tout le contenu et le visualiser (*téléchargement*) : le contenu est stocké au niveau du récepteur avant d'être consommé.
- Regarder le contenu au fur et à mesure (*streaming*) : le contenu est visualisé dès que des ressources du terminal utilisateur et/ou du réseau sont disponibles.

Les deux méthodes ont leurs avantages et leurs inconvénients. La première méthode ne dépend pas des conditions réseaux ; en revanche, elle requiert de l'espace de stockage au niveau du receveur et est consommatrice en temps. La seconde méthode, bien qu'elle économise du temps et de l'espace de stockage, dépend à tout moment des ressources réseau disponibles. La qualité de services est ainsi difficilement garantie étant donné l'hétérogénéité des réseaux d'accès, des terminaux et des conditions d'accès.

3.6.2 Catégorisation des techniques d'adaptation

[Kim05b] distingue trois grandes catégories d'adaptation : transformation (Transfoming), Transmodage (Transmoding), et Transcodage (Transcoding).

[MPe04] divise les solutions d'adaptation de contenu en trois catégories majeures: la summarization qui comprend la conversion de modalités (transmodage) et la réduction temporelle, la réduction de la qualité (transformations temporelle, spatiale et/ou fréquentielle des médias) et transcodage. Notons que la traduction littérale de summarization donnerait résumé mais nous pensons que ce terme est trop réducteur.

Malgré la différence de nommage des catégories de classification et ce qu'elles englobent comme techniques d'adaptation, les travaux cités plus haut tentent de répertorier les mêmes processus d'adaptation.

Dans ce qui suit, nous nous basons sur ces classifications en y incluant une dernière catégorie qui regroupe les techniques d'adaptation qui ne rentrent dans aucune de ces familles et qui n'ont, *a priori*, pas été considérées dans la littérature précitée. Ainsi, nous distinguons ces trois principales catégories d'adaptation : le transmodage, le transcodage, et la transformation.

- ***Le transmodage***, appelé également conversion de modalités, réfère au changement de mode (ou de modalité) d'un média. Il s'agit, par exemple, de transformer un texte en image de ce texte pour un client qui ne dispose pas de la police de caractère permettant l'affichage de ce texte.
- ***Le transcodage*** consiste à changer le format d'un média donné sans changer de modalité, de telle sorte que le contenu d'origine garde le même aspect. Un exemple serait celui de convertir une vidéo du format MOV au format vidéo AVI.
- ***La transformation*** réfère à tout changement d'un média donné sans modifier ni sa modalité ni son format. Ce processus transforme un contenu en réduisant sa taille par exemple, ou en

traduisant du texte. Cette catégorie concerne également des adaptateurs spécifiques qui reconstruisent un document multimédia de synchronisation de type SMIL ou MPEG-21 par exemple et qui prennent en entrée un document multimédia et le transforment en un autre document multimédia.

Le Tableau 1 résume de manière non exhaustive quelques unes des techniques d'adaptation relatives aux modalités les plus connues. Notons que pour des raisons de lisibilité, ce tableau ne tient pas compte des modalités (modes ou mimetypes) telles que Graphics2D, Graphics3D, texte animé, texte avec timing, son naturel synthétique et son naturel 3D.

	Transcodage	**Transmodage**	**Transformation**
Vidéo	- Convertisseur de format vidéo	- Vidéo vers image (snapshot) - Vidéo vers slideshow - Vidéo vers texte - Incrustation de sous-titres dans une vidéo (vidéo vers contenu multimédia) - Incrustation d'un avatar (vidéo vers contenu multimédia)	- Redimensionnement de vidéo - Changement de couleur - Restauration de couleur - Réduction de bruit - Ajusteur de couleur - Modification de débit de lecture d'images d'une vidéo - Compression de vidéo - Résumé - Changement de résolution
Image	- Convertisseur de format image	- Image vers vidéo - Image vers texte	- Compression - Changement de résolution spatiale - Changement de couleur
Audio	- Convertisseur de format audio	- Audio vers texte	- Resampler - Quantifier - Compression d'un contenu audio - Résumé
Texte	- Convertisseur de format texte	- Texte vers image - Texte vers slideshow - Texte vers audio	- Traduction - Résumé - Suppression d'espaces - Changement de la taille du texte
Audiovidéo			- Extraction d'un contenu audio - Extraction de la vidéo - Résumé
Multimédia			- Reconstruction d'un document multimédia

Tableau 1 : Classification de quelques techniques d'adaptation

4 État de l'art sur les architectures de fourniture de contenu multimédia

Cette section a pour but de présenter quelques travaux en relation avec notre problématique : la réalisation d'un système d'adaptation de contenus multimédia par rapport aux contextes des usagers.

Rappelons que les architectures de fourniture de contenus multimédia adaptés au contexte des usagers proposent des solutions différentes mais réalisent néanmoins les mêmes fonctionnalités :

- la gestion du contexte et de sa description
- la gestion des contenus multimédia et de leurs descriptions
- la gestion de la prise de décision pour l'adaptation
- la gestion de l'adaptation - éventuellement distribuée -

Il est important de déterminer, au sein de telles architectures, qui fait quoi, où, quand et comment se fait l'adaptation.

Ensuite, nous pouvons aussi les classifier suivant leur modèle architectural de fourniture et d'adaptation (le modèle client/serveur ou le modèle client/intermédiaire(s)/ serveur). Nous souhaitons ainsi déterminer où est réalisée l'adaptation : au niveau du client, au niveau du serveur, au niveau d'un ou de plusieurs intermédiaires ou au niveau d'un ensemble de pairs d'un réseau P2P.

Les solutions proposées par ces architectures se distinguent également par le type de contenus qu'elles traitent ainsi que par les méthodes d'adaptation qu'elles implémentent.

Certaines architectures comptent quelques fonctionnalités en plus telles que la gestion des changements qui peuvent se produire dans le contexte d'un utilisateur (par exemple les changements de préférences d'un utilisateur ou le changement du débit de la bande passante).

Nous présentons dans ce qui suit un état de l'art sur les architectures de fourniture de contenus multimédia adaptés aux contextes des utilisateurs. Les architectures que nous étudions sont : ADMITS [BHK03], APPAT [LRe05], ISIS [ISIS02], UMA [Mag02] et NAC [Lem04], DCAF [GBP05], M21 [RBu04], MAPS [LHC03] et SATO [MKS06]. Nous avons choisi de présenter des architectures car elles reposent sur des modèles architecturaux différents (cf. section 3.5), traitent des contenus multimédia différents et illustrent des diverses techniques d'adaptation. De plus, l'analyse de ces architectures permet d'en extraire les limitations et de situer ainsi nos objectifs.

4.1 ADMITS

ADMITS (**A**daptation in **D**istributed **M**ultimedia **IT** **S**ystems), a été initié dans l'université de Klagenfurt en Autriche en 2002 [BHK03]. Il vise à réaliser un système multimédia distribué et expérimental en prenant en compte différentes entités : serveur, intermédiaire (proxy) et clients. Ce projet implémente et évalue différents algorithmes d'adaptation de média.

ADMITS cherche à répondre aux questions : quand, où et comment adapter et comment des étapes d'adaptation individuelles et distribuées interopèrent et interagissent.

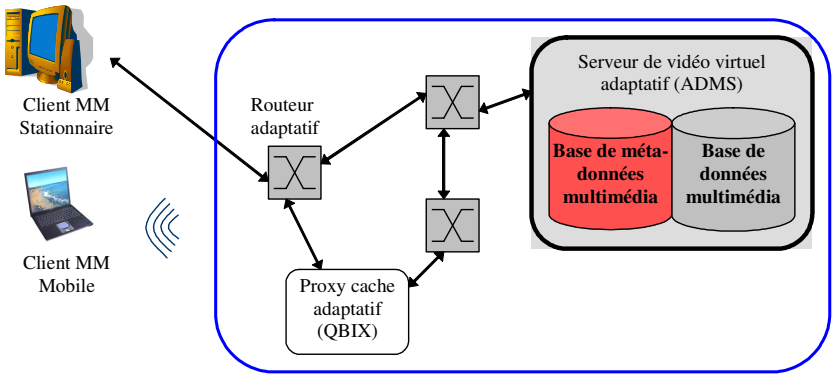

Figure 6: L'architecture du projet ADMITS

Cette architecture a été mise au point afin d'assurer les étapes depuis la création des méta-données jusqu'à la consommation du média, quel qu'il soit (vidéo, audio, etc.) [BHK03].

Le *serveur vidéo virtuel adaptatif* est construit au dessus de Vagabond2, qui est un intergiciel (*middleware*) basé sur CORBA et qui supporte l'instanciation, la migration, la réplication, et l'évacuation dynamique des composants [GTB02]. Il comprend quatre composants adaptatifs qui peuvent être combinés arbitrairement et s'exécuter sur un nombre arbitraire de nœuds du serveur. Ces composants doivent être indépendants, réutilisables, adaptables, déplaçables et combinables et doivent réaliser une tâche dédiée spécifique à un cas d'utilisation. Ces composants se présentent comme un ensemble de *Data Distributors (DDs), Data Managers (DMs), Data Collectors (DCs) et Cluster Managers (CMs)*. La Figure 6 montre l'interaction entre ces composants.

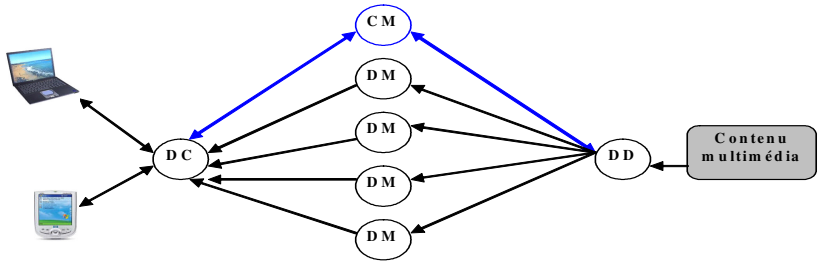

Figure 7: Les composants d'un serveur ADMS

Dans ce qui suit, nous nous intéressons à la manière dont sont affectées les tâches du processus d'adaptation aux acteurs.

La base de données des méta-données multimédia est implémentée sous forme de *MPEG-7 Multimedia Data Cartridge* [DKo03]. Cette base de données contient des informations relatives aux données multimédia stockées au niveau du serveur. Elle permet de traiter les requêtes basées sur le contenu ainsi que l'adaptation basée sur les méta-informations telles que la définition des procédures de transcodage en s'aidant des descripteurs de variations MPEG-7 proposées par ce projet.

QBIX (Quality Based Intelligent Proxy) est un intermédiaire qui supporte une adaptation temps-réel [GTB03] [SBH03]. QBIX utilise l'adaptation pour, entres autres, améliorer les stratégies de replacement du cache dans le proxy.

4.1.1 Gestion du contexte dans ADMITS

Dans la documentation relative au projet ADMITS, l'endroit où est réalisée la gestion du contexte de l'utilisateur n'est pas indiqué.

Nous avons cependant noté l'utilisation de CC/PP pour la description des préférences de l'utilisateur et des capacités de son terminal. Le stockage du contexte est local et il est utilisé par l'interface de présentation adaptative.

4.1.2 Gestion de la prise de décision dans ADMITS

Un module nommé *module MPEG-7* et se trouvant au niveau du proxy cache *QBIX* aide à parser et à générer des descriptions MPEG-7 et décide ainsi de l'adaptation à effectuer.

Le gestionnaire de cluster (Cluster Manager CM) est l'élément central qui gère les autres instances de composants du serveur virtuel. Il propose le nombre et la localisation des gestionnaires de données en appliquant les stratégies d'équilibrage de charge pour le réseau et les ressources. Le gestionnaire de cluster implémente un service de répertoire permettant de garder une trace des collecteurs de données qui stockent les données et quels flux élémentaires leur sont attribués et quels sont parmi ces collecteurs de données ceux qui gardent dans leur cache ces flux élémentaires et quels sont ces derniers.

4.1.3 Gestion de l'adaptation du service dans ADMITS

L'adaptation peut être faite à différents niveaux dans ADMITS : au niveau du proxy, du serveur ADMS, du routeur ou au niveau applicatif.

Le proxy cache QBIX comprend un module appelé *couche E/S (I/O layer)* qui permet de lire et d'écrire les données vidéo. Il comprend aussi un *moteur d'adaptation (adaptation engine)* qui utilise cette couche E/S pour récupérer les frames lues et les transformer. Le *gestionnaire de cache* qui compose le proxy cache applique les stratégies de remplacement du cache des vidéos en utilisant le moteur d'adaptation qui réalise l'adaptation et les méta-données décrivant les flux. Au fur et à mesure, le proxy réduit leur qualité (et donc leur taille) jusqu'à la suppression totale en s'aidant des méta-données MPEG-7 associées aux flux.

Les routeurs adaptatifs peuvent adapter les données multimédia mais d'une manière limitée afin de ne pas compromettre la vitesse de retransmission. Un routeur peut diminuer le nombre de frames ou enlever certaines couches d'amélioration ou même certains flux élémentaires MPEG-4.

Au niveau du serveur ADMS, le *distributeur de données (Data Distributor DD)* reçoit les informations depuis le serveur ou depuis une source externe et les distribue à un ensemble de gestionnaires de données.

Le *gestionnaire de données (Data Manager DM)* est un composant clé au sein de l'architecture d'un ADMS étant donné qu'il offre des outils de récupération et de stockage des flux élémentaires ou de leurs segments.

Le *collecteur de données (Data Collector DC)* réalise l'opération inverse du distributeur de données. Sa principale tâche est de répondre aux requêtes des utilisateurs en collectant les unités segmentées d'un flux donné provenant d'un ensemble de gestionnaires de données. Des mécanismes de cache ont été mis en place au niveau de ce module pour répondre aux problèmes de temps relatifs au chargement et à la consommation du média.

Enfin, l'utilisateur peut spécifier des critères de recherche des médias grâce à *l'interface de présentation adaptative* qui permet d'afficher les résultats, de proposer des outils pour choisir le média parmi les résultats affichés et d'ouvrir les lecteurs adaptés à chaque type de média. Un dernier niveau d'adaptation est réalisé dans ADMITS en chargeant dynamiquement des composants des interfaces suivant l'environnement d'usage (caractéristiques du terminal et préférences de l'utilisateur). Le

Fonctionnalités de base	**Gestion du contexte**	**- Au niveau de l'interface de présentation : CC/PP**
	Gestion des contenus multimédia	- Au niveau du proxy QBIX et/ou de ADMS : Utilisation de MPEG-7 pour décrire le flux
	Gestion de la prise de décision	- Au niveau du proxy QBIX : module MPEG-7 - Au niveau de ADMS : gestionnaire de Clusters
	Gestion de l'adaptation et localité de l'adaptation (distribuée)	- Au niveau du proxy QBIX : moteur d'adaptation - Au niveau des routeurs adaptatifs - Au niveau de ADMS : Gestionnaire de donnée en combinaison avec le gestionnaire et le collecteur de données. - Au niveau du client : Interface de présentation adaptative
	Gestion des changements dans le contexte	OUI
Techniques d'adaptation		- Réduction temporelle (frame dropping) - Réduction de la couleur de la vidéo - Réduction spatiale de la vidéo - Bitrate Scaling
Types de contenus adaptés		- Flux de données (vidéo MPEG-4) en streaming

Tableau 2 résume les fonctionnalités de ADMITS.

Tableau 2 : Fiche descriptive de ADMITS

4.2 APPAT

APPAT pour ***A**daptation **P**roxies **PlAT**form* est une plate-forme proposée par une équipe de recherche du LIFC (Laboratoire d'Informatique de l'Université de Franche-Comté) [LRe05].

APPAT une architecture à base de proxies (intermédiaires). Elle introduit la notion de distribution de l'adaptation sur plusieurs hôtes appelés *Proxies d'Adaptation* qui est poste aussi l'appellation « adaptation globale ». Cette thèse a comme cadre de travail les applications collaboratives.

La plate-forme APPAT gère la coordination des proxies, l'échange d'information et l'adaptation.

Un client est connecté à la plate-forme APPAT par l'intermédiaire d'un *Proxy d'Adaptation.*

L'idée derrière ces travaux est de réaliser l'adaptation globale qui est définie comme « l'adaptation des données en fonction de l'ensemble des paramètres impliqués par la multiplicité des utilisateurs ».

La Figure 8 montre plusieurs entités amenées à collaborer afin de réaliser l'adaptation globale.

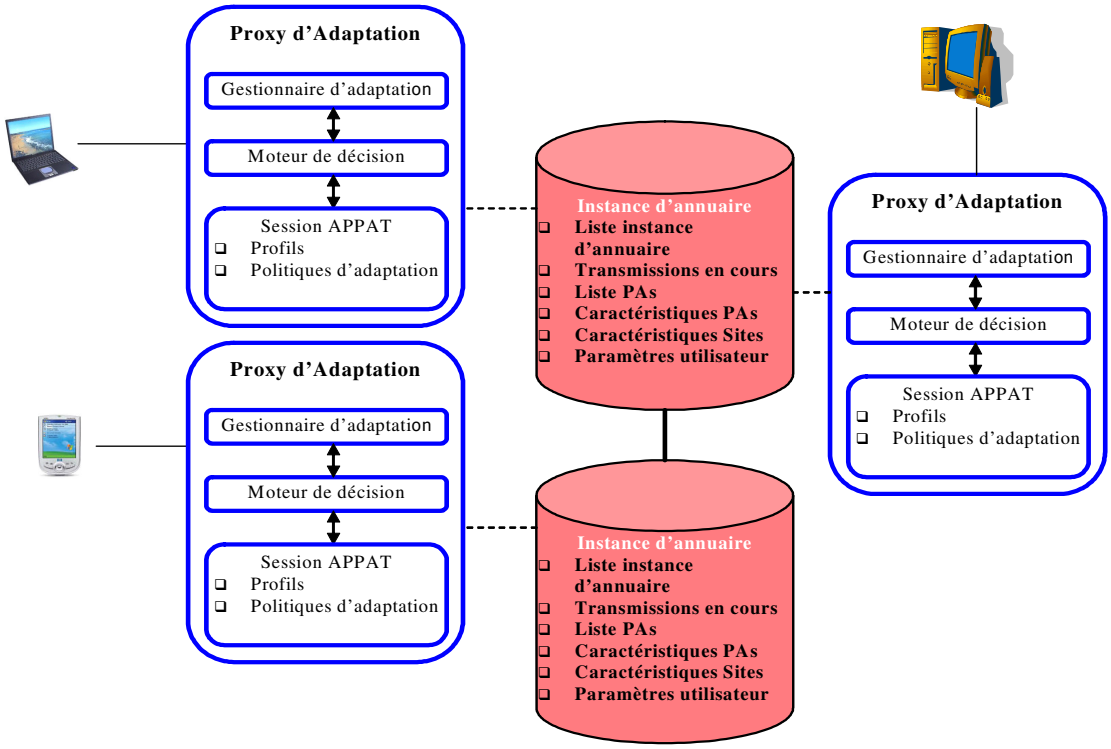

Figure 8: Architecture de la plate-forme APPAT

Un *Proxy d'Adaptation* (PA) est au cœur de la plate-forme APPAT. Il concerne trois fonctionnalités : la partie communication, la partie décision et enfin la partie adaptation. C'est à travers ceci que nous déduisons que APPAT correspond parfaitement à notre architecture de référence.

Un *site* est soit émetteur, soit récepteur. Ainsi, chaque site représente à la fois un serveur et un client.

Une *Interface de Proxy d'Adaptation* (IPA) est le point d'accès de chaque site à la plate-forme.

Un *annuaire distribué* est constitué de plusieurs *Instances d'Annuaire* (IA) qui partagent les informations sur l'ensemble de la plate-forme. L'annuaire distribué a des fonctionnalités similaires à un annuaire LDAP sauf que le protocole de communication entre les IAs repose sur HTTP/SOAP.

Une *Instance d'Annuaire* contient des informations sur l'état de la plate-forme notamment celles relatives aux PAs.

Les informations partagées par un annuaire distribué concernent la liste des IAs, des caractéristiques statiques et dynamiques des PAs et des sites, les paramètres utilisateurs pour chaque site et la liste des transmissions en cours.

4.2.1 Gestion du contexte dans APPAT

Ces travaux proposent trois manières pour un site ou un nœud de communiquer ses préférences et les caractéristiques de son terminal. La première idée est d'envoyer un profil CC/PP dans l'en-tête HTTP de la requête. Une deuxième idée est de proposer une page de configuration située au niveau de l'interface de proxy d'adaptation, et à travers laquelle les utilisateurs peuvent spécifier leurs préférences qui sont mémorisées. La dernière solution est celle qui a été proposée dans le cadre d'un projet autour d'une application de télé-neurologie appelé TeNeCi. Il s'agit d'avoir une application externe ou interne (plug-in) au client, indépendante de la requête, et permettant le contrôle de chaque client.

Le composant *Session APPAT* est un module existant sur chaque PA et permettant de collecter les informations du contexte via différentes interfaces :

L'interface annuaire permet au PA de s'inscrire et de se désinscrire, de se mettre à jour pour les informations communes et de localiser les autres PAs.

L'interface plate-forme permet à chaque PA de communiquer avec un autre PA.

L'interface client permet de réaliser la communication entre APPAT et un client donné en gérant les requêtes, les négociations et les transmissions de données.

Les informations collectées par le composant *Session APPAT* à travers ces différentes interfaces (qui correspondent aux informations partagées par l'annuaire distribué) sont conservées dans des tables pendant une durée limitée (par exemple une session).

Des MIBs (Management Information Base), qui sont des bases d'informations sur l'état d'une machine, sont déployées au niveau des sites et des PAs afin de récupérer l'état de l'activité réseau ainsi que celui des ressources des machines. Un client SNMP[3] au sein du composant *Session APPAT* met à jour ces MIBs. Il s'agit là d'une fonctionnalité appelée supervision des ressources des machines. Elle intervient à deux niveaux. Dans un premier temps, lors de la connexion d'un client pour permettre au moteur de décision de trouver la meilleure configuration d'une transmission en fonction de ses ressources. Dans un second temps, la supervision permet de répondre aux variations des ressources locales et distantes pendant une transmission de données continues.

3 Simple Network Management Protocol (protocole simple de gestion de réseau). Il s'agit d'un protocole de communication qui permet aux administrateurs réseau de gérer les équipements du réseau, superviser et de diagnostiquer des problèmes réseaux, matériels à distance

Le composant *Session APPAT* est indispensable pour envoyer ces informations au moteur de décision, un autre composant du PA.

4.2.2 Gestion de la prise de décision dans APPAT

Le moteur de décision au niveau du proxy d'adaptation (PA) permet de prendre des décisions sur l'adaptation. Il correspond dans l'architecture de référence au gestionnaire de prise de décision.

Il se base sur une procédure de décision multicritères prenant comme paramètres des informations relatives aux clients, aux PAs, au type de données à transmettre et à adapter.

Les politiques d'adaptation concernent le lieu d'adaptation. Trois cas peuvent se présenter : adapter au plus près de l'émetteur, adapter au plus près du récepteur et adapter au cœur de la plate-forme. Chacun de ces cas a une incidence sur l'état du réseau, sur les transmissions, et sur la qualité de service.

Quant aux profils, ils font correspondre des médias, des formats et des attributs à des caractéristiques données. Ils sont utilisés lorsque aucune préférence n'a été spécifiée, ou lorsque les préférences de l'utilisateur ne coïncident pas avec les caractéristiques du terminal récepteur.

Ces politiques et ces profils sont dupliqués sur chaque proxy d'adaptation.

4.2.3 Gestion de l'adaptation dans APPAT

Le gestionnaire d'adaptation met en place des mécanismes d'adaptation (propres à chaque proxy d'adaptation), démarre et arrête des processus d'adaptation (des processus sont des threads java qui reçoivent et transmettent des données, adaptées ou non) et modifie des paramètres d'adaptation.

Une liste des processus en cours est mise à jour par ce gestionnaire en archivant la référence de chaque processus en cours dans une table.

Les mécanismes d'adaptation introduits par ces travaux concernent le filtrage de données discrètes ou continues, le transcodage et le mixage de flux continus.

Le Tableau 3 résume les fonctionnalités intrinsèques à APPAT.

Fonctionnalités de base	**Gestion du contexte**	**-** Au niveau du client (requête http contenant un profil CC/PP) - Au niveau du IPA - Au niveau d'une application interne ou externe au terminal client - Au niveau du proxy d'adaptation : Composant Session
	Gestion des contenus multimédia	NON
	Gestion de la prise de décision	- Au niveau du proxy d'adaptation : Moteur de décision
	Gestion de l'adaptation et localité de l'adaptation	- Au niveau du proxy d'adaptation : Gestionnaire d'adaptation
	Gestion des changements dans le contexte	- Utilisation des MIBs
Techniques d'adaptation		- Filtrage, transcodage et mixage.
Types de contenus adaptés		- Flux continus ou données discrètes (images)

Tableau 3 : Fiche descriptive de APPAT

4.3 ISIS

ISIS pour ***I**ntelligent **S**calability for **I**nteroperable **S**ervices* est un projet *IST* débuté en Septembre 2002 et terminé en Février 2004 [ISIS02].

Le but d'ISIS était de concevoir, d'implémenter et de valider un cadre de travail (*framework*) multimédia qui permet à du contenu audio-visuel d'être créé une fois et adapté suivant une série de scénarii, prenant en compte les caractéristiques de transport, les capacités du terminal final et les préférences de l'utilisateur final. Les objectifs de ISIS étaient alors de développer des formats de représentation de contenu innovants en insistant sur les techniques de gestion du contenu scalable et de concevoir un cadre de travail de customisation et de personnalisation capable de traiter des méta-données et des formats complexes.

La figure qui suit représente une chaîne de production de bout en bout, de contenu multimédia [GCK04]:

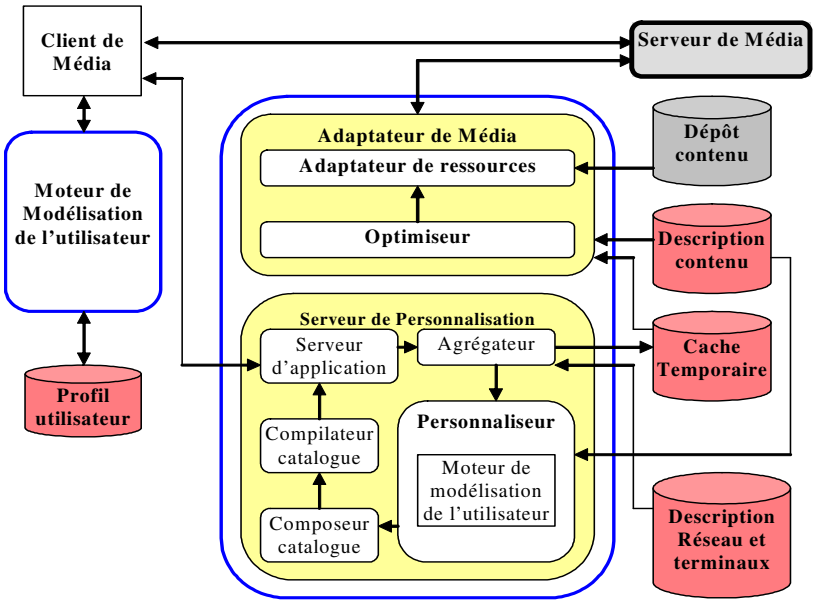

Figure 9: L'architecture de bout en bout de ISIS

4.3.1 Gestion du contexte dans ISIS

La gestion du contexte dans ISIS conduit à la construction d'un catalogue grâce auquel le client choisit le service. Le but de ce processus est la génération de la page d'accueil personnelle dans laquelle figure seul le contenu disponible (de la base de données de description du contenu) et demandé par cet utilisateur spécifique.

Les technologies MPEG-4 et MPEG-21 sont au cœur de ISIS : les flux sont codés en MPEG-4 et que le contexte est décrit grâce aux outils de description MPEG-21.

Une phase de négociation entre l'utilisateur et le serveur est nécessaire pour présenter à cet utilisateur les services demandés qui peuvent être exécutés suivant son profil et son environnement d'exécution.

Le *moteur de modélisation de l'utilisateur* côté client calcule les préférences de l'utilisateur en utilisant des techniques variées telles que l'approche stéréotypée, l'information explicite ou l'apprentissage dynamique.

Le *moteur de modélisation de l'utilisateur* côté serveur gère les profils client légers et suggère pour cela le contenu préféré en s'aidant des techniques spécifiques telles que le filtrage collaboratif. Ce module n'a pas été implémenté dans le projet ISIS mais un outil réalisant les mêmes fonctions a été utilisé. Cependant, la gestion du contexte dans ISIS se fait de manière distribuée : une partie chez le client, une autre chez le serveur.

4.3.2 Gestion de la prise de décision dans ISIS

Deux modules s'occupent de gérer la prise de décision dans ISIS : Le *personnaliseur* et l'*optimiseur.*

Le *personnaliseur* recommande les éléments préférés d'un utilisateur donné, le but étant d'avoir une sélection de contenus correspondants au profil utilisateur. Ce module prend en entrée les méta-données décrivant le contenu ainsi que les préférences utilisateur provenant à la fois du moteur de modélisation côté client et côté serveur.

L'*optimiseur* analyse les contraintes utilisateur et essaye de les faire correspondre aux différentes options d'adaptation du contenu scalable. L'*adaptateur de ressources* est informé de l'adaptation choisie.

4.3.3 Gestion de l'adaptation du service dans ISIS

L'adaptateur de ressources applique l'adaptation choisie et met séquentiellement, sous forme de pipeline, les données du flux adapté au *Serveur de Streaming* pour la packétisation et le streaming au niveau du client.

Le tableau 4 résume les caractéristiques de la plate-forme ISIS.

Fonctionnalités de base	**Gestion du contexte**	- moteur de modélisation de l'utilisateur côté client ou coté serveur : MPEG-21
	Gestion des contenus multimédia	- Description du contenu avec MPEG-4 et MPEG-21
	Gestion de la prise de décision	- Au niveau du serveur : personnaliseur et optimiseur
	Gestion de l'adaptation et localité de l'adaptation	- Adaptateur de ressources
	Gestion des changements dans le contexte	NON
Techniques d'adaptation		- Transmodage : -- Texte vers image -- Graphics2D vers vidéo -- Graphics2D vers image -- Vidéo vers Graphics2D (présentation de slides)
Types de contenus adaptés		- Vidéo MPEG-4

Tableau 4 : Fiche descriptive de ISIS

4.4 UMA ou l'accès universel au multimédia

UMA pour **U**niversal **M**ultimedia **A**ccess est un concept qui émerge de deux laboratoires : l'institut de recherche en électronique et en télécommunications (ETRI) en Corée et le laboratoire DVMM (Digital Video / MultiMedia) de l'université de Colombie, New York. Il a débuté en Juin 2000. Plusieurs projets ont repris ce concept et l'ont implémenté dans leur architecture de fourniture de services multimédia adaptables. Les travaux auxquels nous allons nous intéresser sont ceux relatifs à la thèse [Mag02] qui a été soutenue en Juin 2002 à l'université technique de Lisbonne. D'autres articles [MPe04] reprenant les mêmes idées que celles de cette thèse nous servent également de support.

Le but dans ce projet est aussi de fournir dynamiquement différentes présentations du média afin de s'adapter aux terminaux, au réseau et aux préférences de l'utilisateur.

UMA repose largement sur les deux standards MPEG-7 et MPEG-21 décrits dans les sections 6 et 7 [MKP02, DMa02] [MPEG21]. La Figure 10 montre les différentes entités que comprend l'architecture UMA ainsi que les interactions entre elles :

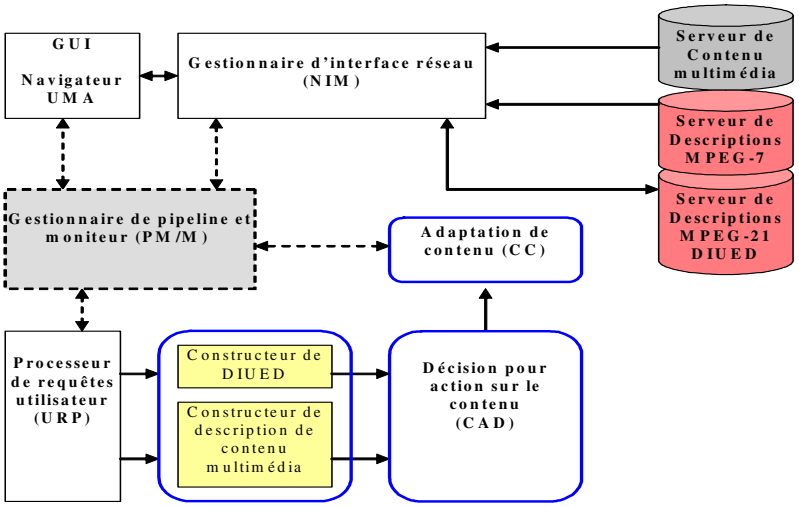

Figure 10: Architecture de la plate-forme UMA

Certaines entités ne contribuent pas directement à la boucle d'adaptation mais plutôt à la communication entre ces différentes entités.

Un élément central appelé *moniteur/gestionnaire de pipeline (PM/M : Pipeline Manager/Monitor)* est informé de tous les messages qui transitent d'un module à un autre et les met à la disposition de l'utilisateur à travers l'*interface utilisateur (GUI)* pour un meilleur contrôle humain. Il permet ainsi de garder une trace de toutes les requêtes envoyées et leur état et de tous les modules. Il a pour tâche de démarrer/créer de nouveaux modules, d'arrêter les modules et de gérer les tables de cache.

Le *moniteur/gestionnaire de pipeline* est la seule entité autorisée à écrire dans les tables de cache qui concernent deux entités. D'une part, la table des utilisateurs permet de les identifier par leurs adresses IP et contient des liens vers les descriptions de leur environnement naturel. Enfin, la table des contenus

multimédia permet, pour chaque entrée, de référencer le contenu, ses descriptions et ses variations existantes.

Le *gestionnaire d'interface réseau (NIM : Network Interface Manager)* a pour but d'intercepter des requêtes de l'utilisateur, ou tout autre information que l'utilisateur souhaite transmettre à la plate-forme UMA, de retirer un contenu ou une description demandée par un module tiers et correspondant à une URL donnée, et de diffuser en retour vers l'utilisateur le contenu adapté.

Le *processeur de requêtes utilisateur (URP)* quant à lui demande au NIM de réaliser les opérations d'extraction des descriptions des différents serveurs, et envoie ces descriptions à parser aux modules conçus pour le parsing. Il est aussi responsable d'utiliser la technique des tables de cache pour une meilleure performance. Il vérifie ainsi si des descriptions sont référencées dans ces tables.

4.4.1 Gestion du contexte et des contenus dans UMA

Plusieurs entités de l'architecture UMA contribuent à collecter les informations de l'environnement de l'utilisateur. UMA se base sur les descriptions MPEG-21 pour décrire les informations relatives à l'environnement utilisateur telles que ses préférences, les caractéristiques de son terminal ainsi que de son réseau d'accès et les paramètres de son environnement naturel. UMA repose sur MPEG-7 pour décrire le contenu de ses données multimédia.

La gestion du contexte peut se faire de manière locale à l'utilisateur grâce à un module appelé *navigateur UMA* qui inclut un serveur Web. L'utilisateur a alors le moyen d'accéder au contenu et la possibilité de gérer les descriptions de son environnement d'usage. Ces descriptions sont envoyées soit directement à la plate-forme UMA (au NIM) soit au serveur DIUED MPEG-21 pour des mises à jour.

Un *serveur DIUED (Digital Item User Environment Description) MPEG-21* s'occupe de stocker les descriptions de l'environnement de l'utilisateur.

Un *serveur de description MPEG-7* stocke les descriptions MPEG-7 relatives au flux, chaque description correspond à une URL.

Les descriptions de l'environnement d'usage de l'utilisateur et du contenu multimédia sont demandées par le URP qui utilise le NIM pour les retirer. Ce dernier va les chercher sur le serveur DIUED et sur le serveur de description MPEG-7.

Les différentes descriptions sont envoyées aux modules de parsing pour récupération.

La plate-forme UMA correspond à l'application qui implémente l'adaptation du contenu. Cette plate-forme joue le rôle du serveur d'adaptation de contenu. Elle comprend la gestion de la prise de décision et celle de l'adaptation du service.

4.4.2 Gestion de la prise de décision dans UMA

Nous retrouvons dans le module de *décision pour action sur le contenu (CAD : Content Action Decision)* la notion de gestionnaire de prise de décision. Le CAD fait la correspondance entre la

description du contenu avec celles de l'environnement utilisateur afin de décider des transformations à appliquer au contenu, en tenant compte des processus d'adaptation existants.

Les politiques d'adaptation peuvent être vues comme l'ensemble des actions des algorithmes de prise de décision qui ont été proposés dans cette thèse et qui concerne la vidéo et l'image.

4.4.3 Gestion de l'adaptation du service dans UMA

Le module *adaptation de contenu (CC : Content Customisation)* reçoit les actions à réaliser et fait appel aux modules d'adaptation qui sont responsables d'adapter le contenu.

Les techniques d'adaptation relatives à l'image et à la vidéo relatives à ce projet sont présentées dans [Mag02]. Dans le cas de l'image, les méthodes d'adaptation sont, par exemple, la réduction de la couleur, la réduction de la résolution spatiale ou encore de la conversion de format. Dans le cas de la vidéo, les méthodes d'adaptation telles que la conversion de format ou la sélection d'autres versions sont implémentées et utilisent un décodeur MPEG-1/2 pour accéder aux frames décompressées prêtes à être adaptées. Un encodeur MPEG-1/2 s'occupe de les compresser de nouveau.

Dans [Mag02], l'adaptation peut être réalisée à différents niveaux : au niveau du serveur, du proxy ou du terminal du client. Les deux premiers cas ont été implémentés à l'inverse du dernier cas car les capacités d'un terminal mobile restent limitées pour effectuer les actions d'adaptation.

Le Tableau 5 reprend les caractéristiques de l'architecture UMA.

Fonctionnalités de base	**Gestion du contexte**	- moteur de modélisation de l'utilisateur côté client ou coté serveur : MPEG-21 - serveur DIUED stocke les informations du contexte
	Gestion des contenus multimédia	- serveur de description MPEG-7 stocke les descriptions
	Gestion de la prise de décision	- Au niveau du CAD
	Gestion de l'adaptation et localité de l'adaptation	- Adaptateur de contenu (CC) gère l'adaptation. - Adaptation au niveau du serveur ou du proxy
	Gestion des changements dans le contexte	NON
Techniques d'adaptation		- réduction de la couleur d'une vidéo - réduction de la résolution spatiale d'une vidéo - conversion de format d'une vidéo ou image - sélection d'autres versions des images
Types de contenus adaptés		- Vidéo et image

Tableau 5 : Fiche descriptive de UMA

4.5 WAM et son architecture NAC

WAM pour **W**eb **A**daptation **M**ultimedia est un projet de recherche à l'INRIA Rhône-Alpes entamé en Janvier 2003 [WAM03]. Il succède au projet OPERA qui est un projet INRIA mené dans l'unité de recherche Rhône-Alpes et qui a pour objectif la conception d'un environnement pour le développement et la maintenance de grosses documentations complexes multimédia [OPE02].

Le but de WAM est d'explorer le domaine du multimédia dans le web tout en mettant l'accent sur la transformation et l'adaptation des documents structurés.

Une architecture d'adaptation et de négociation NAC pour **N**egociation and **A**daptation **C**ore a été développée et a pour but de fournir une solution pour délivrer du contenu multimédia adapté aux contraintes du contexte du client dans des environnements hétérogènes.

La Figure 11 décrit les différents composants de l'architecture NAC présentée dans les travaux de recherche de la thèse [Lem04].

Le module *d'adaptation et de négociation ANM (Adaptation and Negociation Module)* est au cœur de l'architecture NAC. Il est en permanence à l'écoute d'une nouvelle connexion d'une application cliente

Le *module de contexte utilisateur UCM (User Context Module)* se trouve côté client et permet une négociation avancée en configurant le proxy et en choisissant le port de négociation du contenu indépendamment de l'application cliente.

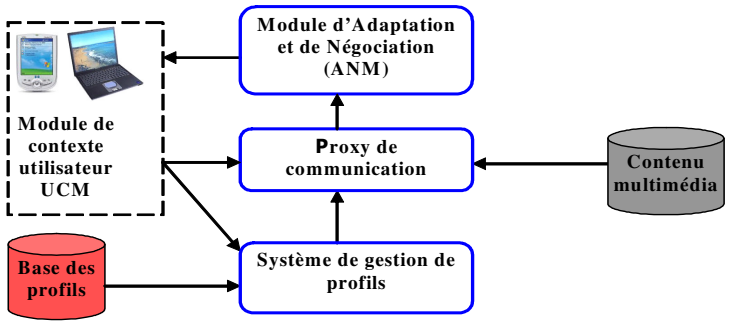

Figure 11: Organisation générale de l'architecture NAC

Le *proxy de communication* reçoit les requêtes du client et les réponses du serveur et assure la communication entre ces deux acteurs. Le proxy gère aussi la négociation avec le module ANM côté client.

Un *protocole de négociation* est utilisé afin d'assurer une négociation avancée**.** Il définit un mode d'interaction entre le UCM et le ANM qui se présente sous forme de requêtes et de réponses.

4.5.1 Gestion du contexte dans NAC

La gestion du contexte dans NAC peut être faite de manière locale à l'utilisateur. Les paramètres pris en compte pour le contexte sont l'environnement du terminal et du réseau utilisé.

Le module UCM fournit le profil utilisateur et donne la possibilité de le changer. Ce profil est envoyé au module ANM une seule fois, tant qu'il ne subit aucun changement.

Le modèle de description utilisé est UPS (Schémas universels pour la description des profils : Universal Profile Schemes). Il est basé sur plusieurs extensions de CC/PP et RDF [CCPP] [RDF].

Les profils peuvent être stockées à trois niveau : au niveau du client, du proxy ou du serveur.

La base de profils agit comme un serveur, les clients étant des modules ANM. Des requêtes RPC sont utilisées pour interroger les profils du dépôt. Les profils sont écrits en UPS. La gestion des profils n'est pas dédiée mais suppose que chaque entité (client, serveur ou proxy) peut à tout moment effectuer ces tâches

Le *système de gestion des profils* s'occupe d'analyser et de gérer les descriptions du contexte. L'architecture NAC offre des services qui peuvent être exploités par le *système de gestion des profils*, et qui permettent d'extraire des profils, des parties des profils ou les caractéristiques élémentaires d'un profil donné. Des moyens d'interaction optimisés avec les dépôts de profils existent.

L'extraction des informations appropriées peut être statique ou dynamique. L'extraction statique consiste en une interrogation paramétrée de la base des profils tandis que l'extraction dynamique fait appel à des méthodes déjà disponibles qui permettent de calculer les valeurs de certaines caractéristiques de l'environnement qui sont généralement dynamiques telles que la bande passante courante.

4.5.2 Gestion de la prise de décision et d'adaptation du service dans NAC

Les fonctionnalités de prise de décision et d'adaptation du document sont confondues et réalisées par la même entité qui est le module d'adaptation et de négociation (ANM) et au même moment.

L'interface du module d'adaptation et de négociation comporte un menu qui permet de configurer les ports utilisés pour accéder au contenu du réseau (le port de communication) et le port réservé pour le protocole de négociation utilisé par le module ANM et le module de contexte utilisateur (le port de négociation). Le menu associe des profils de documents (disponibles côtés ANM) au contenu qui peut être demandé par les applications clientes. La déclaration des profils de documents décrivant le contenu et la disponibilité des versions associées aide le processus de négociation à prendre la meilleure décision après la réception de la requête du client. Cela est réalisé grâce à la connaissance des caractéristiques du contenu demandé et des alternatifs qui peuvent le remplacer.

Le module ANM doit fonctionner en permanence afin d'assumer des services d'adaptation et de négociation au sein d'environnements hétérogènes.

Le module AMN adapte alors le contenu en appliquant des méthodes de transformation structurelle telles que l'adaptation logique, spatiale ou temporelle du flux. L'adaptation réalisée est dynamique étant donné qu'elle tient compte de certaines dimensions du contexte telles que l'application cliente utilisée, la taille de l'écran du terminal, etc.

Le Tableau 6 résume les caractéristiques de NAC :

Fonctionnalités de base	**Gestion du contexte**	- UCM stocke le profil et permet de le changer - UPS pour décrire le profil - Système de gestion des profils au niveau du proxy gère et analyse le profil
	Gestion des contenus multimédia	- Descriptions SMIL
	Gestion de la prise de décision, gestion de l'adaptation et localité de l'adaptation	- Le module ANM
	Gestion des changements dans le contexte	NON
Techniques d'adaptation		- Adaptation structurelle des documents SMIL - Réduction d'image et vidéo
Types de contenus adaptés		- Document multimédia (SMIL, documents XML)

Tableau 6 : Fiche descriptive de NAC

4.6 DCAF

DCAF pour Distributed Content Adaptation Framework est une architecture orientée service qui a été développée au sein du laboratoire LIRIS (Laboratoire d'InfoRmatique en Image et Systèmes d'information) à Lyon, en France [GBP05].

Le but de DCAF est de fournir une architecture d'adaptation de contenu. L'architecture DCAF (Figure 12) comprend six composants principaux décrits ci-dessous :

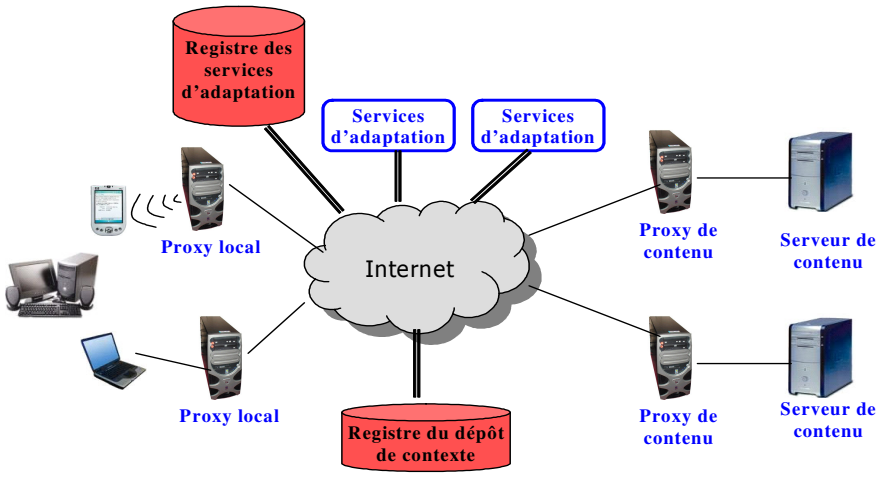

Figure 12 : Architecture DCAF

4.6.1 Gestion du contexte dans DCAF

Le *gestionnaire des profils utilisateurs* (*Context Profile Repository* : CPR) stocke les informations du contexte utilisateur, c'est-à-dire les caractéristiques et les préférences de l'utilisateur, les capacités et les propriétés du terminal et les propriétés du réseau. Les utilisateurs peuvent mettre à jour et modifier leur profil à tout moment. Les informations contextuelles dynamiques telles que la localisation d'un utilisateur ou les paramètres dynamiques du réseau sont déterminées lors de l'exécution des requêtes de données.

4.6.2 Gestion des contenus dans DCAF

Les *serveurs de contenus* représentent des entrepôts standards de données, tels que des sites Web, des bases de données et des serveurs multimédias.

Les *proxies de contenus* (*Content Proxies* : CP) fournissent un accès aux serveurs de contenus, formulent les requêtes des utilisateurs dans un format adéquat aux sources, gèrent et livrent (en réponse aux requêtes des utilisateurs) des descriptions de contenu (méta-données).

4.6.3 Gestion de la prise de décision dans DCAF

Les *proxy locaux* (*Local Proxies* : LP) prennent en charge la récupération et le traitement des profils de contexte. Ils décident du type et du nombre de traitements adaptatifs, découvrent les services d'adaptation, planifient l'exécution des services et les invoquent. Un élément fondamental du proxy local est le module de négociation et d'adaptation de contenu (CNAM). Ce module détermine un schéma d'adaptation optimal et invoque les services d'adaptation ad hoc. Le module CNAM est responsable de la construction du graphe d'adaptation.

4.6.4 Gestion de l'adaptation dans DCAF

Le *répertoire des services d'adaptation* (*Adaptation Service Registry* : ASR) est un annuaire de services d'adaptation semblable à un annuaire UDDI[4]. Il stocke les descriptions fonctionnelles (profils) et non fonctionnelles (ex : coût) des services d'adaptation multimédia et offre des APIs de recherche des services.

Les *services d'adaptation* (*Adaptation Services* : AS) sont des serveurs hébergeant un ou plusieurs service(s) d'adaptation. Les services d'adaptation sont implémentés en tant que services Web et sont développés de manière indépendante de l'architecture DCAF.

Le

Tableau 7 résume les fonctionnalités de l'architecture DCAF :

Fonctionnalités de base	**Gestion du contexte**	- Context Profile Repository : CPR UPS pour gérer le profil utilisateur. - CSCP(Comprehensive Structured Context Profiles) pour décrire le contexte.
	Gestion des contenus multimédia	- Description XML propriétaire pour décrire des méta-données de contenus média
	Gestion de la prise de décision, gestion de l'adaptation	- Au niveau des proxys locaux, plus spécialement le module CNAM.
	localité de l'adaptation	- Au niveau des services d'adaptation.
	Gestion des changements dans le contexte	NON
Techniques d'adaptation		- Traduction de texte - Transmodage du texte vers l'audio - Redimensionnement d'image - Conversion de la couleur vers le gris (greyscale) - Changement du format Image (transcodage)
Types de contenus adaptés		- Médias élémentaires tels que du texte, audio et image.

Tableau 7 : Fiche descriptive de l'architecture DCAF

4.7 Architecture SATO du projet *Ambient*

Le projet « Ambient Networks » est un projet IST (6th FP RTD) dans lequel plusieurs partenaires participent (tels que France Telecom, Ericsson ou encore l'université d'Ottawa) [Ambient]. Il a débuté

4 Universal Description, Discovery and Integration

en Janvier 2006 et prendra fin en Décembre 2007. Il a pour but de réaliser une solution réseau innovatrice, mobile et exploitable en industrie, permettant la composition des réseaux à travers les frontières économiques et technologiques afin d'établir une utilisation efficace des ressources de l'infrastructure et de stimuler de nouveaux développements économiques dans le monde du sans fil.

SATO pour Service-aware Transport Overlay est développé au sein du projet « Ambient Networks » et étend les concepts définis dans l'architecture SMART (pour Smart MultimediA Routing and Transport) dont le but est l'optimisation des services de fourniture de médias en tirant avantage des capacités réseau de traitement des médias. Contrairement à SMART, SATO ne se limite pas aux applications multimédia [SATO06] [MKS06].

Le concept des SSONs, pour Service-Specific Overlay Networks, a été introduit dans les deux plates-formes (SATO et SMART). Un SSON spécifie un réseau de recouvrement pouvant être déployé à la demande pour fournir du contenu média. Dans un SSON, des modules de traitement de média, hébergés au sein de « *Media Ports* » (MP), sont déployés suivant certains critères tels que les besoins utilisateur (terminal, réseau), le prix ou les besoins de sécurité.

[MKS06] présente une approche P2P pour sélectionner et instancier des MP réalisant des traitements sur des médias tels que le transcodage, la mise en cache (caching) ou la synchronisation.

Chaque MP s'exécute sur un nœud du réseau de recouvrement. Plusieurs MP peuvent tourner sur le même Overlay Node.

Un modèle abstrait décrit un Media Port sous forme de triplet (I,P,O) où :

- I représente les formats d'entrée possibles.
- P représente les fonctionnalités de traitement présentes dans le MP
- O représente les formats de sortie que le MP produit.

Afin de rechercher les fonctionnalités de traitement des médias dans le réseau, une base de données distribuée appelée MPDS (pour MediaPort Directory Service) qui maintient une liste de caractéristiques de chaque Media Port telles que les modules de traitements et la charge. Cette liste est obtenue grâce à l'annonce, auprès du MPDS de chaque nœud du réseau de recouvrement, des MP disponibles.

Le MPDS support les opérations :

- « *join* » : un nouveau un nœud du réseau de recouvrement rejoint un MPDS.
- « *leave* » : un un nœud du réseau de recouvrement quitte son MPDS.
- « *register* » : un un nœud du réseau de recouvrement enregistre les informations relatives à un MP dans le MPDS.
- « *deregister* » : retirer depuis le MPDS les informations relatives à un MP.

- « search » : l'entrée de la recherche est la description d'un MP. La valeur de retour est une liste, possiblement vide, des adresses des Overlay Nodes hébergeant les MP correspondant à la requête.

4.8 MAPS

MAPS pour Media Accelerating Peer Services étend les infrastructures P2P existantes avec quelques modules qui permettent de personnaliser les fonctions de recherche et de livraison de documents [LHC03]. MAPS contient un module qui permet l'accès à des contenus adaptés aux caractéristiques du terminal d'un client, à ses capacités de stockage et aux contraintes de bande passante de son réseau d'accès.

4.9 M21

Rong et al. proposent l'architecture M21 qui facilite l'adaptation dynamique des ressources dans un environnement P2P [RBu04]. Le standard MPEG-21 est largement utilisé pour décrire des contenus multimédia, en utilisant, en particulier, les informations décrites dans des descripteurs DIA (Digital Item Adaptation) telles que les descriptions de l'environnement d'usage [MPEG21] [Vet04]. La dynamicité est prise en compte et un large spectre de contenus multimédia (y compris les documents composés) est considéré.

5 Discussion et illustrations du scénario

Cette section a pour but de comparer les architectures étudiées. Nous commençons par rappeler l'énoncé de « Julie and her PDA » présenté au chapitre 1 afin de mieux déterminer ce que permet de réaliser chacune des architectures présentées plus haut. Ce scénario motive l'intérêt d'un système d'adaptation entre le consommateur et le fournisseur de contenu facilitant l'accès et la consommation du contenu multimédia, et ce à des utilisateurs différents possédant des contextes différents.

5.1 Rappel du scénario « Suzy and her PDA »

Suzy est anglaise. Elle se connecte souvent aux forums de discussion, spécialement ceux qui parlent de musique. Compte tenu de son travail, Suzy est souvent en déplacement et se connecte souvent depuis un assistant personnel (PDA). L'abonnement de son fournisseur d'accès Internet/Téléphonie lui permet de se connecter à tout moment, via différents réseaux, suivant l'endroit où elle se trouve : Wifi-ADSL depuis chez elle, UMTS dans la rue, le train ou la voiture. Suzy souhaite utiliser son PDA avec une configuration minime afin d'économiser la batterie de son PDA. Ainsi Suzy ne souhaite récupérer que les documents contenant des images, des bandes sons ou du texte.

En quittant son bureau, Suzy veut récupérer un document multimédia relatif à sa chanteuse préférée. Il est composé d'un contenu audio-vidéo du dernier clip de « Shakira », d'un texte sur sa bibliographie, écrit en français, et de l'image de la pochette de son dernier album. Ce document multimédia doit être

adapté aux préférences de Suzy (langue parlée et restriction sur les contenus vidéo) et aux capacités de son PDA (taille réduite de l'écran).

Certaines fonctions d'adaptation telles que la traduction de texte du français vers l'anglais, la conversion du format (mimetype) de l'image et le transmodage[5] de la vidéo vers l'audio peuvent être exécutées en parallèle par des adaptateurs correspondants, éventuellement offerts par des nœuds dans le réseau. La réduction de l'image ne se fera qu'après la conversion de son format.

Les médias adaptés doivent être re-synchronisés et réassemblés pour former un document multimédia composé adapté aux préférences de Suzy ainsi qu'aux capacités de son contexte.

Si le pair réalisant la traduction du texte disparaît, ou que Suzy décide de s'affranchir de l'audio, le système qui gère l'adaptation doit s'adapter à ces changements.

5.2 Illustration du scénario

Ce scénario concerne des utilisateurs dans des circonstances inédites et possèdent des contextes différents. Ils ont un point commun : les contenus considérés sont des documents multimédia composés de plusieurs médias élémentaires (vidéo, image, texte ou audio).

Mis à part NAC et DCAF, aucune des autres architectures ne manipule des documents multimédia composés, qui nécessitent un effort supplémentaire d'analyse des descriptions du document composé et sa reconstruction en tenant compte de la synchronisation temporelle et/ou d'organisation spatiale des médias élémentaires composant ce document.

Les caractéristiques spécifiques de l'utilisateur (par exemple le fait de ne pas vouloir télécharger de la vidéo) ne sont pas prises en compte par les outils de descriptions du contexte utilisés dans les architectures étudiées.

D'autres caractéristiques spécifiques, qui ne sont pas illustrées par le scénario, tels que le handicap, peuvent être prises en compte par des architectures qui utilisent MPEG-21 comme support de description du contexte utilisateur. Néanmoins, les architectures utilisant MPEG-21 (par exemple UMA ou M21) ne proposent pas de politiques d'adaptation ni d'algorithme de prise de décision relatifs à l'handicap.

5.3 Discussion

Les éléments présentés dans la section 3 nous permettent de comparer chacune des architectures étudiées et de discuter leurs avantages et leurs inconvénients. Les fiches descriptives présentées pour les architectures nous ont permis de réaliser une première distinction entre ces architectures. Nous comparons dans ce qui suit les architectures suivant d'autres critères : le modèle architectural, la capacité à passer à l'échelle, l'extensibilité, la distribution et la description des ressources d'adaptation et la capacité à décrire des caractéristiques contextuelles spécifiques.

5. Le changement de la modalité d'un média. Par exemple, passer d'une vidéo à une image.

5.3.1 Modèle architectural

Les architectures étudiées ne suivent pas toutes le même modèle architectural et ne réalisent pas toutes l'adaptation au niveau de la même entité :

- ISIS suit le modèle architectural client / serveur où l'adaptation est réalisée au niveau du serveur.
- ADMITS, APPAT, UMA et NAC, DCAF reposent sur le modèle architectural client / intermédiaire(s) / serveur où l'adaptation est réalisée au niveau des intermédiaires.
- M21, MAPS et SATO suivent le modèle architectural P2P où l'adaptation est partiellement réalisée au niveau des pairs du réseau.

Une discussion a été réalisée sur les avantages et les inconvénients de chaque modèle architectural et sur les conséquences de chaque solution concernant la localité de l'adaptation dans la section 3.5.4.

5.3.2 Passage à l'échelle et l'extensibilité

Les architectures ISIS et NAC, UMA ne se prêtent pas à l'extensibilité car les modules réalisant les différentes fonctionnalités sont décidés statiquement et aucun choix dynamique des modules n'est permis.

D'autres architectures telles que M21 et MAPS ne passent également pas à l'échelle car elles sont basées sur plate-forme P2P JXTA qui ne le permet pas.

5.3.3 Distribution et la description des ressources d'adaptation

ADMITS, APPAT et SATO sont les seules architectures à introduire la notion de description des ressources d'adaptation (Data Manager dans ADMITS, proxies d'adaptation dans APPAT et Overlay Node dans SATO) en vue d'une recherche par critère dans un annuaire ou une base de données distribuées. Cette fonctionnalité permet le choix adéquat des adaptateurs.

5.3.4 Description des caractéristiques contextuelles spécifiques

MPEG-21 est le standard de fait largement utilisé dans ADMITS, ISIS et UMA. Ce point est à la fois positif et négatif. Le plus grand avantage de MPEG-21 est qu'il offre de nombreux outils pour décrire, entre autres, l'environnement d'usage de l'utilisateur. MPEG-21 et CC/PP (utilisée par ADMITS et APPAT) ne permettent pas cependant de décrire certaines préférences spécifiques telles que le refus d'avoir un contenu vidéo (cf. scénario « Suzy and her PDA »). Une description plus détaillée de ce standard sera réalisée dans la section 6.2). Nous allons néanmoins décrire ces deux outils de description du contexte dans la section qui suit.

6 Outils de description de contexte

Il existe plusieurs standards dont le but est d'offrir des mécanismes pour décrire le contexte de l'utilisateur. Nous décrivons dans cette section CC/PP et MPEG-21 car ils permettent de décrire une

grande partie des informations contextuelles d'un utilisateur et qu'ils sont largement utilisés dans les architectures étudiées [CCPP] [MPEG21]. Notons également que nous nous inspirons MPEG-21 et CC/PP afin de décrire les informations contextuelles qui nous intéressent.

6.1 CC/PP

CC/PP pour Composite Capability/Preference Profiles est né des efforts de standardisation du W3C (World Wide Web Consortium). Le but de CC/PP est d'offrir les moyens pour un client d'exprimer les capacités logicielles et matérielles de son terminal ainsi que ses préférences (profil user agent). CC/PP est ainsi un standard de description du contexte pour la négociation et l'adaptation de contenu. Malgré le fait que CC/PP soit pensé au départ pour les terminaux mobiles, il reste cependant assez général pour décrire n'importe quel environnement d'utilisateur (appelé *user profile* dans la terminologie CC/PP). La figure qui suit montre que CC/PP repose sur les technologies XML/RDF.

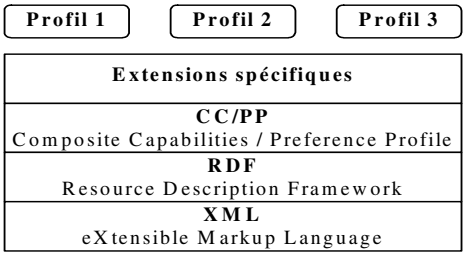

Figure 13 : CC/PP est une application RDF qui repose sur XML

Basé sur XML, RDF a été conçu pour fournir un langage de description de méta-données [RDF]. CC/PP utilise RDF, qui grâce aux schémas, lui permet de développer et implémenter de manière indépendante un vocabulaire, et offre des moyens de réutilisation et d'extensibilité des méta-données.

Le cadre de travail CC/PP définit une structure et un vocabulaire. Une structure définit l'organisation du profil sous forme d'arbre à 2 niveaux de hiérarchie. Cette structure inclut les composants (components) et les attributs (attributes). Les composants possibles sont la plate-forme matérielle (*TerminalHardware*), la plate-forme logicielle (*TerminalSoftware*), l'application qui récupère et présente le contenu à l'utilisateur tel que le navigateur (*TerminalBrowser*). Chaque composant est un sous-arbre dont les branches représentent les capacités et les préférences de ce composant.

RDF permet la modélisation d'un grand nombre de structures de données. Un exemple d'un profil CC/PP est illustré par la Figure 14 :

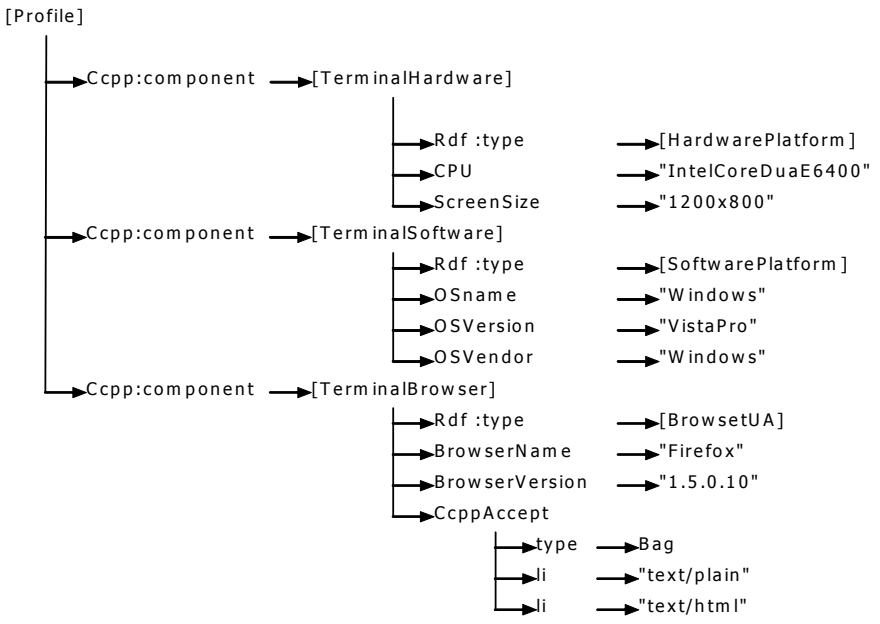

Figure 14 : Organisation hiérarchique d'un profil CC/PP

UAProf pour User Agent Profile (UAProf), une extension de CC/PP, est à l'initiative du forum *Wireless Application Protocol* (WAP) dont l'objectif est de créer des spécifications afin de développer des protocoles se prêtant à la mobilité [UAPROF] [WAP]. Les éléments de descriptions de UAProf incluent les caractéristiques logicielles et matérielles du terminal ainsi que les informations sur le réseau d'accès. UAProf est spécifique aux besoins des terminaux mobiles à ressources limitées, par conséquent, il ne procure pas de descriptions relatives aux préférences de l'utilisateur.

6.2 MPEG-21

MPEG-21 est une initiative du groupe MPEG -Moving Picture Experts Group- (ISO/IEC JTC1 SC29 WG11) formellement appelé cadre de travail multimédia (*Multimedia Framework*) [BVH03] [Vet04]. MPEG-21 ou ISO/IEC 21000, plus récent des standards MPEG, offre plusieurs éléments pour construire un cadre de travail ouvert pour la fourniture et la consommation du contenu multimédia. MPEG-21 décrit ces éléments en fournissant une « vue globale ». MPEG-21 a pour but de permettre l'utilisation transparente et massive des ressources multimédia à travers une multitude de types de réseaux et de terminaux utilisés par différentes communautés.

Le cadre de travail multimédia MPEG-21 repose sur deux concepts essentiels : le concept d'élément numérique (*Digital Item*), le « quoi », et le concept d'utilisateurs (*Users*) interagissant avec cet élément numérique, le « qui ».

Un utilisateur (*User*) est toute entité qui interagit au sein de l'environnement MPEG-21 ou qui manipule un élément numérique. Un utilisateur peut être un créateur de contenu, un distributeur de contenu ou un consommateur de contenu.

L'élément numérique ou l'élément digital (*DI* pour *Digital Item*) est le concept architectural de base de distribution et de transaction entre les utilisateurs dans le cadre de travail MPEG-21. Il s'agit d'une combinaison de ressources (par ex : pistes audio, vidéo ou image) et de méta-données (par ex des descripteurs MPEG-7).

MPEG-21 procure un cadre de travail pour les interactions des utilisateurs à travers l'élément numérique. MPEG-21 a identifié 16 parties (la partie 13 n'étant pas attribuée) nécessaires pour supporter une chaîne de fourniture de contenus multimédia. Les spécifications qui nous intéressent le plus, et surtout qui répondent aux besoins de description du contexte d'un usager sont les spécifications relatives à la déclaration et à l'adaptation d'éléments numériques.

- La partie déclaration d'un élément numérique (Partie 2: *DID* pour *Digital Item Declaration*) fournit une abstraction uniforme et flexible et un schéma inter-opérable pour déclarer les éléments numériques et pouvoir ainsi former un modèle utile pour les définir. La Figure 15 est un exemple montrant les éléments importants dans ce modèle et la relation qui les lie. Un contenaire (container) peut être vu comme une structure regroupant plusieurs items et/ou contenaires. Ce groupage peut être par exemple utilisé pour former des packages logiques facilitant le transport ou l'échange de ces items. De la même façon, un *item* est un groupement d'*items* et/ou de composants liés à des descripteurs. Les *items* peuvent contenir des balises *choice* pouvant être personnalisées ou configurées ou des balises conditionnelles. Un composant (*component*) lie une ressource à un ensemble de descripteurs. Ces descripteurs sont des informations relatives à tout ou partie d'une instance de ressource spécifique. Une ressource (*resourc*e) est un média identifiable de manière individuelle tel qu'une vidéo ou un clip audio. Toutes les ressources doivent être localisable via une adresse non ambiguë.

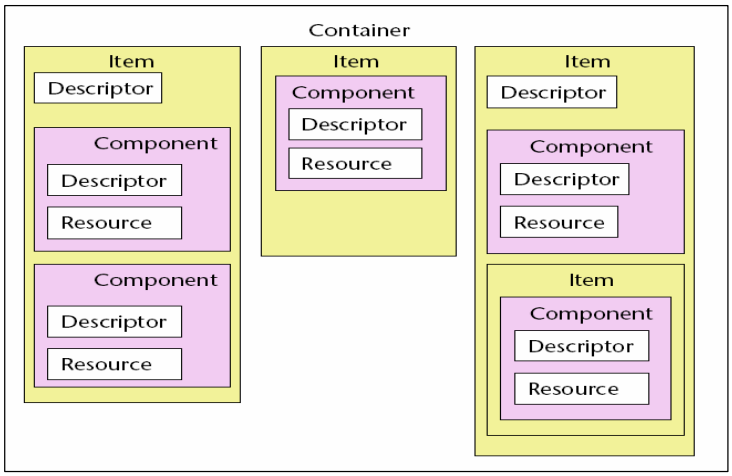

Figure 15: Exemple d'un modèle de déclaration d'un élément numérique ([BVH03])

- La partie adaptation des éléments digitaux (Partie 7 : *DIA* pour *Digital Item Adaptation*) définit les outils de description de l'environnement d'utilisation et du format de contenu des

équipements qui influent sur l'accès transparent au contenu multimédia comme par exemple les terminaux, les réseaux, les utilisateurs et leur environnement naturel.

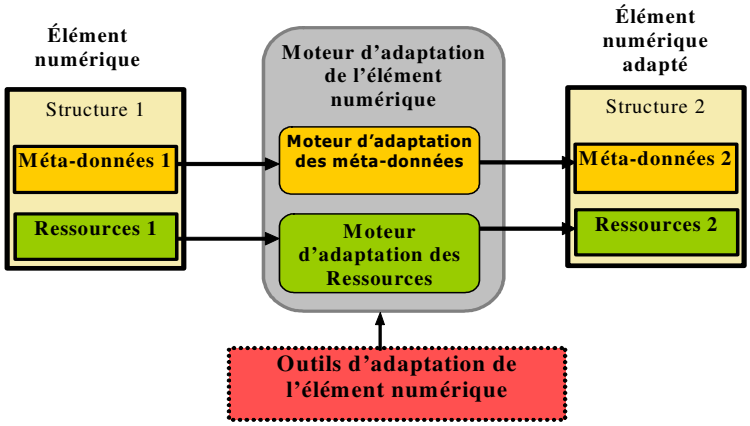

Figure 16: Adaptation d'un élément numérique (DIA)

Les outils d'adaptation de l'élément numérique se composent de huit parties comme l'illustre la Figure 17. Nous mettons l'accent sur les outils de description de l'environnement d'utilisation.

Les outils de description du contexte, appelés outils de description de l'environnement d'usage dans la terminologie de MPEG-21, constituent la partie qui nous peut le mieux à nos besoins de représentation et description du contexte de l'utilisateur tel que présenté dans la section 3.1. En effet, ces outils fournissent une information descriptive concernant les diverses propriétés de l'environnement d'utilisation, à savoir les préférences de l'utilisateur, les capacités de son terminal et les caractéristiques du réseau et de l'environnement naturel.

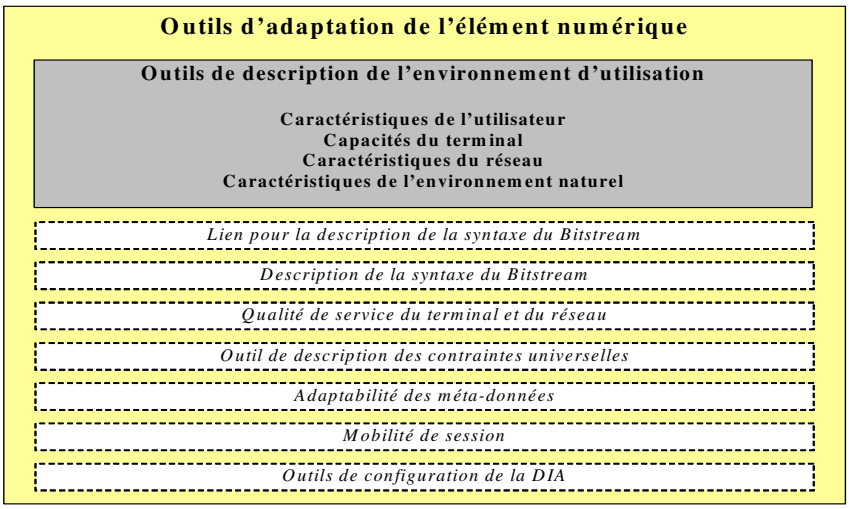

Figure 17: Aperçu et organisation d'outils d'adaptation d'élément digital

Nous illustrons dans ce qui suit un exemple de description MPEG-21 d'un contexte utilisateur. Notons que la description des éléments du contexte, qui est considérée comme un élément numérique en MPEG-21, est incluse dans la déclaration de cet l'élément numérique.

```
<did:DIDL>
<did:Item>
<did:Descriptor>
<did:Statement mimeType="text/xml">
 <DIA>
  <Description xsi:type="UsageEnvironmentType">
   <UsageEnvironmentProperty xsi:type="UsersType">
    <User>
    <UserCharacteristic xsi:type="UsagePreferencesType">
    <UsagePreferences>
          < mpeg7:FilteringAndSearchPreferences>
                    < mpeg7:ClassificationPreferences >
                              < mpeg7:Language>fr</ mpeg7:Language>
                    </ mpeg7:ClassificationPreferences>

                   < mpeg7:SourcePreferences>
                              < mpeg7:MediaFormat>
                              < mpeg7:Content>image</ mpeg7:Content>
                              < mpeg7:FileFormat href="urn:mpeg:MPEG7FileFormatCS:11">
                              < mpeg7:Name>bmp</ mpeg7:Name>
                              </ mpeg7:FileFormat>
                              </ mpeg7:MediaFormat>
                            < mpeg7:MediaFormat>
                              < mpeg7:Content>video</ mpeg7:Content>
                              < mpeg7:FileFormat href="urn:mpeg:MPEG7FileFormatCS:16">
                              < mpeg7:Name>Quick Time</ mpeg7:Name>
                              </ mpeg7:FileFormat>
                              </ mpeg7:MediaFormat>
                           < mpeg7:MediaFormat>
                              < mpeg7:Content>audio</ mpeg7:Content>
                              < mpeg7:FileFormat href="urn:mpeg:MPEG7FileFormatCS:4">
                              < mpeg7:Name>mp3</ mpeg7:Name>
                              </ mpeg7:FileFormat>
                              </ mpeg7:MediaFormat>
                    </ mpeg7:SourcePreferences>
          </mpeg7:FilteringAndSearchPreferences>
   </UsagePreferences>
   </UserCharacteristic>
   </User>
   </UsageEnvironmentProperty>
   <UsageEnvironmentProperty xsi:type="TerminalsType">
   <Terminal>
   <TerminalCapability xsi:type="DisplaysType">
   <Display>
   <DisplayCapability xsi:type="DisplayCapabilityType">
   <Mode>
   <Resolution horizontal="1600" vertical="1200"/>
   </Mode>
   </DisplayCapability>
   </Display>
   </TerminalCapability>
   <TerminalCapability xsi:type="CodecCapabilitiesType">
          <Decoding xsi:type="AudioCapabilitiesType">
          <Format href="urn:mpeg:mpeg7:cs:MPEG7SystemCS:2001:4.4">
          <mpeg7:Name>RealMedia</mpeg7:Name>
          </Format>
          </Decoding>
   </TerminalCapability>
   </Terminal>
   </UsageEnvironmentProperty>
  </Description>
 </DIA>
```

```
</did:Statement>
</did:Descriptor>
</did:Item>
</did:DIDL>
```

Figure 18 : Exemple de description des caractéristiques d'un terminal et des préférences d'un utilisateur en MPEG-21

La présence de balises *mpeg7* est due au fait que MPEG-21 réutilise les schémas de classification introduit par les spécifications de MPEG-7 que nous abordons dans la section 7.1.

7 Outils de description de contenus multimédia

Parmi les standards existants relatifs au multimédia et à la description, nous distinguons entre ceux qui permettent de décrire un contenu multimédia ou mono-média et les modèles et langages qui permettent de décrire une scène multimédia. MPEG-7 [MKP02] est un standard appartenant à la première famille, tandis que Dublin Core [DUBLIN], SMIL [SMIL], MPEG-21 ou encore ZγX [ZYX] appartiennent à la seconde. Nous choisissons de décrire MPEG-7, Dublin Core et SMIL car ils répondent plus ou moins à nos besoins en matière de description d'un média ou d'un document multimédia composés.

7.1 MPEG-7

Le nom formel de MPEG-7 est « interface de description de contenu multimédia » [MKP02, DMa02]. Tout comme MPEG-21, MPEG-7 est une initiative du groupe MPEG (ISO/IEC JTC1 SC29 WG11). MPEG-7 fournit un ensemble d'outils pour décrire un contenu multimédia facilitant la recherche, la navigation et la consommation de ce contenu.

Ces outils MPEG-7 se traduisent par :

- Descriptor (D): un descripteur décrit les informations multimédia, les attributs ou les groupes d'attributs d'un contenu multimédia. Un descripteur définit la syntaxe et la sémantique d'une représentation d'un média (méta-données). Des descripteurs MPEG-7 peuvent par exemple décrire la couleur, la texture, la forme, le mouvement ou le son d'un média.

- Descriptor Scheme (DS): un schéma de description représente la structure et la sémantique des relations entre les descripteurs et les schémas de description. Un exemple d'un schéma de description MPEG-7 est un schéma de description d'un segment d'une vidéo qui peut être décrit avec les descripteurs cités plus haut.

- Description Definition Language (DDL) : ce langage permet de créer de nouveaux schémas de description et même des descripteurs. Il permet aussi la modification et l'extension de schémas de description existants.

- Outils système: ces outils sont des supports pour le multiplexage des descriptions, la synchronisation des descriptions avec le contenu, les mécanismes de fourniture du contenu, et les représentations codées (format textuel et binaire) pour un stockage et une transmissions

efficaces et pour la gestion et la protection de la propriété intellectuelle des descriptions MPEG-7.

Une description MPEG-7 est donc un schéma de description (structure) et un ensemble de valeurs de descripteurs (instanciations) décrivant les données.

La Figure 19 est un exemple de la description MPEG-7 du « logo MPEG-7 ».

Cette description contient le schéma de description (Descriptor Schema) « Creation » qui décrit par qui, quand et où ce logo a été créé et le schéma de description « RelatedMaterial » qui apporte plus d'informations relatives à cette image. Ces deux schémas de description sont inclus au sein du schéma de description CreationInformation.

Plusieurs outils (decriptions, descripteurs et schémas de description) ont été développés dans le cadre de MPEG-7 afin de décrire :

- Le contenu multimédia
- L'information générée par l'auteur relative au processus de création / production d'un contenu multimédia (comme le titre, le créateur, et le but de la création)
- L'information relative au processus de l'usage du contenu multimédia.
- La structure temporelle et spatiale du contenu multimédia.
- La sémantique du contenu multimédia qui peut être utilisée pour décrire les mondes narratifs représentés dans ou reliés au contenu multimédia en décrivant les objets, les événements, les concepts, les places et le temps dans ces mondes narratifs.
- Les sommaires, résumés, vues du contenu, les partitions, les décompositions temporelles, spatiales ou fréquentielle d'une image, d'une vidéo, ou d'un signal audio, et les relations entre les différentes variantes des programmes multimédia. Ces outils permettent la navigation et l'accès au contenu multimédia.
- Les préférences de l'utilisateur ainsi que l'historique d'usage des utilisateurs d'un contenu multimédia, rendant l'utilisateur interactif et permettant la personnalisation de l'accès et la consommation du contenu.

```
<Mpeg7 xmlns="http://www.mpeg7.org/2001/MPEG-7_Schema" xml:lang="en"
type="complete">
<ContentDescription xsi:type="ContentEntityType">
<MultimediaContent xsi:type="ImageType">
<Image>
       <MediaLocator>
              <MediaUri>
              http://www.tilab.org/mpeg/mpeg_logo-anim_1.gif
              </MediaUri>
       </MediaLocator>
       <CreationInformation">
              <Creation>
                    <Title xml:lang="en">The animated MPEG Logo</Title>
                    <Creator>
                          <Role href="urn:mpeg:mpeg7:cs:RoleCS:AUTHOR">
                                <Name xml:lang="en">Author</Name>
                          </Role>
                          <Agent xsi:type="OrganizationType">
                                <Name>MPEG</Name>
                          </Agent>
                    </Creator>
              </Creation>
              <RelatedMaterial>
                    <MediaLocator>
                          <MediaUri>http://www.tilab.com/mpeg/</MediaUri>
                    </MediaLocator>
              </RelatedMaterial>
       </CreationInformation>
</Image>
</MultimediaContent>
</ContentDescription>
</Mpeg7>
```

Figure 19: Exemple d'une description MPEG-7

7.2 Dublin Core

Le NCSA (National Center for Supercomputing Applications) et l' OCLC (Online Computer Library Center), réunis en 1995 au siège de l'OCLC à Dublin, Ohio, ont défini un ensemble de méta-données communes à diverses communautés tels que les musées et les librairies : le *Dublin Core Metadata Initiative* (DCMI), abrégé souvent en *Dublin Core* ou en *DC* [DUBLIN]. Dublin Core est un modèle de descriptions de données (méta-données) simple relatif aux ressources informatiques.

Le *Dublin Core* est un ensemble de 15 éléments de méta-données ayant trait:

- Au *Contenu*: Title, Description, Subject, Source, Coverage, Type, Relation
- À la *Propriété intellectuelle*: Creator, Contributor, Publisher, Rights
- À la *Version*: Date, Format, Identifier, Language

Ces éléments sont résumés dans le Tableau 8.

Pour faire référence à un élément du *Dublin Core*, l'OCLC préconise d'utiliser le PURL (Persistent **U**niform **R**esource **L**ocator) défini pour le *Dublin Core* [PURL]. Un PURL est en fait une URL réputée persistante et redirigée vers un service de résolution de noms. Ce système garantit une meilleure stabilité référentielle que les URL classiques. Ainsi, l'élément *Creator* du *Dublin Core* selon la version 1.1, fait référence univoquement à http://purl.org/dc/elements/1.1/creator, redirigé par le système PURL vers :

http://dublincore.org/2003/03/24/dces#creator par exemple.

Dublin Core a été proposé pour faciliter la recherche de ressources peu complexes tels que des documents textuels et ne propose pas encore assez d'extensions pour décrire des informations telles que la résolution spatiale d'une vidéo par exemple.

Nom de l'élément	Identifiant	Définition
titre	Title	Le nom donné à la ressource
Créateur	Creator	L'entité principalement responsable de la création du contenu de la ressource
sujet et mots-clefs	Subject	Le sujet du contenu de la ressource
description	Description	Une description du contenu de la ressource
éditeur	Publisher	L'entité responsable de la diffusion de la ressource, dans sa forme actuelle, tels, un département universitaire, une entreprise.
contributeur	Contributor	Une entité qui a contribué à la création du contenu de la ressource
date	Date	Une date associée avec un événement dans le cycle de vie de la ressource
type	Type	La nature ou le genre du contenu de la ressource
format	Format	La matérialisation physique ou digitale de la ressource
identifiant	Identifier	Une référence non ambiguë à la ressource dans un contexte donné
source	Source	Une référence à une ressource à partir de laquelle la ressource actuelle a été dérivée
langue	Language	La langue du contenu intellectuel de la ressource
relation	Relation	Une référence à une autre ressource qui a un rapport avec cette ressource
couverture	Coverage	La portée ou la couverture spatio-temporelle de la ressource
droits	Rights	Informations sur les droits sur et au sujet de la ressource

Tableau 8 : Éléments de méta-données du Dublin Core, Version 1.1 [DUBLIN]

Les éléments décrits dans le tableau ci-dessus et utiles pour l'adaptation sont par exemple le type, le format, la langue, la couverture et les droits.

7.3 SMIL

SMIL ou Synchronized Multimedia Integration Language [SMIL] est un standard W3C. SMIL permet l'édition simple de présentations interactives audio visuelle. SMIL est utilisé pour les présentations multimédia/média riche qui intègrent le streaming audio et vidéo aux images, aux textes et à tout autre type de média. SMIL est un langage basé sur XML, facile à prendre en main et ressemble à du HTML.

Avec la version 2.1 de SMIL, un auteur peut décrire le comportement temporel d'une présentation multimédia, associer des hyperliens aux objets médias et décrire la disposition spatiale d'une présentation sur l'écran. La spécification de SMIL permet la réutilisation de sa syntaxe et de sa sémantique au sein d'autres langages basés sur XML, plus particulièrement ceux qui ont besoin de représenter des aspects temporels et de synchronisation tels que XHTML [XHTML].

La Figure 20 introduit le contenu de la description XML d'un document SMIL ; la Figure 21 en est sa représentation graphique dans le lecteur RealOne. Le document est composé d'une vidéo, d'une image et d'un texte qui se déroule en parallèle.

```
<smil>
   <head>
      <meta name="noname" content="empty" skip-content="true" />
      <layout type="text/smil-basic-layout">
      <root-layout height="248" width="403" background-color="black" skip-content="true" />

          <region id="region-video" title="video shakira" height="140" width="200"
          background-color="black" left="1" top="1" z-index="1" fit="meet" skip-
content="true" />

          <region id="region-image" title="image shakira" height="156" width="200"
          background-color="black" left="212" top="2" z-index="2" fit="meet" skip-
content="true" />

          <region id="region-text" title="text shakira" height="137" width="198"
background-color="black" left="10" top="150" z-index="3" fit="fill" skip-content="true" />
                </layout>
    </head>

<body>
<par endsync="last" repeat="1">

<video id="videoshak" title="video shakira" author="Zakia" region="region-video"
src="file:///d:/shak.avi" repeat="1" fill="remove" />

<img id="imageshakira" title="image shakira" author="Zakia" region="region-image"
src="file:///d:/shakira.jpg"  repeat="1"
fill="remove" begin="id(videoshak)(begin)" end="id(videoshak)(end)" system-bitrate="0" />

<text id="text-0" title="text shakira" author="Zakia" region="region-text"
src="file:///d:/shakira.txt" dur="indefinite" repeat="1" fill="remove" />
</par>
</body>
</smil>
```

Organisation Spatiale

Synchronisation temporelle

Figure 20 : Exemple du document SMIL shakira.smil

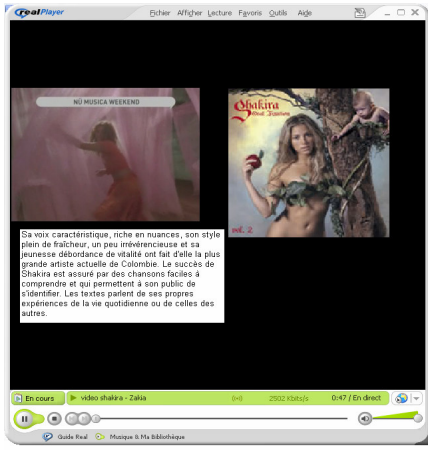

Figure 21 : Représentation avec RealOne du document SMIL shakira.smil

L'élément racine d'un document XML est l'élément *smil.* Cet élément contient deux éléments fils *body* et *head.* L'élément *head* ne contient pas des informations temporelles mais renseigne sur l'auteur et la présentation spatiale. L'élément *body* contient des informations sur les médias et leurs relations temporelles.

Un document SMIL est structuré sous forme de conteneurs temporels (*time containers* en anglais), appelés également éléments composites ou opérateurs. Un opérateur comprend une sémantique temporelle particulière qui permet de définir le placement temporel des objets média. Ces opérateurs sont les éléments *seq, par* et *excl*.

Le conteneur temporel *seq* (séquence) définit une présentation en séquence des ressources médias. Le conteneur temporel *par* (parallèle) permet de jouer les ressources en parallèle. Finalement, l'opérateur *excl* est base sur *par* mais en ajoutant la contrainte que seul un objet enfant soit joué à un moment donné.

8 Conclusion : Constatations et objectifs

Dans ce chapitre, nous comparons les architectures de fourniture de contenus multimédia adaptables existantes par rapport aux fonctionnalités de base qu'elles réalisent (la gestion du contexte, la gestion des contenus multimédia, la gestion de la prise de décision pour l'adaptation et la gestion de l'adaptation distribuée), le modèle architectural adopté, la gestion de la dynamicité et les techniques d'adaptation gérées.

Toutes ces architectures proposent des solutions différentes pour répondre aux besoins fondamentaux de l'adaptation de contenus multimédia. Cependant, nous constatons que certaines besoins ne sont pas ou que partiellement atteints telles que le passage à l'échelle, l'extensibilité, la tolérance aux fautes et aux déconnexions, la prise en compte des préférences particulières (par ex. les modalités préférées ou les lecteurs multimédia présents sur le terminal) et des caractéristiques spécifiques d'utilisateur tel que le handicap, la prise en compte de contenus multimédia riches et composés.

De ce fait, notre objectif principal est la conception d'une architecture d'adaptation de documents multimédia composés intégrant les fonctionnalités de bases requises et permettant également :

- La prise en compte de besoins spécifiques de l'utilisateur tels que la possibilité de choisir quelles modalité accepter.
- L'adaptation de documents multimédia composés de plusieurs médias élémentaires.
- Le passage à l'échelle pour autoriser un grand nombre d'utilisateurs à adapter leur contenus multimédia via notre système d'adaptation.
- La distribution et la parallélisation de l'adaptation profitant ainsi des ressources inutilisées et offertes par des nœuds du réseau.
- L'extensibilité en permettant à des ressources d'adaptation d'être facilement intégrées au système d'adaptation.
- La prise en compte d'adaptateurs spécifiques tels que les traducteurs LSF (Langage des Signes Française).
- La gestion de la dynamicité en gérant la disparition des adaptateurs.

Le chapitre qui suit décrit PAAM (pour *Architecture for the **P**rovision of **A**d**A**ptable **M**ultimedia composed documents*), l'architecture de fourniture d'adaptation que nous proposons.

Chapitre 3

PAAM : Une architecture pour la fourniture de contenus multimédia adaptables

1 Introduction

Dans le chapitre précédent, nous avons présenté plusieurs architectures ayant pour but d'adapter des contenus multimédia au contexte des utilisateurs. La plupart de ces solutions suivent le modèle architectural client / serveur comme c'est le cas pour la plate-forme ISIS, le modèle architectural client / intermédiaire(s) / serveur comme c'est le cas pour les plates-formes UMA, ADMITS, NAC et APPAT ou encore le modèle architectural P2P comme c'est le cas pour les plates-formes M21, MAPS et SATO.

L'architecture d'adaptation que nous proposons, PAAM (pour *Architecture for the **P**rovision of **A**d**A**ptable **M**ultimedia composed documents*), s'attaque aux limitations des projets précédemment décrits, en proposant, en particulier, des mécanismes pour intégrer de nouveaux adaptateurs et pour composer à la demande des adaptations complexes. Certains projets tels que SATO proposent des mécanismes de recherche et de composition d'adaptateurs mais ne gèrent pas la disparition des adaptateurs et ne traitent pas le même type de contenus que notre architecture. En effet, PAAM permet l'adaptation dynamique de documents multimédia composés.

Notre objectif est de mettre en œuvre une adaptation distribuée sur différents nœuds du réseau évitant le schéma classique où l'adaptation est dédiée à la source du contenu ou à un intermédiaire. PAAM est une généralisation du modèle architectural client / intermédiaires / serveur car tous les nœuds peuvent à la fois être consommateur de contenus multimédia (client), producteur de contenus multimédia (serveur) et adaptateurs de contenus multimédia (intermédiaire). PAAM est ainsi assez proche du modèle architectural P2P (voir la section 3.5 du chapitre 2 pour une description des modèles architecturaux). Néanmoins, nous ne classons pas PAAM dans les modèles architecturaux P2P car nous considérons deux autres dimensions qui sont la dimension économique (le modèle économique) ainsi que la dimension technologique (la plate-forme d'implémentation).

1.1 PAAM et la dimension économique

Dans le cadre de la fourniture de services d'adaptation, nous distinguons trois modèles économiques (business model) en considérant les deux métriques suivantes : le coût de ce service d'adaptation et la garantie du bon déroulement de l'adaptation. Ces trois modèles sont :

- Un modèle économique intégrant des fournisseurs d'adaptateurs offrant un accès payant aux adaptateurs. Les fournisseurs d'adaptateurs doivent dans ce cas garantir le bon déroulement de l'adaptation.
- Un modèle économique intégrant des fournisseurs de services isolés offrant un accès gratuit aux adaptateurs. Les fournisseurs d'adaptateurs se portent ainsi volontaires pour exécuter des fonctions d'adaptation en donnant un peu de leurs ressources matérielles et logicielles. Cependant, l'intégrité de l'adaptation offerte n'est pas forcément garantie et peut seulement reposer sur l'appartenance à un réseau de confiance d'adaptateurs.
- Un modèle économique hybride qui autorise la présence de fournisseurs d'adaptateurs aussi bien payants que gratuits, garantissant ou pas le bon déroulement de l'adaptation, appartenant ou pas à un réseau de confiance d'adaptateurs. PAAM repose sur ce modèle économique.

1.2 PAAM et la dimension technologique

Il existe deux technologiques différentes offrant des mécanismes d'annonce, de découverte, de recherche et d'instanciation de ressources : les plates-formes P2P et les services Web.

- Les plates-formes P2P grand public telles que Emule [Emule] donnent accès à un nombre important de ressources. Bien qu'il ne soit pas exclu que ces ressources soient des ressources d'adaptation, ces plates-formes P2P n'offrent pas d'outils pour décrire et composer des ressources d'adaptation. JXTA est une plate-forme plus académique qui pourrait offrir des outils pour annoncer et rechercher des ressources d'adaptation. Elle ne permet cependant pas de passer à l'échelle [JXTA].
- Les services Web : Les adaptateurs ainsi que le système d'adaptation peuvent être vus comme des services Web qui offrent non seulement des outils de descriptions et de recherche des services Web mais également permettent, à travers BPEL, d'offrir des moyens d'orchestration et de composition de services Web [Mat06].

En pratique, PAAM repose sur la technologie des services Web car ils sont plus faciles à prendre en main, largement utilisés et facilement déployables à l'inverse des plates-formes P2P, qui souvent propriétaires, ou qui ne passent pas toujours à l'échelle. Nous verrons dans le chapitre 4 plus de détails sur la technologie des services Web et sur BPEL.

Le reste du chapitre s'articule comme suit : nous commençons par analyser le scénario illustratif présenté dans les chapitres précédents, qui illustre les besoins fonctionnels PAAM. Nous présentons par

la suite l'architecture fonctionnelle de PAAM ; nous détaillons ainsi le gestionnaire de contexte, le gestionnaire de documents composés, le planificateur ainsi que le gestionnaire d'adaptation. Avant de conclure, nous discutons le passage à l'échelle de notre solution étant donné qu'il représente l'un de nos principaux objectifs.

2 Analyse du scénario « Suzy and her PDA »

Le scénario « Suzy and her PDA » a été présenté dans la section 5.1 du chapitre 2. Il nous a servi comme support de comparaison et d'analyse des architectures présentées dans le chapitre précédent. Nous reprenons ce même scénario afin de répertorier les besoins fonctionnels de PAAM.

Nous rappelons que le document que souhaite avoir Suzy contient :

- la vidéo de son dernier clip,
- une image représentant la pochette de son album,
- et un texte sur la bibliographie de la chanteuse.

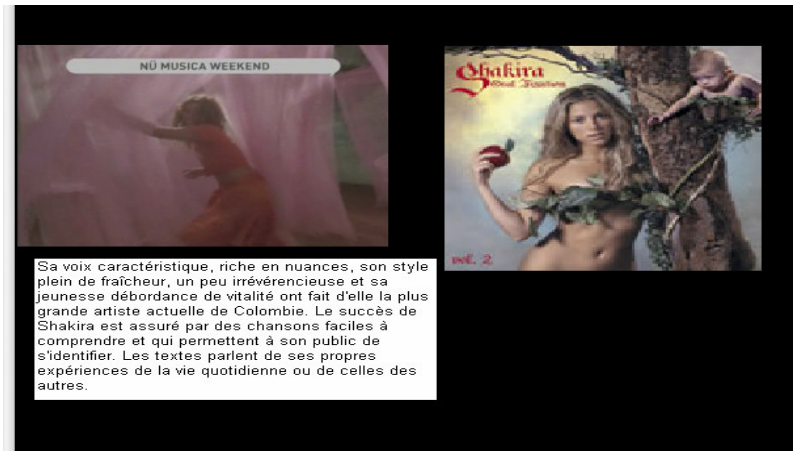

Figure 22 : Document source du scénario *Suzy and her PDA*

Suzy décide de télécharger le document. Grâce à un système adaptatif, le document sur « Shakira » est adapté aux préférences de Suzy et aux caractéristiques de son PDA :

- la vidéo est adaptée (transmodée) en contenu audio uniquement,
- le texte, initialement en français est traduit (transformé) en anglais
- et la taille de l'image est compressée (transformée) et transcodée vers un format pris en compte par son lecteur.

Plusieurs adaptateurs qui réalisent le transmodage existent. Un choix doit cependant être fait dépendant de la disponibilité et la charge de chaque adaptateur.

Si un des adaptateurs instanciés pour réaliser ce scénario disparaît avant la fin de l'adaptation, il doit être remplacé par un autre réalisant les mêmes fonctionnalités.

Figure 23 : Le document adapté au contexte de Suzy

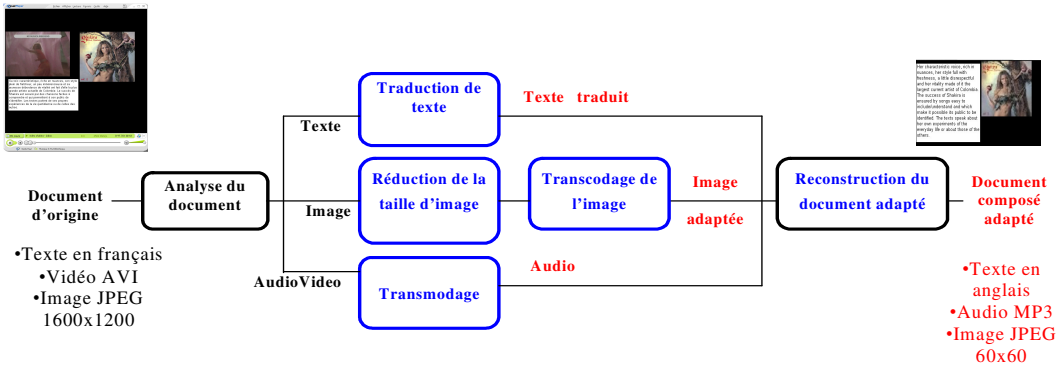

Figure 24 : Adaptations nécessaires pour le scénario *« Suzy and her PDA »*

2.1 Besoins fonctionnels pour réaliser le scénario

Le scénario énoncé ci-dessus suggère la présence des fonctionnalités suivantes :

- L'analyse du contexte de Suzy afin de détecter ses préférences et les capacités réduites de son PDA. Cette fonctionnalité est réalisée par le ***gestionnaire de contexte*** de notre système d'adaptation.
- L'analyse des médias composant le document multimédia. Cette fonctionnalité est réalisée par le ***gestionnaire de documents multimédia*** de notre système d'adaptation.
- La prise de décision pour adapter le document multimédia composé. Cette fonctionnalité est réalisée par le ***planificateur*** de notre système d'adaptation.
- La recherche, l'instanciation, la composition (dans le cas des adaptations relatives à l'image) et l'orchestration des adaptateurs. Cette fonctionnalité est réalisée par le ***gestionnaire d'adaptation*** de notre système d'adaptation.

- L'adaptation du document multimédia composé.
- La gestion des disparitions des adaptateurs réalisée par le ***gestionnaire d'adaptation***.

3 Architecture fonctionnelle de PAAM

La première contribution de cette thèse est de concevoir une architecture de fourniture de contenus multimédia adaptables au contexte des utilisateurs. Dans ce qui suit, nous présentons PAAM en décrivant les modules requis et leurs rôles dans le processus de fourniture d'un contenu multimédia à l'utilisateur final, se plaçant au plus près de ses exigences et des caractéristiques de son environnement d'usage. La Figure 25 représente une abstraction de l'architecture PAAM sous forme de trois couches:

- La couche applicative qui représente la logique d'adaptation PAAM.
- La couche de services qui permet à la couche applicative d'utiliser les ressources d'adaptation disponibles sur le réseau.
- La couche des adaptateurs qui englobe l'ensemble des acteurs réalisant l'adaptation.

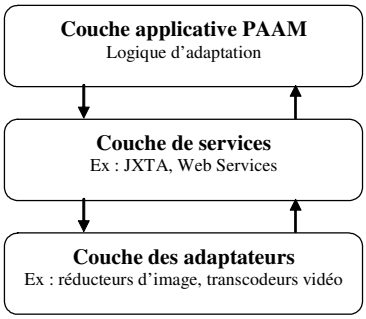

Figure 25 : Vue globale de la logique PAAM

La Figure 26 illustre l'architecture PAAM avec plus de détails tout en respectant la vue globale présentée ci-dessus :

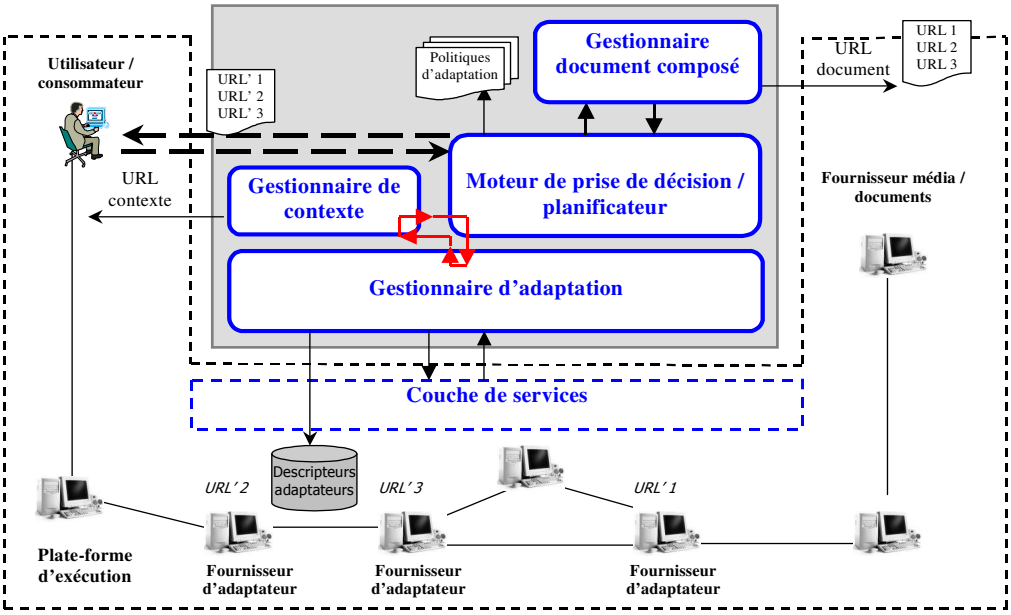

Figure 26 : L'architecture fonctionnelle de PAAM

Étant une architecture d'intégration, PAAM englobe plusieurs fonctionnalités. Certaines de ces fonctionnalités, telles que la gestion du contexte ou la gestion de la prise de décision, sont communes à d'autres architectures d'adaptation au contexte d'un utilisateur de contenus multimédia comme présenté dans le chapitre précédent. En revanche, PAAM intègre de nouveaux modules tels que le gestionnaire de documents composés multimédia ou encore le gestionnaire des adaptateurs distribués.

Chaque utilisateur désirant utiliser le système PAAM pour adapter, des documents multimédia, suivant son contexte, doit choisir un planificateur PAAM. Le choix du « meilleur » planificateur PAAM peut dépendre, par exemple, d'un de ces critères :

- La meilleure qualité du lien réseau entre l'utilisateur et le planificateur PAAM.
- Le planificateur PAAM qui prend en charge le plus grand nombre d'utilisateurs.
- Le planificateur PAAM le plus proche géographiquement de l'utilisateur.
- Le planificateur PAAM spécialisé dans les adaptations de transcodage vidéo
- Le planificateur PAAM spécialisé dans les adaptations relatives à un handicap.

Un utilisateur peut, s'il n'est pas satisfait du service rendu par le système PAAM choisi (par exemple temps de réponse trop long, absence de certains adaptateurs spécifiques, un nombre élevé de pannes), choisir un autre système PAAM.

3.1 Déroulement d'une requête typique dans PAAM

La Figure 27 présente le diagramme de séquence relatif au déroulement d'une requête typique d'un utilisateur qui utilise PAAM afin d'obtenir un document multimédia composé adapté à son contexte. Ce diagramme ne montre pas les adaptateurs mais intègre les quatre gestionnaires.

Nous supposons que le point d'entrée du système d'adaptation est le planificateur.

Durant tout le déroulement de la requête, le flux de données transite par une boucle d'adaptation initiée par le nœud consommateur (utilisateur). Cet utilisateur envoie une requête au planificateur de ce système d'adaptation. Cette requête contient un lien (interne ou externe) vers son contexte et l'URL (adresse) du document composé multimédia. Le planificateur :

- envoie au gestionnaire de contexte la description ou le lien vers la description du contexte de l'utilisateur ; ce gestionnaire analyse le contexte.
- envoie au gestionnaire de documents composés un lien vers la description du document multimédia ; ce gestionnaire analyse la description.

À la suite de cela, le gestionnaire de contexte et le gestionnaire de documents composés multimédia fournissent le résultat de leur analyse au planificateur.

Le planificateur, guidé par les politiques d'adaptation, décide si une adaptation est nécessaire ; dans ce cas le planificateur détermine les opérations d'adaptation requises.

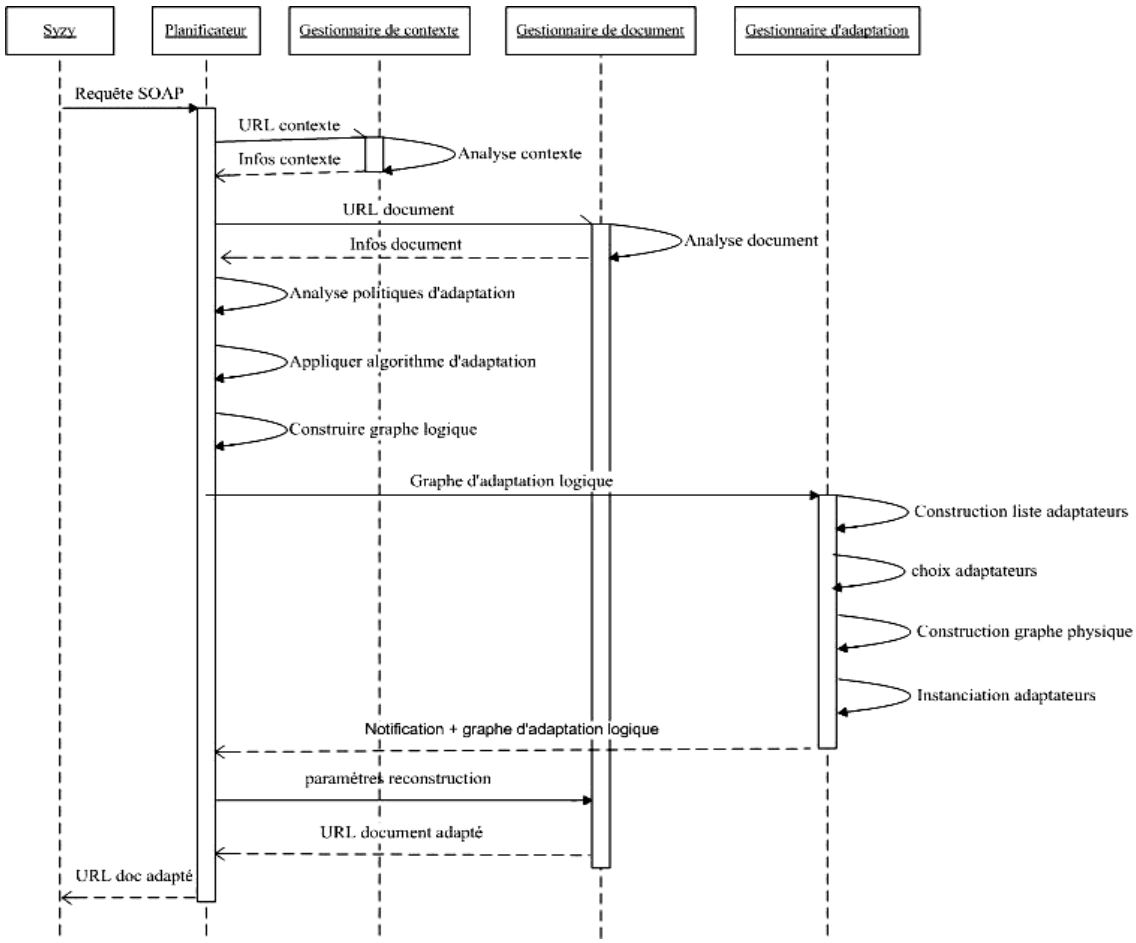

Figure 27 : Diagramme de séquence d'un scénario typique

Dans le cas du scénario *Suzy and her PDA*, le document de Suzy étant composé de trois médias, trois analyses par média sont nécessaires. Tout d'abord, Suzy a précisé dans ses préférences ne pas vouloir de contenu vidéo ; la vidéo doit être transmodée en audio. En analysant la langue du texte présent dans le document d'origine, le planificateur décide du besoin de traduire le texte du français vers l'anglais. Une autre analyse est réalisée par rapport aux capacités du PDA de Suzy ; ainsi la taille réduite de son écran suggère la réduction du document et plus spécialement de toutes les régions composant le document. L'image doit donc être réduite afin d'être contenue dans sa région (son espace) qui a été réduite.

Une fois ces prises de décision d'adaptation réalisées par le planificateur, un graphe d'adaptation logique est construit puis envoyé au gestionnaire d'adaptation. Un graphe d'adaptation logique est une description d'opérations d'adaptation enchaînées en séquence ou en parallèle. Ainsi, un graphe d'adaptation enchaînant en séquence deux adaptateurs fait référence au fait que la sortie du premier adaptateur est l'entrée du second adaptateur. De la même manière, un graphe d'adaptation parallélisant deux adaptateurs signifie que les deux adaptations sont complètement indépendantes et peuvent être

réalisées au même moment. Dans le graphe d'adaptation logique, aucune instanciation d'adaptateurs n'est réalisée.

Le gestionnaire d'adaptation construit au préalable une liste temporaire des adaptateurs présents dans son sous-réseau d'adaptateurs. La recherche de ces adaptateurs est facilitée par la présence de descripteurs fournis avec chaque adaptateur. Le gestionnaire d'adaptation choisit, grâce à cette liste, les bons adaptateurs. Il en résulte un graphe d'adaptation physique qui sert à instancier les adaptateurs.

Le planificateur est notifié de la fin des adaptations et analyse le graphe physique final afin d'envoyer une requête au gestionnaire de document pour une reconstruction du document multimédia composé.

Enfin, l'utilisateur reçoit via le planificateur l'adresse du document multimédia composé adapté à son contexte.

Remarque

Le diagramme de séquence ne tient pas compte de la gestion des disparitions d'adaptateurs ou des changements dans le contexte d'un utilisateur. Néanmoins, le graphe d'adaptation doit pouvoir être reconstruit dans le cas de changements significatifs aussi bien dans l'environnement de l'utilisateur que dans celui de la plate-forme des adaptateurs.

Nous détaillons dans ce qui suit les différents blocs composant l'architecture PAAM.

3.2 Gestionnaire de contexte

La Figure 28 illustre les fonctionnalités du gestionnaire de contexte ainsi que ses entrées / sorties.

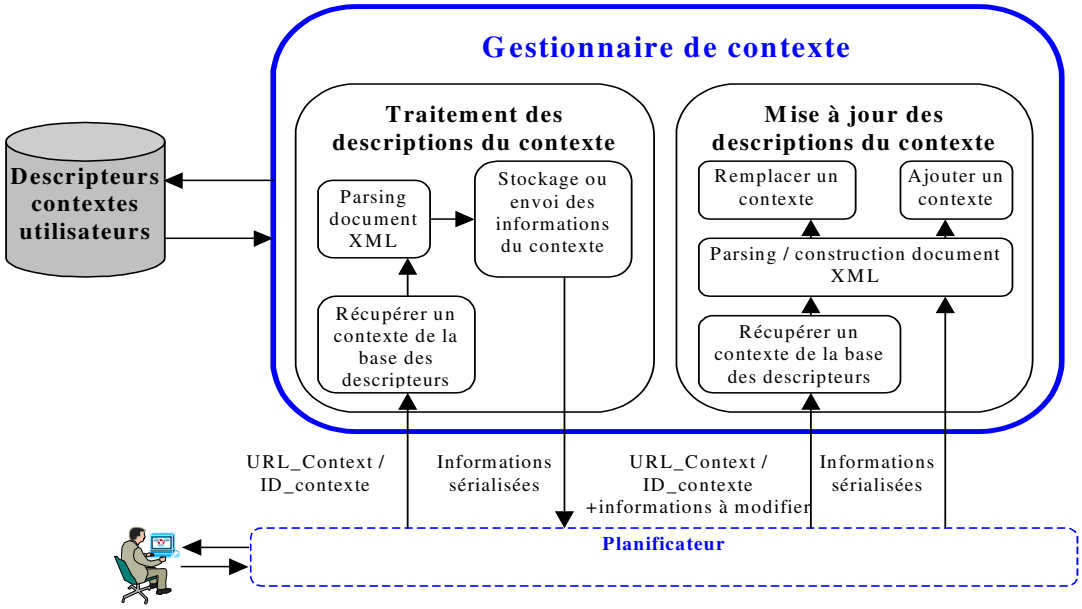

Figure 28 : Éléments du gestionnaire de contexte

Ce gestionnaire a pour but de récupérer les informations contextuelles, les analyser, les filtrer, et envoyer celles qui sont pertinentes au planificateur. Les informations du contexte utilisateur pertinentes peuvent être par exemple sa langue préférée, son handicap, la taille de l'écran de son terminal ainsi que les codecs supportés de son terminal.

Nous distinguons deux principales fonctionnalités fortement connexes : Le traitement du contexte et la mise à jour du contexte. Le gestionnaire du contexte collabore avec le planificateur comme le montre la figure ci-dessus.

Chaque usager désirant utiliser PAAM pour adapter des documents multimédia par rapport à son contexte doit fournir un lien à son planificateur PAAM vers son contexte. Ce lien donne accès aux préférences de l'utilisateur, aux capacités de son terminal et aux caractéristiques de son réseau d'accès. Ce lien peut être un identifiant, une adresse URL ou des coordonnées personnelles de l'utilisateur.

Les informations du contexte doivent respecter une syntaxe et une sémantique ; cela peut, par exemple, être réalisé en proposant à l'utilisateur, lors de la phase d'enregistrement au service d'adaptation PAAM, de remplir des fiches descriptives ou des formulaires préparés à cet effet et permettant de récupérer des informations du contexte de cet utilisateur. D'autres informations contextuelles, telle que la bande passante ou le type de terminal peuvent être récupérées sans l'intervention de l'utilisateur. La Figure 28 introduit un dépôt de descriptions contextuelles. Ce dépôt peut être une base de données locale au gestionnaire de contexte ou une base de données distribuée. À noter qu'un contexte peut également être généré à la volée et servir pour la durée de la session de l'utilisateur sans avoir à le stocker dans un dépôt mais uniquement dans le cache temporaire du gestionnaire d'adaptation.

Le gestionnaire de contexte intègre un module de parsing (analyse) des contextes utilisateurs (qui sont des documents basés sur XML).

En cas de changement dans le contexte utilisateur, le planificateur envoie au gestionnaire de contexte les informations à modifier avec un lien vers le contexte utilisateur concerné. Le gestionnaire de contexte met à jour le contexte en récupérant l'ancien contexte et en le modifiant.

La gestion du contexte est considérée comme un bloc fonctionnel élémentaire dans PAAM. Cependant, les contributions de notre thèse ne s'articulent pas autour de la conception et la réalisation d'un nouveau système de gestion de contexte. Nous nous reposons sur l'existant et ne faisons aucun choix quant à la façon dont sera distribué ce module.

Nous nous sommes néanmoins inspirés de certains standards tels que MPEG-21 [MPEG21], et nous avons pris la liberté d'y ajouter quelques modifications (cf. chapitre 6).

3.2.1 Descripteur du contexte utilisateur

L'objectif de cette thèse n'est pas basé autour de la réalisation d'un système de gestion du contexte. Nous proposons néanmoins une solution permettant de décrire le contexte.

Chaque langage de description offre des possibilités et tous ne permettent pas de décrire toutes les caractéristiques possibles d'un usager donné. Plus spécifiquement, compte tenu du scénario présenté dans la section 2, nous cherchons à exprimer les préférences utilisateur et les capacités du terminal suivantes:

- Les préférences de base de l'utilisateur telles que la langue parlée et le handicap.
- Les caractéristiques du terminal de base telles que la résolution de l'écran et son support ou pas de la couleur.
- Garder ou pas un média donné : dans le cas où Suzy décide de se passer de la vidéo même si la configuration logicielle et matérielle lui permet de lire une vidéo, et ce pour économiser la batterie et le volume des données transportées. Suzy peut également être en déplacement, ne pouvant regarder son PDA, elle peut décider de ne garder que l'audio d'un contenu multimédia.
- Les mimetypes supportés par chaque lecteur présent dans le terminal : Ces informations sont utiles pour rechercher des adaptateurs et adapter selon un format supporté par le terminal.

Notons que les caractéristiques du réseau ne sont pas présentées dans la liste ci-dessus mais pourraient être également pris en considération. Nous présentons notre modèle de description du contexte dans la section 4.1 du chapitre 6.

3.3 Gestionnaire du document multimédia composé

Le gestionnaire du document multimédia composé est l'entité qui manipule des documents multimédia pris en charge par PAAM. La Figure 29 illustre les fonctionnalités du gestionnaire de documents ainsi que ses entrées / sorties.

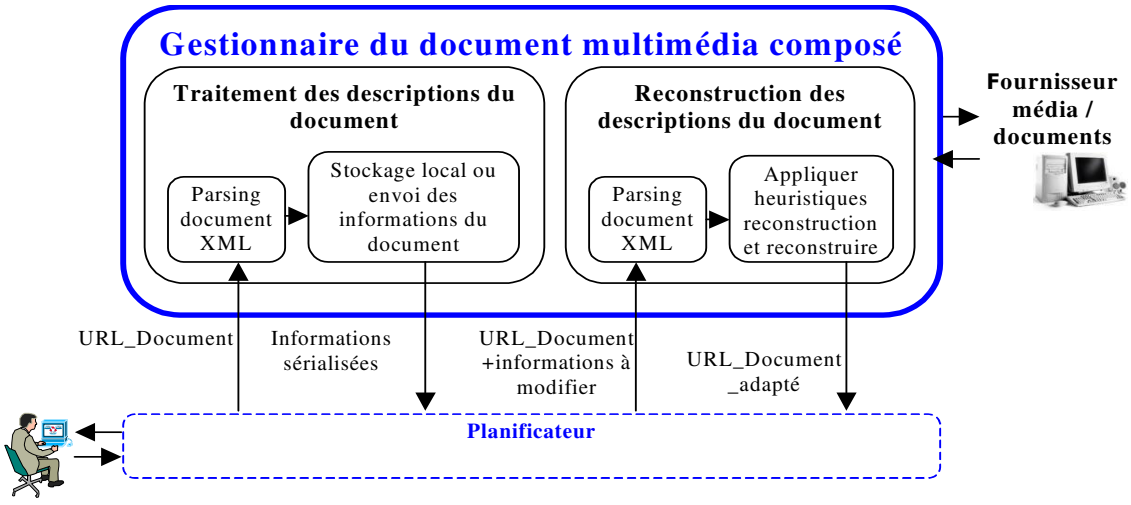

Figure 29 : Éléments du gestionnaire du document multimédia composé

Le gestionnaire du document multimédia composé a deux principales fonctionnalités : le traitement de la description du document multimédia composé et la reconstruction du document multimédia composé.

L'utilisateur envoie le lien vers un contenu multimédia au planificateur qui le transfert au gestionnaire du document multimédia composé. Ce dernier récupère auprès de la source du contenu multimédia la description relative à ce contenu. Le résultat de l'analyse de cette description est envoyé au planificateur. Ce dernier doit en tenir compte dans la phase de prise de décision pour l'adaptation.

Si un ou plusieurs médias élémentaires subissent des adaptations, le gestionnaire du document multimédia composé est de nouveau sollicité afin de transformer (réorganiser spatialement et re-synchroniser temporellement) le document composé initial en fonction de ces changements.

Le gestionnaire utilise quelques politiques de reconstruction simples afin de réaliser cette reconstruction. Deux catégories de politiques de reconstruction existent : les politiques relatives à la réorganisation spatiale et les politiques relatives à la synchronisation temporelle. Nous citons quelques unes de ces politiques :

- Deux régions contenant des images ou des des vidéos ne peuvent jamais se chevaucher
- Deux médias de types audio ou audio-visuel ne peuvent pas être joués en parallèle (synchronisation temporelle)

3.3.1 Descripteur du document composé :

Afin qu'un contenu multimédia puisse être pris en compte par le gestionnaire du document multimédia composé, le fournisseur de ce document doit en fournir une description. En effet, ce gestionnaire a besoin de connaître des informations relatives à chaque média élémentaire composant le document composé, et des informations relatives au document composé lui-même (par ex : la synchronisation spatiale des médias).

Les informations relatives au média et nécessaires au gestionnaire du document multimédia composé sont les suivantes :

- Le format des médias (ou mimetypes) composants le document multimédia
- La langue du ou des textes du document
- La taille de chaque média.
- La largeur et la hauteur de la région du document contenant chaque média composant le document multimédia
- La largeur et la hauteur du document multimédia

Les outils offerts par le standard MPEG-21 et par le langage multimédia SMIL permettent de décrire des contenus multimédia riches composés de plusieurs médias élémentaires et tenant compte de l'organisation spatiale et de la synchronisation temporelle de ces médias.

Nous avons présenté dans la section 7.3 du chapitre 2 un exemple d'un document SMIL composé d'une vidéo, d'une image et de texte. Les dimensions des régions (organisation spatiale) de chaque média y

sont représentées ainsi qu'une certaine synchronisation temporelle (par ex : la vidéo et l'image démarre en parallèle).

Nous présentons notre choix pour décrire les documents multimédia composés dans la section 4.2 du chapitre 6.

3.4 Planificateur

Ce module implémente des algorithmes de prise de décision pour l'adaptation de documents multimédia composés. Ces algorithmes utilisent les données traitées par les gestionnaires du contexte utilisateur et du document multimédia composé. Le planificateur s'appuie sur des politiques d'adaptation et sur des heuristiques.

La Figure 30 schématise les fonctions du planificateur.

Nous distinguons quatre grandes fonctionnalités : la récupération des informations du contexte utilisateur et du document multimédia composé, la prise de décision, la gestion des changements dans le contexte utilisateur et l'analyse du graphe d'adaptation final.

La récupération des informations du contexte utilisateur et du document multimédia composé est initiée par l'analyse de la requête d'un utilisateur qui doit contenir des informations relatives au contexte utilisateur (ou seulement un identifiant ou un lien vers son contexte si l'utilisateur s'est préalablement enregistré via le système PAAM choisi) et au document (URL du document) qu'il demande. Le planificateur récupère les informations manipulées par les gestionnaires du contexte et du document multimédia composé comme décrits dans les deux précédentes sections.

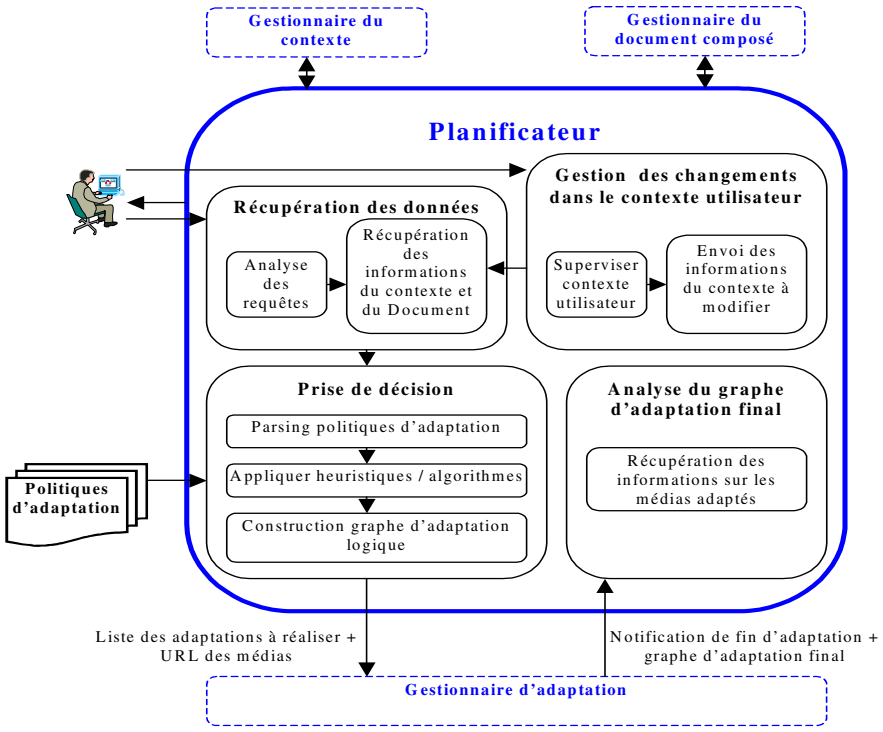

Figure 30 : Les fonctions du planificateur

La prise de décision est la principale fonctionnalité du planificateur. Elle prend en entrée les informations agrégées du contexte et du document multimédia composé et analyse les politiques d'adaptation. Rappelons qu'une politique d'adaptation est une règle déclarative contrôlant le choix du comportement d'un système [MLS04]. Le planificateur exécute ainsi l'algorithme de prise de décision en consultant, à des moments précis, ces politiques, plus précisément lorsqu'un choix, non donné *a priori* dans l'algorithme, doit être opéré.

Le processus de prise de décision génère une liste des adaptations à réaliser pour chaque média. Il construit ainsi un graphe d'adaptation logique. Un ***graphe d'adaptation*** est constitué d'un ensemble d'adaptations (adaptateurs) assemblés en séquence ou en parallèle. Le terme « logique » fait référence au fait qu'il n'y a pas encore d'instanciation à ce niveau. La Figure 24 illustre schématiquement un graphe d'adaptation logique.

Si, par exemple, un utilisateur change de terminal avant la fin des adaptations relatives au document qu'il a demandé, son contexte (les capacités de son terminal) change. Afin de prendre en considération ces changements, l'utilisateur doit faire parvenir ces changements à son planificateur. Ce dernier a pour rôle de modifier le graphe d'adaptation logique relatif au document multimédia composé demandé par cet utilisateur en réalisant une seconde prise de décision. Une fois ce graphe établi, une comparaison est réalisée entre lui et l'ancien graphe d'adaptation logique (gardé en mémoire tant que les adaptations relatives sont en cours). Cette comparaison sert à déterminer quels sont les adaptateurs à arrêter et à remplacer par d'autres adaptateurs tout en gardant les autres adaptateurs. Ceci peut minimiser les pertes d'informations et de temps. Ainsi, la gestion des changements dans le contexte utilisateur donne un aspect dynamique à PAAM.

D'autres changements ou événements relatifs aux adaptateurs pouvant se produire tels que la disparition d'un fournisseur d'adaptateur ou l'impossibilité de trouver le bon adaptateur sont gérés au niveau du gestionnaire d'adaptation. Ainsi, le graphe d'adaptation peut être modifié au niveau du gestionnaire d'adaptation (par ex. suppression du ou des processus d'adaptation relatifs à un média composant le document multimédia d'origine si l'adaptateur correspondant n'existe pas) ; ce dernier doit informer le planificateur en lui envoyant la dernière version du graphe d'adaptation à la fin de toutes les adaptations. L'analyse du graphe d'adaptation final permet de récupérer les informations nécessaires au gestionnaire du document multimédia composé et qui sont utilisées lors de la reconstruction du document multimédia adapté.

3.4.1 Algorithme générique du processus de prise de décision et d'adaptation

Nous présentons dans cette section un algorithme générique et simplifié du processus d'adaptation d'un document multimédia composé par rapport au contexte d'un utilisateur donné. Un algorithme plus détaillé est présenté dans le chapitre 6.

1. *Établir la liste locale des adaptateurs*

2. *envoi requête au gestionnaire du contexte et au gestionnaire du document*
3. *analyse de la description du contexte*
4. *analyse de la description du document*
5. *appliquer les politiques d'adaptation relatives au handicap*
6. *appliquer les politiques d'adaptation relatives aux médias*
7. *Construire le graphe d'adaptation logique*
8. *Rechercher les adaptateurs physiques*
9. *Appeler les adaptateurs*
10. *Superviser les adaptateurs*
11. *Reconstruction du document multimédia composé*
12. *Envoyer l'URL du document adapté à l'utilisateur*

3.5 Gestionnaire d'adaptation

Le gestionnaire d'adaptation utilise le graphe d'adaptation logique construit par le planificateur et recherche les adaptateurs qui implémentent les adaptations identifiées.

La Figure 31 illustre les principales fonctionnalités du gestionnaire d'adaptation.

La constitution de la liste temporaire des adaptateurs consiste à récupérer, à un moment donné, les descriptions des adaptateurs, stockées dans une base de données ou dans un annuaire, à les analyser et à en retirer les informations relatives au nom (identifiant) de chaque processus d'adaptation réalisé, les types de médias supportés en entrée et types de médias possibles en sortie. Ce point sera présenté en détail dans la section 2.4 du chapitre 5.

La recherche d'adaptateurs est la phase qui précède l'instanciation.

Tout d'abord, le gestionnaire d'adaptation récupère la liste des adaptations à réaliser depuis le planificateur et en récupère les types (noms) d'adaptateurs à rechercher. Le chapitre suivant présente notre proposition pour identifier de manière unique des ressources d'adaptation. Afin d'affiner la recherche, le gestionnaire d'adaptation considère d'autres critères contenus dans le graphe d'adaptation logique qui sont les types de formats en entrée des médias à adapter et les types de formats en sortie déterminés par le planificateur.

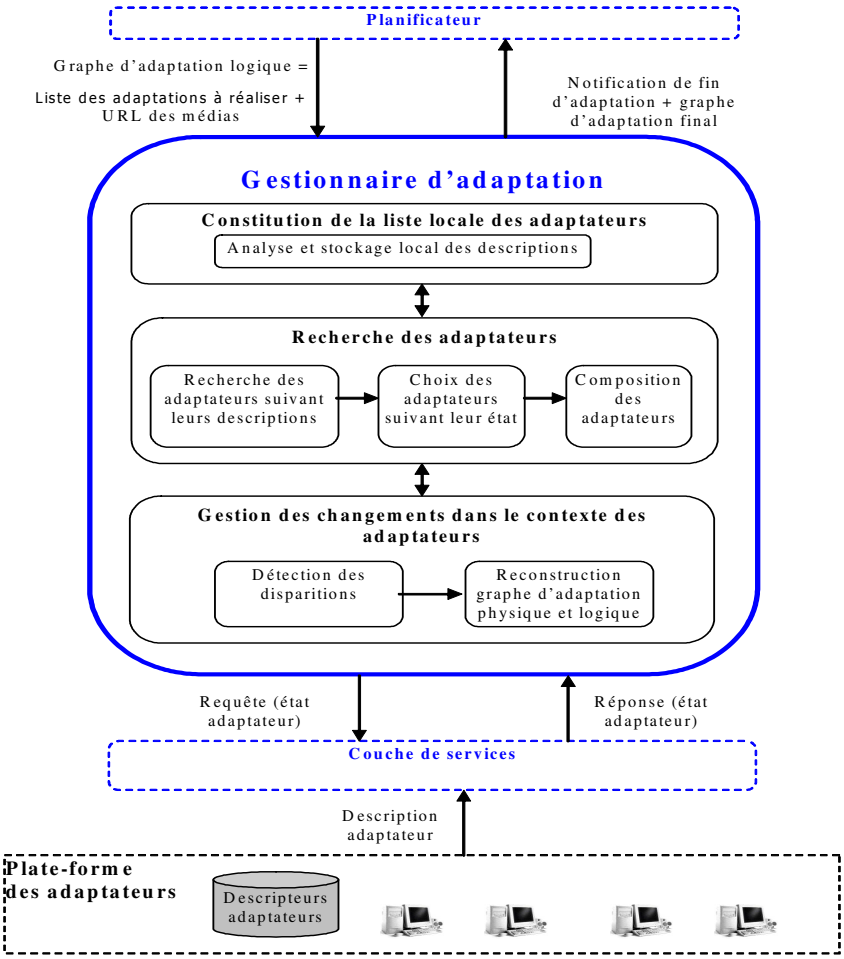

Figure 31 : Les éléments du gestionnaire d'adaptation

À l'issue de cette recherche, la liste des adaptateurs peut contenir plusieurs adaptateurs répondant aux mêmes critères. Le gestionnaire d'adaptation doit alors choisir le « meilleur » adaptateur.

Nous verrons dans le chapitre 5 le protocole de choix d'un adaptateur à travers une fonction que nous appelons *protocole de négociation et d'acceptation*. Néanmoins, le meilleur adaptateur peut être celui qui est offert par le fournisseur :

- Qui se déconnecte le moins souvent.
- Qui a le taux d'utilisation CPU le moins élevé.
- Qui appartient à un réseau de confiance.
- Qui offre un service gratuit.
- Qui offre un service payant avec garantie du bon déroulement de l'adaptation.

Si un adaptateur n'existe pas, le gestionnaire d'adaptation gère, s'il peut, la composition des adaptateurs. Prenons par exemple le graphe d'adaptation logique présenté dans la Figure 24. La traduction du texte (*TextTranslation*) se fait du français vers l'anglais. Le gestionnaire d'adaptation

recherche parmi les traducteurs, présents dans la liste temporaire, ceux qui prennent en entrée du français et ceux qui produisent en sortie de l'anglais. S'il n'existe pas de traducteur du français vers l'anglais, la traduction peut se faire en passant par un langage intermédiaire tel que l'espagnol, et en composant le traducteur du français vers l'espagnol et de l'espagnol vers le français. Ainsi, la partie *TextTranslation* du graphe d'adaptation logique se transformera finalement en deux adaptateurs dans le graphe d'adaptation physique de telle sorte que la sortie du premier traducteur soit l'entrée du second. Ceci met en évidence l'une des difficultés intrinsèques à la construction du graphe d'adaptation physique qui est la nécessité de composer plusieurs adaptateurs si le mécanisme de recherche ne trouve pas l'adaptateur souhaité.

Le gestionnaire d'adaptation, tout comme le planificateur, gère la dynamicité. En effet, d'autres changements (mis à part ceux qui concernent le contexte utilisateur) peuvent se produire concernant les adaptateurs. À travers les outils proposés par la couche de services, le gestionnaire d'adaptation dispose de mécanismes lui permettant de connaître la connectivité d'un adaptateur donné, plus spécialement, d'un adaptateur faisant partie d'un graphe d'adaptation physique. Ainsi, si un adaptateur vient à disparaître, le gestionnaire doit le remplacer par un ou plusieurs adaptateurs offrant le même service. Le graphe d'adaptation physique est reconstruit dynamiquement et le ou les nouveaux adaptateurs sont instanciés. C'est de cette manière que *PAAM* gère la dynamicité en présence de nœuds offrant des adaptateurs et sujets aux déconnexions fréquentes.

Ce changement dans le graphe d'adaptation physique implique un changement dans le graphe d'adaptation logique final (qui sert à reconstruire le document multimédia final). Par exemple, si aucun adaptateur n'est trouvé pour adapter un média présent dans le graphe d'adaptation logique initial, il est supprimé du document multimédia adapté final.

3.5.1 Descripteur de l'adaptateur :

Ce descripteur identifie de manière unique l'opération d'adaptation qu'il réalise. Il décrit également les paramètres d'entrée et de sortie de l'adaptateur.

Un adaptateur étant une ressource réseau offrant un service, il est logique de considérer un adaptateur comme un service. Les langages de description de services tels que SDL, WSDL ou OWL-S sont des exemples de langages permettant de décrire un service suivant différents critères [SDL] [WSDL] [OWLS]. La constatation faite à l'issue de l'analyse de ces outils descriptifs est qu'aucun langage n'est dédié à la description d'un service d'adaptation.

L'une de nos contributions est d'étendre le langage WSDL pour décrire plus spécifiquement une ressource d'adaptation et faciliter sa recherche. Nous avons proposé une nomenclature d'adaptateurs ainsi qu'un moyen d'identifier de manière unique un adaptateur. Nous développons en détails ce point dans le chapitre 5.

3.6 Couche de services

Comme le montre la Figure 26, *PAAM* repose sur une couche de services similaire, par exemple, à la couche de services de la plate-forme P2P telle que JXTA ou aux outils qu'offre une architecture orientée services telle que les services Web [JXTA]. Nous rappelons que ces deux plates-formes de service offrent des moyens d'annoncer, de répertorier et de rechercher des ressources ou des services. Elles permettent également à deux entités distantes de s'envoyer des messages et de communiquer.

Ainsi, la couche de services de PAAM :

- Fournit des outils aux fournisseurs d'adaptateurs pour déclarer et annoncer leurs ressources d'adaptation.
- Fournit des outils au gestionnaire d'adaptation pour rechercher les adaptateurs, communiquer avec eux et éventuellement les composer.

3.7 Plate-forme d'exécution

L'un des principes fondamentaux de PAAM est le fait qu'un nœud du réseau peut potentiellement jouer différents rôles. Un nœud peut être un :

- producteur en mettant à disposition des autres utilisateurs des contenus multimédias
- fournisseur d'adaptateur en mettant à disposition ses ressources matérielles et logicielles et en proposant d'adapter localement des contenus multimédia grâce à ses adaptateurs
- consommateur en récupérant des contenus multimédia adaptés au contexte d'usage de l'utilisateur concerné, i.e. aux préférences de cet utilisateur, aux capacités de son terminal, ainsi qu'aux caractéristiques de son réseau d'accès.

La plate-forme d'exécution de PAAM repose ainsi sur un modèle architectural client / intermédiaires / serveur étendu à un modèle où chaque nœud est client, intermédiaire et serveur.

Nous venons de décrire en détail les principales fonctionnalités de l'architecture PAAM. Nous avons présenté les principales entrées/sorties de chaque module.

La section qui suit décrit le déploiement à large échelle de PAAM.

4 Passage à l'échelle de PAAM

L'un des besoins auquel doit répondre PAAM est le passage à l'échelle (scalability en anglais). En effet, l'idée est que tous les utilisateurs d'Internet puissent profiter d'un système d'adaptation de contenus multimédia tel que PAAM. Le nombre de ces utilisateurs ne cessant de croître, nous discutons dans cette section du passage à l'échelle de PAAM.

La Figure 32 montre quatre groupes d'utilisateurs, chaque groupe utilisant un système d'adaptation PAAM différent. Un **système d'adaptation PAAM** est composé d'un gestionnaire de contexte, d'un

gestionnaire de document multimédia composé, d'un planificateur et d'un gestionnaire d'adaptation ; le point d'entrée de l'utilisateur étant le planificateur. Les systèmes d'adaptation PAAM partagent éventuellement les mêmes adaptateurs.

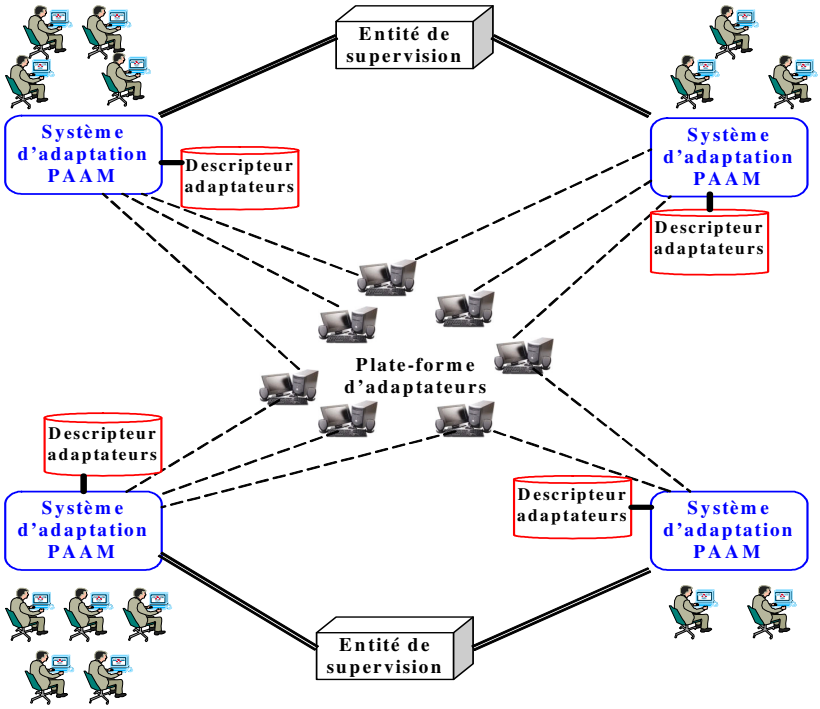

Figure 32 : La passage à l'échelle de PAAM

Afin qu'un grand nombre d'utilisateurs profitent du service d'adaptation de PAAM, plusieurs systèmes d'adaptation PAAM peuvent être hébergés et déployés sur des nœuds du réseau.

Un système d'adaptation PAAM peut être vu comme un serveur (fournisseur de services) ayant pour but de recevoir les requêtes des utilisateurs, de les analyser, de prendre des décisions d'adaptation et d'instancier les « meilleurs » adaptateurs.

Un système d'adaptation PAAM :

- Supporte un nombre fixe d'utilisateurs (consommateurs de documents multimédia composés).
- possède un planificateur utilisant ses propres politiques d'adaptation, son propre algorithme d'adaptation et ses propres heuristiques. Par exemple, si un utilisateur envoie la même requête à deux systèmes d'adaptation PAAM, il est possible que les planificateurs respectifs ne produise pas le même graphe d'adaptation et que le document final ne soit pas le même. Un utilisateur peut ainsi changer de système d'adaptation PAAM s'il n'est pas satisfait du service d'adaptation rendu.
- Est indépendant des autres systèmes d'adaptation PAAM : Un couplage faible caractérise le lien entre les systèmes d'adaptation PAAM. Il n'est cependant pas exclu que deux systèmes

d'adaptation PAAM communiquent afin, par exemple, d'enrichir leur liste temporaire des descriptions des adaptateurs.

- Peut, à la suite d'une requête utilisateur, ne pas posséder les bons outils (politiques ou algorithme d'adaptation) ou ne pas connaître les bons adaptateurs ou peut avoir atteint son nombre maximum d'utilisateurs. Cette requête utilisateur ne pouvant être satisfaite, ce dernier en est informé et doit choisir un autre système d'adaptation PAAM.

La Figure 34 montre également la présence dans le réseau d'entité de supervision. Nous le décrivons dans ce qui suit.

4.1 Entité de supervision

La présence d'une ou de plusieurs entités de supervision n'a de sens que s'il y a plusieurs systèmes d'adaptation PAAM dans le réseau. Le rôle d'une entité de supervision est de fournir, aussi bien pour un système d'adaptation PAAM que pour un utilisateur ou un fournisseur d'adaptateur, une interface d'accès à PAAM. Nous donnons plus de détails sur les interfaces d'accès dans la section qui suit.

Une entité de supervision contient, localement, des informations telles que les URLs et les positions géographiques des systèmes d'adaptation et des descriptions des adaptateurs.

Un utilisateur, un adaptateur ou un système d'adaptation PAAM n'utilise l'entité de supervision que pour s'annoncer, s'enregistrer et récupérer les informations qui les intéressent.

4.2 Interfaces d'accès à PAAM

Une interface d'accès à PAAM est nécessaire pour chacune des trois entités suivantes : l'utilisateur, le système d'adaptation PAAM et l'adaptateur comme le montre la Figure 33. Chaque interface offre des possibilités différentes suivant le type d'entité et permet l'ajout d'un utilisateur PAAM, d'un adaptateur PAAM ou d'un système d'adaptation PAAM.

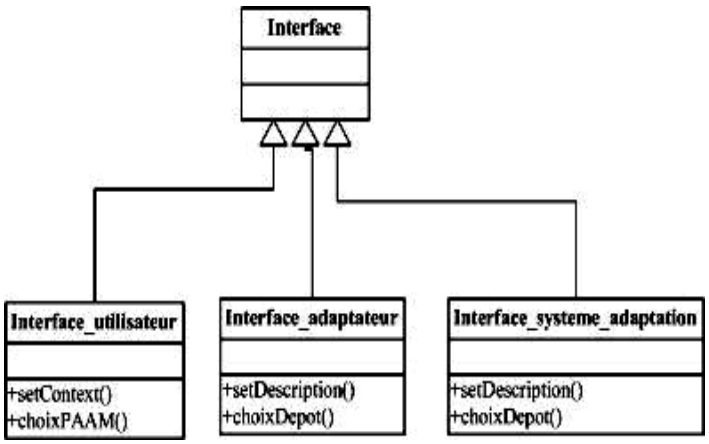

Figure 33 : Les différentes interfaces d'accès

4.2.1 Ajout d'un système d'adaptation PAAM

Chaque nœud du réseau, peut implémenter et proposer son propre système d'adaptation PAAM à condition de supporter en entrée les descriptions PAAM du contexte et du document multimédia composé et de pouvoir en sortie instancier et dialoguer avec les adaptateurs. Ce nœud doit également posséder une bonne connectivité réseau et être robuste et pouvoir monter en charge.

Un système d'adaptation PAAM doit être décrit et répertorié dans un dépôt (par exemple un annuaire). Cette description contient une référence vers le système d'adaptation PAAM (par exemple son adresse IP, le port d'écoute et le nom du service) et éventuellement une description de ce que sait faire le système d'adaptation PAAM (par ex : système d'adaptation spécialiste des transcodages vidéo ou des adaptations en relation avec le handicap) et son prix.

4.2.2 Ajout d'un adaptateur

La description d'un adaptateur peut être répertoriée dans plusieurs bases ou annuaires. Ainsi un adaptateur peut être recherché et instancié par plusieurs systèmes PAAM.

Un fournisseur d'adaptateur souhaitant participer à la logique d'adaptation PAAM doit, via une interface :

- Donner la catégorie d'adaptation à laquelle appartient cet adaptateur. Le chapitre 5 présente une identification unique des adaptateurs.
- Fournir une description des paramètres d'entrée et de sortie de l'adaptateur conforme à PAAM. Nous verrons dans le chapitre 5 les caractéristiques d'une telle description.
- Annoncer, sa description à un ou plusieurs dépôts de descriptions (bases de données locales ou distribuées, ou annuaires)

4.2.3 Ajout d'un utilisateur

L'interface utilisateur propose à l'usager un certain nombre de systèmes d'adaptation PAAM. Ce dernier choisit celui qui convient le mieux suivant certains critères tels que la spécialité du système d'adaptation PAAM, sa localité, son prix et éventuellement les appréciations d'autres utilisateurs. La notion de réseau de confiance a également tout son sens pour choisir un système d'adaptation PAAM.

Cette interface peut permettre, par exemple, à l'utilisateur d'éditer son contexte et de le modifier si un changement dans son contexte a lieu (par exemple un changement de terminal).

4.3 Discussion sur le passage à l'échelle

Le passage à l'échelle de PAAM est rendu possible grâce aux mécanismes d'ajout ou de suppression d'entités (utilisateur, adaptateur ou système d'adaptation PAAM) sur différents nœuds du réseau.

Le passage à l'échelle entraîne cependant des connexions massives d'utilisateurs à des systèmes d'adaptation, pouvant créer ainsi des goulots d'étranglement. Ceci est évité grâce à la supervision des

systèmes d'adaptation. Ainsi, un mécanisme de négociation entre l'entité de supervision et système d'adaptation PAAM est mis en place afin de récupérer l'équilibrer la charge entre les systèmes d'adaptation PAAM. Ce mécanisme est identique au protocole de négociation et d'acceptation que nous décrivons dans la section 3.2 du chapitre 5. Ces informations sont mises à jour au niveau de l'interface utilisateur afin que ce dernier puisse choisir un système d'adaptation capable suivant sa charge.

5 Conclusion

Dans ce chapitre, nous avons présenté l'architecture fonctionnelle PAAM. Nous avons décrit les fonctionnalités principales de PAAM qui sont : la gestion du contexte, la gestion du document multimédia, la gestion de la prise de décision et la gestion des adaptateurs.

Nous avons présenté le fonctionnement à large échelle de PAAM pour qu'il puisse être utilisé par le plus grand nombre d'utilisateurs.

Le passage à l'échelle de PAAM implique des caractéristiques telles que l'extensibilité, la modularité et la portabilité. Nous présentons, dans le chapitre suivant, l'intérêt d'une implémentation basée sur la technologie des services Web afin de répondre à ces besoins.

Chapitre 4

PAAM et l'utilisation des services Web

1 Introduction

L'objectif principal de PAAM est la mise en place d'une adaptation distribuée sur différents nœuds du réseau, généralisant le schéma classique où l'adaptation est réalisée à la source du contenu ou à un intermédiaire. PAAM doit alors reposer sur une plate-forme technologique qui lui permettant d'être décentralisée, extensible, modulable, tolérante aux fautes et passant à l'échelle (scalable), répondant ainsi aux limitations des autres architectures d'adaptation. Cette solution technologique doit également permettre à PAAM de décrire des ressources d'adaptation, de les publier, de les rechercher et de les instancier. Les plates-formes P2P répondent à ce dernier besoin mais nous avons plutôt opté pour la technologie des services Web pour les raisons suivantes :

- Les services Web sont plus faciles à prendre en main
- Les services Web sont, tout autant que les plates-formes P2P commerciales, largement utilisés.
- Les services Web sont, tout autant que les plates-formes P2P sont facilement déployables
- Certaines solutions P2P ne passent pas à l'échelle (par exemple JXTA [JXTA]).
- Les plates-formes P2P sont souvent propriétaires à l'inverse des services Web qui peuvent être utilisés par tout développeur à condition d'avoir les bonnes librairies, souvent gratuites et standardisées.
- Les services Web offrent des outils de description, de recherche et d'annonce de ressources (éventuellement des adaptateurs), ce qui n'est pas le cas d'une solution basée sur les outils P2P.
- Des efforts relatifs à l'orchestration et à la composition des services Web sont réalisés à travers le standard BPELque nous décrivons plus loin dans ce chapitre [Mat06].

Le choix d'utiliser des services Web pour implémenter PAAM étant réalisé, ce chapitre a pour but de présenter les concepts de base des services Web : SOAP (Simple Object Access Protocol), le protocole de communication des services Web, WSDL (Web Services Description Language), le langage de

description des services Web et UDDI (Universal Description Discovery and Integration), un annuaire pour les services Web.

La suite de ce chapitre s'organise comme suit : Tout d'abord, nous donnons une définition des architectures orientées services (SOA[6]) auxquelles appartiennent les services Web. Ceci permet de situer les services Web par rapport à un concept plus générique. Nous donnons par la suite un aperçu des outils de services Web qui permettent à PAAM d'annoncer les ressources d'adaptation (ou adaptateurs), les rechercher et les exécuter. Nous présentons ensuite BPEL, un langage permettant de faire de la composition et de l'orchestration de services Web. Nous concluons ce chapitre en présentant les notions de modularité, d'extensibilité et de portabilité associées aux services Web, notions fondamentales dans le cadre de PAAM.

2 Outils des services Web pour annoncer, découvrir et exécuter

Le système d'adaptation de PAAM est composé d'adaptateurs distribués pouvant être annoncés, découverts et exécutés à distance via une façade exposant les opérations offertes par ces adaptateurs. Les architectures orientées services et plus spécifiquement les services Web permettent de réaliser ces opérations.

2.1 Services Web et SOA

Nous distinguons une application qui utilise des services Web et une application basée sur une architecture orientée services. En effet, les services Web représentent l'implémentation la plus utilisée des SOA à travers le Web. Il existe cependant d'autres implémentations SOA telles que les applications à base de composants EJB (Enterprise Java Beans) [EJB].

Une SOA est un modèle de conception basé sur concept l'encapsulation de la logique de l'application par des services interagissant via des protocoles de communication communs [Erl04]. Les SOA se caractérisent par :

- Une forte cohérence interne utilisant un format d'échange pivot, le plus souvent XML.
- Des couplages externes lâches, le plus souvent via une couche d'interface interopérable.

La SOA représente la relation entre trois types de participants : le fournisseur de service, l'organisme de découverte de services et le demandeur du service [Papa03]. Les interactions comprennent les opérations « publish », « find » et « bind » (littéralement publier, trouver et lier) comme le montre la figure qui suit :

6 SOA : Service Oriented Architecture

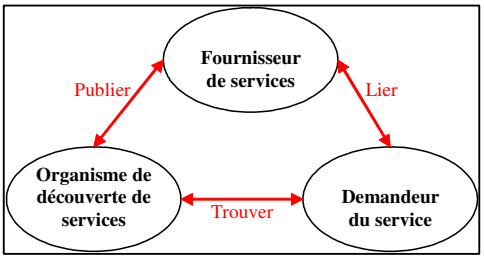

Figure 34 : L'architecture orientée services de base

Dans un scénario typique, un fournisseur de service héberge un module logiciel accessible via le réseau (l'implémentation d'un service donné), définit une description de ce module (ou service) et le publie vers un client ou vers un organisme de découverte.

En étant implémenté avec des services Web, PAAM suit le modèle SOA. En effet, le fournisseur de services d'un SOA représente le fournisseur d'adaptateurs, le demandeur de services représente le système d'adaptation PAAM et l'organisme de découverte des services représente une partie de la couche de services sur laquelle repose PAAM.

Nous définissons dans ce qui suit les services Web et présentons les outils qu'ils proposent et que PAAM utilise.

2.2 Définition des services Web

Un service Web est un composant logiciel modulaire, complet, fonctionnel et auto-descriptif. Un service Web est disponible à travers le Web grâce à la publication (ou l'annonce). Il peut être facilement localisé, invoqué et consommé à travers le Web [Gur04].

La Figure 35 introduit les bases des services Web : **SOAP** (*Simple Object Access Protocol*), **WSDL** (*Web Services Description Language*) et **UDDI** (*Universal Description Discovery and Integration*) que nous présentons dans ce qui suit.

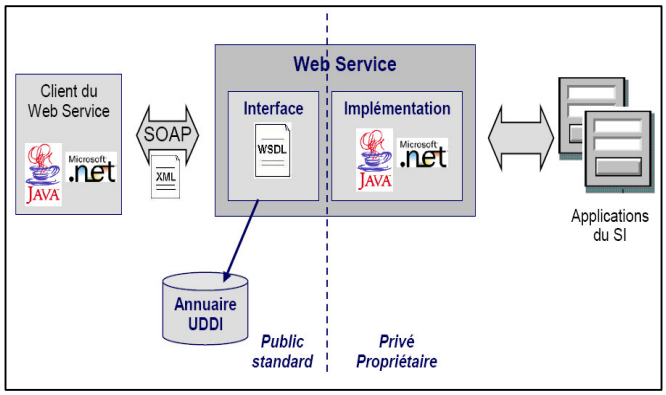

Figure 35 : Le concept de base des Services Web

À travers ce qui a été présenté, les outils standards des services Web permettent :

- De publier des ressources grâce à UDDI
- De rechercher des ressources grâce à UDDI et WSDL
- D'instancier des ressources grâce à une description WSDL
- À deux applications de communiquer et d'échanger des données grâce à SOAP

2.3 SOAP

SOAP est une recommandation W3C qui le définit comme un protocole léger destiné à l'échange d'informations structurées dans un environnement distribué et décentralisé [Chauv02] [SOAP].

SOAP est basé sur XML afin de mettre en place un mécanisme extensible d'échanges de messages à travers une variété de protocoles. Il a été conçu dans l'optique d'être indépendant du modèle de programmation de l'application ou d'une sémantique particulière.

SOAP vise deux buts : la simplicité et l'extensibilité. SOAP peut être utilisé dans tous les styles de communication : synchrone ou asynchrone, point à point ou multipoints, intranet ou internet.

Un message SOAP comprend :

- Une enveloppe dans laquelle sont définis le contexte du message, son destinataire, son contenu et éventuellement des options.
- Des règles de codage, nécessaire afin de définir la représentation des données d'une application à l'intérieur du corps d'un message SOA.
- Un protocole d'appel distant de méthodes tel que RPC (Remote Procedure Call) dont le but est de déterminer la succession des requêtes et des réponses.
- Le choix du protocole de transport des messages SOAP (souvent HTTP) .

XML Schema est souvent utilisé par les règles d'encodage pour décrire la structure des données formant le message SOAP.

2.4 WSDL

WSDL ou Web Service Description Language, une recommandation W3C, est un format de description des services Web également basé sur XML [Chauv02] [WSDL].

WSDL définit les services réseau sous forme d'un ensemble d'opérations et de messages décrits de manière abstraite et reliés à des protocoles et à des serveurs réseaux concrets.

WSDL est extensible afin de décrire des opérations et les messages correspondant à ces opérations, en fonction du format de ces messages et des protocoles utilisés pour communiquer.

La spécification WSDL 1.0 définit les éléments suivants :

- **Types :** Un contenaire de définitions des types de données. Ce contenaire comprend quelques types système tels que XSD.
- **Message :** Une définition abstraite et typée des données échangées entre services Web.
- **Operation :** Une description abstraite d'une action implémentée par un service Web.
- **Port Type :** Un ensemble abstrait d'opérations réalisées par une ou plusieurs extrémités (endpoints).
- **Binding :** Un protocole et une spécification de format de données concrets pour un *port-type* donné.
- **Port :** Représente une extrémité (endpoint) définie comme une combinaison de liaison (binding) et une adresse réseau.
- **Service :** une collection de ports.

La Figure 36 montre la description d'un service Web d'adaptation qui prend en entrée l'URL d'une image et retourne sa largeur et sa hauteur.

```
<definitions targetNamespace="http://server.async/" name="GetImageHWImplService">
<types>
<xsd:schema>
        <xsd:import namespace=http://server.async/
        schemaLocation="http://localhost:8084/GetImageHW/GetImageHW?xsd=1"/>
</xsd:schema>
</types>

<message name="getHW">
        <part name="imageURL" element="tns:getHW"/>
</message>

<message name="getHResponse">
        <part name="height" element="tns:getHResponse"/>
</message>

<message name="getWResponse">
        <part name="width" element="tns:getWResponse"/>
</message>

<portType name="GetImageHWImpl">
        <operation name="getHW">
                <input message="tns:getHW"/>
                <output message="tns:getHResponse"/>
                <output message="tns:getWResponse"/>
        </operation>
</portType>

<binding name="GetImageHWImplPortBinding" type="tns:GetImageHWImpl">
<soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
        <operation name="getHW">
                <soap:operation soapAction=""/>
                <input>
                        <soap:body use="literal"/>
                </input>
                <output>
                        <soap:body use="literal"/>
                </output>
        </operation>
</binding>

<service name="GetImageHWImplService">
        <port name="GetImageHWImplPort" binding="tns:AddNumbersImplPortBinding">
        <soap:address location="http://localhost:8084/GetImageHW/GetImageHWImpl"/>
        </port>
</service>
```

Figure 36 : Exemple de description WSDL

2.5 UDDI

UDDI ou Universal Description, Discovery and Integration a pour but d'établir un format d'annuaire des services Web. UDDI est le fruit d'un effort commun des grands fournisseurs de plates-formes et de logiciels au sein du Consortium de standards OASIS [OASIS].

UDDI a créé une plate-forme interopérable qui permet aux compagnies et aux applications de trouver et utiliser rapidement, facilement et dynamiquement les services Web à travers le Web [Chauv02] [UDDI].

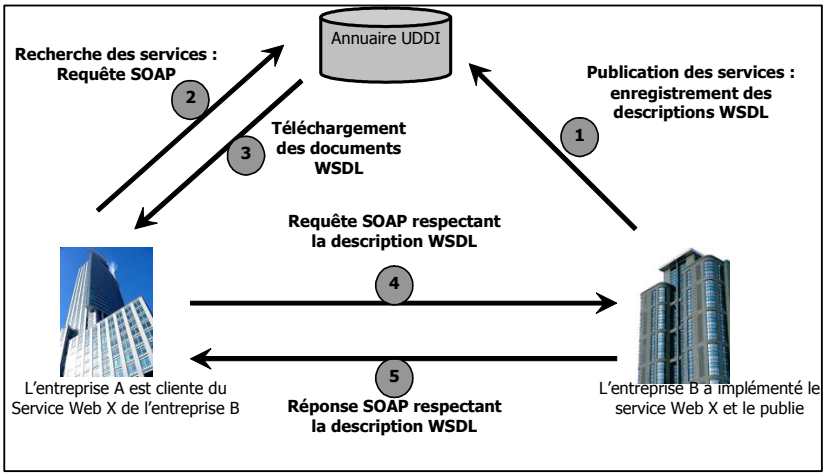

Figure 37 : UDDI et le protocole d'annonce et de découverte de descriptions WSDL en utilisant des messages SOAP

3 Composition et orchestration des services Web avec BPEL/WS-BPEL

BPEL (*Business Process Execution Language* for Web Services) appelé également WS-BPEL ou BPEL4WS est un langage utilisé pour la composition, l'orchestration et la coordination des services Web [Mat06]. Il fournit un vocabulaire riche pour exprimer le comportement des processus d'entreprise.

Un **processus d'entreprise** (*business process* en anglais) est une collection d'activités à travers lesquelles les services sont invoqués.

BPEL permet l'appel synchrone ou asynchrone des services Web. L'invocation des opérations des services Web se fait en parallèle ou en séquence, avec ou sans conditions et avec ou sans synchronisation. Des mécanismes évolués permettent de gérer les erreurs, de déclarer des variables, de les copier et de leur affecter des valeurs. Ainsi, BPEL définit, de manière algorithmique, des processus complexes.

La composition des services peut se faire suivant deux méthodes : l'orchestration et la chorégraphie.

L'**orchestration** est réalisée par un processus central dont le but est de contrôler les services Web et de coordonner l'exécution des opérations cibles. Les services Web concernés ne savent pas qu'ils sont inclus dans la composition. L'orchestration est ainsi centralisée.

La **chorégraphie** n'utilise pas de coordinateur central. Chaque service Web participant à la chorégraphie sait quand exécuter l'opération et avec qui interagir. La chorégraphie est un effort collaboratif basé sur l'échange de messages.

Du point de vue de la composition de services Web dans le cadre de PAAM, l'orchestration est plus avantageuse que la chorégraphie. En effet, l'orchestration permet, à partir d'un point central, de

superviser le bon déroulement des adaptations, ce qui permet une bonne maîtrise de la composition. Notons également que la chorégraphie suppose une cohésion forte entre les adaptateurs, ce qui n'est le cas dans PAAM. Le seul cas où deux adaptateurs doivent se communiquer des résultats est lorsque l'entrée d'un adaptateur dépend de la sortie d'un autre ; ceci est géré par le gestionnaire d'adaptation de PAAM, ce qui place PAAM dans une logique d'orchestration.

Les avancées de BPEL ont été trop tardives pour être introduites dans notre implémentation. Cependant, nous pensons que BPEL peut contribuer à automatiser l'exécution du graphe d'adaptation. Ainsi, nous décrivons dans ce qui suit comment pourrait être intégré l'outil BPEL au sein de PAAM. Nous présentons également les principales caractéristiques de BPEL permettant d'orchestrer les adaptateurs d'un graphe d'adaptation de PAAM avant de présenter un exemple illustrant la composition de trois adaptateurs.

3.1 Utilisation de BPEL dans PAAM

BPEL permet de composer et d'orchestrer des services Web de manière statique et/ou dynamique. La figure qui suit schématise l'intégration d'un moteur d'exécution de processus BPEL au sein du gestionnaire d'adaptation de PAAM.

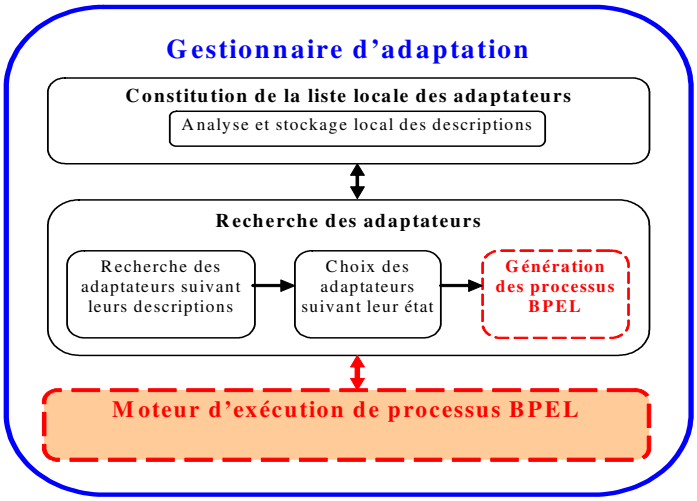

Figure 38 : Intégration du moteur d'exécution des processus BPEL au sein du gestionnaire d'adaptation de PAAM

La Figure 38 montre de manière fonctionnelle comment intégrer au gestionnaire d'adaptation un moteur d'exécution de processus BPEL permettant d'orchestrer des adaptateurs.

La phase de génération du processus BPEL doit consister en :

- La récupération des descriptions WSDL des adaptateurs et leur extension en ajoutant les balises *partnerLinkType*.
- La génération de la description du processus BPEL d'adaptation respectant le contenu du graphe d'adaptation physique construit par le gestionnaire d'adaptation.

Le moteur d'exécution des processus BPEL est un serveur BPEL (voir les exemples de serveur BPEL dans la section 3.4) qui utilise les descriptions WSDL et la description du processus BPEL d'adaptation en entrée et réalise les appels aux adaptateurs tout en respectant les enchaînements que décrit le graphe d'adaptation (séquence ou parallélisme).

PAAM peut de cette manière utiliser BPEL afin de composer les adaptateurs et orchestrer leurs exécutions. L'intégration d'un moteur d'exécution de processus BPEL au gestionnaire d'adaptation permettrait une automatisation de la composition d'adaptation et une extensibilité de PAAM.

Il existe des travaux qui utilisent BPEL pour composer et orchestrer les services. Nous citons par exemple les auteurs de [SEc07] qui proposent une architecture appelée MCDN (Multimedia Content Discovery and Delivery). L'objectif de MCDN est de fournir des solutions standardisées permettant à des producteurs de contenus de rechercher et fournir du contenu adapté et personnalisé dans des environnements dynamiques et hétérogènes en développant une infrastructure accessible à n'importe quel fournisseur de contenu.

Un des composants de MCDN est l'aggrégateur qui intègre une API extensible permettant à des développeurs d'accéder facilement aux services MCDN. Un de ces services est la recherche personnalisée de contenu en assemblant les composants MCDN dédiés au contenu, aux méta-données, à la personnalisation, à la sécurité et à la gestion transparente de session. L'aggrégateur a été implémenté en utilisant les services Web et BPEL.

3.2 Exemple générique d'un processus BPEL

Pour ses clients, un processus BPEL prend l'aspect d'un service Web. La définition d'un processus BPEL consiste à définir un nouveau service Web qui est une composition de services existants. L'interface du nouveau processus BPEL (ou du service Web composite) utilise un ensemble de port types à travers lesquels il propose ses opérations comme n'importe quel service Web. Pour invoquer un service Web décrit avec BPEL, il suffit d'invoquer le service Web composite résultant. La figure qui suit schématise un processus BPEL.

La Figure 39 met en scène un processus BPEL qui attend la requête d'un client grâce à la balise <receive>, et qui lui répond via la balise <reply>. Le processus BPEL compose deux services Web (service Web 1 et 2). Les invocations de ces services Web sont réalisées par le biais des balises <invoke> et d'autres balises (par exemple des balises de séquence ou de condition) que nous ne détaillons pas dans cette figure.

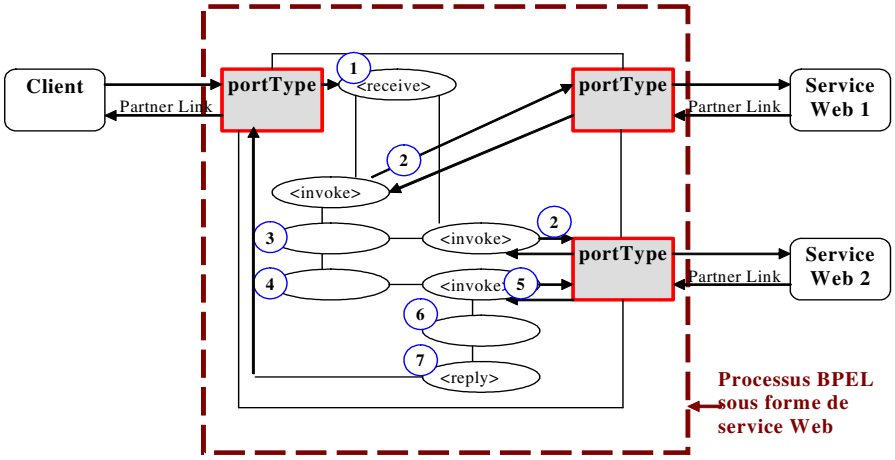

Figure 39 : Exemple générique d'un processus BPEL

Nous présentons dans ce qui suit les différents concepts de BPEL qui permettent de générer une composition de services Web.

3.3 Concepts de base de BPEL

Un processus BPEL interagit avec les services Web externes de deux manières :

1. en invoquant les opérations des services Web
2. en recevant des invocations des clients. Un client du processus BPEL peut être un utilisateur qui appelle en premier le processus BPEL. Les services Web externes peuvent également être des clients dans le cas où ils ont été invoqués par le processus BPEL et le rappellent afin de retourner leurs réponses de manière asynchrone (cas des *callbacks*).

Nous présentons dans ce qui suit les concepts de base de BPEL permettant de décrire les relations entre les entités, de déclarer des variables et de définir le processus BPEL.

3.3.1 Liens partenaires

BPEL permet de spécifier les interactions ou relations entre les services Web composant le processus d'entreprise. Ces relations sont appelées liens partenaires (en anglais *partner links).* Un lien partenaire est une référence d'un service avec lequel interagit un processus d'entreprise BPEL. Une analogie peut être faite entre un *partner link* et un canal à travers lequel une conversation téléphonique a lieu entre deux partenaires.

Chaque processus BPEL possède au moins un lien partenaire « client », car c'est le client qui invoque le processus. Un processus BPEL doit également avoir au moins un (généralement plus car il s'agit de composition de services Web) lien partenaire « invoqué ».

Chaque lien partenaire est typé par un *partnerLinkType* qui est chargé de définir le rôle que joue chacun des deux partenaires dans une conversation.

3.3.2 Variables

Les variables sont utilisées pour stocker les messages échangés entre le processus BPEL et ses partenaires ou pour stocker les informations relatives à l'état du processus. Afin de spécifier une variable, il faut spécifier son nom (*name*) et son type (*type*). Le type est représenté par l'un des trois attributs :

- *messageType :* dans le cas où la variable stocke un message WSDL.
- *element :* Dans le cas où la variable stocke un élément (type complexe) du schéma XML.
- *type :* Dans le cas où la variable stocke un type simple du schéma XML.

3.3.3 Processus BPEL

Un processus BPEL est représenté par des étapes. Chaque étape est appelée activité.

Les activités basiques sont :

- Invoquer des services Web (<*invoke*>)
- Attendre qu'un client envoie une requête au processus (<*receive*>)
- Générer une réponse pour les opérations synchrones (<*reply*>)
- Manipuler les variables (<*assign*>)
- Réagir aux erreurs (<*throw*>)
- Attendre (<*wait*>) et terminer tout le processus (<*terminate*>)

Les activités complexes peuvent être associées aux activités simples. Les activités complexes les plus importantes sont l'appel d'un ensemble d'activités en séquence (<*sequence*>), l'appel d'un ensemble d'activités en parallèle (<*flow*>), le branchement conditionnel (<*switch*>) et les boucles (<*while*>).

3.3.4 Exemple

Pour mieux voir à quoi ressemble un processus BPEL, l'exemple qui suit représente une description WSDL étendue (qui ne contient que la partie abstraite du WSDL et la définition des *partnerLinkType*) et le processus BPEL simple lui correspondant. Il s'agit d'un utilisateur, muni d'un PDA, qui souhaite récupérer une image avec un titre incrusté écrit en anglais. Il fournit en entrée l'URL d'une scène multimédia composée d'une image et d'un texte représentant un titre en français. L'image étant trop grande, une opération de réduction est nécessaire. Le texte doit être traduit vers l'anglais puis incrusté dans l'image réduite. Ainsi, ce scénario fait intervenir trois adaptateurs :

- Réducteur d'image de moitié : il prend la référence de l'image et rend l'URL de l'image réduite.
- Traducteur de titre de français à anglais: il prend le titre en paramètre et rend le titre traduit.
- Incrustateur : il prend une image et un texte en entrée et rend une image avec le texte incrusté.

Pour des raisons de simplicité et de lisibilité, nous regroupons les descriptions WSDL (abstraites) des adaptateurs ainsi que les déclarations des liens partenaires au sein du même fichier (en réalité, trois fichiers WSDL étendus sont nécessaires). Nous présentons par la suite le processus BPEL qui orchestre ces services Web adaptateurs. La figure qui suit schématise le processus BPEL ainsi que les liens partenaires correspondants.

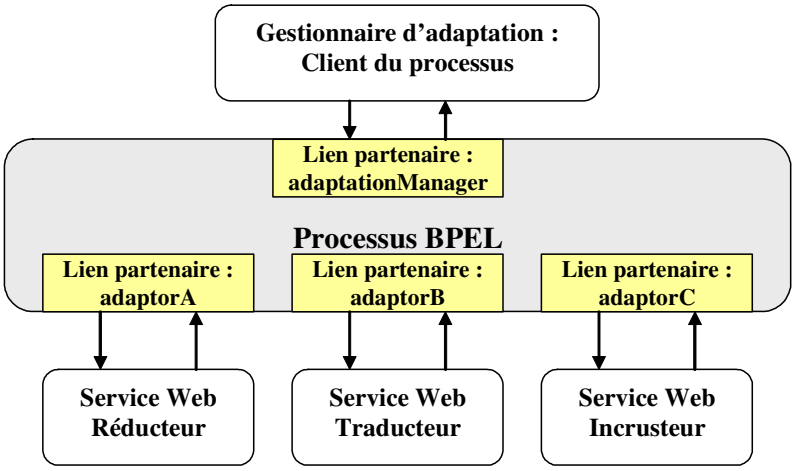

Figure 40 : interactions entre le processus BPEL de composition d'adaptateurs et ses liens partenaires

Nous commençons tout d'abord par donner les espaces de nommage relatifs aux descriptions WSDL étendues :

```
<wsdl:definitions
  targetNamespace=" http://www.paam.com/wsdl/adapt"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:plnk=http://docs.oasis-open.org/wsbpel/2.0/plnktype
  xmlns:tns="http://www.perso.enst.fr/~kazi/adaptor_000001.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

Ensuite vient la déclaration des messages envoyés entre les services Web et le processus de composition :

```
<wsdl:message name="UserParameters">
  <wsdl:part name="IMAGEURL" type="xsd:string"/>
  <wsdl:part name="TEXTURL" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="ImageURL">
  <wsdl:part name="IMAGEURL" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="TextURL">
  <wsdl:part name="TEXTURL" type="xsd:string"/>
</wsdl:message>
```

```
<wsdl:message name="AdaptedImageURL">
  <wsdl:part name="ADAPTEDIMAGEURL" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="TranslatedTextURL">
  <wsdl:part name="TRANSLATEDTEXTURL" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="AdaptedParametersURL">
  <wsdl:part name="ADAPTEDIMAGEURL" type="xsd:string"/>
  <wsdl:part name="TRANSLATEDTEXTURL" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="AdaptedMSURL">
 <wsdl:part name="ADAPTEDMSURL" type="xsd:string"/>
</wsdl:message>
```

Ensuite vient la déclaration des portTypes utilisés :

```
  <!-- portTypes supportés par le processus de composition des adaptateurs  -->
  <wsdl:portType name="AdaptationMultimediaScenePT">
    <wsdl:operation name="sendURLs">
      <wsdl:input message="UserParameters"/>
      <wsdl:output message="AdaptedMSURL"/>
      </wsdl:operation>
  </wsdl:portType>
 <wsdl:portType name="ReduceImageCallbackPT">
    <wsdl:operation name="sendAdaptationResult">
     <wsdl:input message="AdaptedImageURL"/>
    </wsdl:operation>
  </wsdl:portType>
 <wsdl:portType name="TranslateTextCallbackPT">
    <wsdl:operation name="sendAdaptationResult">
     <wsdl:input message="TranslatedTextURL"/>
    </wsdl:operation>
  </wsdl:portType>
 <wsdl:portType name="IncrustCallbackPT">
    <wsdl:operation name="sendAdaptationResult">
     <wsdl:input message="AdaptedMSURL"/>
    </wsdl:operation>
  </wsdl:portType>
 <!-- portTypes supportés par le service Web Réducteur, Traducteur et Incrusteur. -->
<wsdl:portType name="ReduceImagePT">
    <wsdl:operation name="reduce">
      <wsdl:input message="ImageURL" />
     </wsdl:operation>
```

```
</wsdl:portType>
<wsdl:portType name="TranslateTextPT">
    <wsdl:operation name="translate">
      <wsdl:input message="TextURL "/>
    </wsdl:operation>
  </wsdl:portType>
<wsdl:portType name="IncrustPT">
    <wsdl:operation name="incrust">
      <wsdl:input message="AdaptedParametersURL"/>
    </wsdl:operation>
</wsdl:portType>
```

Enfin, pour compléter cette description WSDL étendue, nous déclarons les partnerLinkType que nous utiliserons par la suite dans la définition du processus :

```
 <!-- le partnerLinkType qui caractérise le lien partenaire entre le client et le processus BPEL -->
 <plnk:partnerLinkType name="graphExecutionLT">
    <plnk:role name="callAdaptorsService"
      portType="AdaptationMultimediaScenePT" />
 </plnk:partnerLinkType>

<!-- les partnerLinkTypes qui caractérisent les liens partenaires entre le processus BPEL et les adaptateurs-->
<plnk:partnerLinkType name="ReduceImageLT">
    <plnk:role name="reducingService"
      portType="ReduceImagePT" />
    <plnk:role name="reducingRequestor"
      portType="ReduceImageCallbackPT" />
  </plnk:partnerLinkType>

<plnk:partnerLinkType name="TranslateTextLT">
    <plnk:role name="translatingService"
      portType="TranslateTextPT" />
    <plnk:role name="translatingRequestor"
      portType="TranslateTextCallbackPT" />
</plnk:partnerLinkType>

<plnk:partnerLinkType name="IncrustLT">
    <plnk:role name="incrustingService"
      portType="IncrustPT" />
    <plnk:role name="incrustingRequestor"
      portType="IncrustCallbackPT" />
</plnk:partnerLinkType>
```

```
</wsdl:definitions>
```

Passons maintenant à la définition du processus de composition proprement dit. Là aussi, nous commentons étape par étape, en commençant par les espaces de nommage :

```
<?xml version="1.0" encoding="utf-8"?>
<process name="AdaptorsCompositionProcess"
targetNamespace="http://exemple.com/bpel/example/"
xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
```

Suit la déclaration des liens partenaires. À noter que l'exemple de ce processus BPEL introduit quatre partenaires : l'utilisateur (« user »), le réducteur (« reducer »), le traducteur (« translator ») et l'incrusteur (« incrustor »).

« myRole » représente le rôle du processus dans la relation avec le partenaire. Par exemple « reducingRequestor » représente le rôle du processus BPEL dans sa relation avec le service Web partenaire « Reducer ». Le processus BPEL implémente ainsi le *portType* « ReduceImageCallbackPT », ce qui signifie que le processus implémente une méthode qui sera appelée par le service Web Reducer afin que ce dernier envoie l'URL de l'image réduite au processus (appel en *callback*).

```
<partnerLinks>
<partnerLink name="user" partnerLinkType="graphExecutionLT" myRole="callAdaptorsService"/>
<partnerLink name="Reducer" partnerLinkType="ReduceImageLT" myRole="reducingRequestor"
partnerRole=" reducingService"/>
<partnerLink name="Translator" partnerLinkType="TranslateTextLT" myRole=" translatingRequestor"
partnerRole="translatingService"/>
<partnerLink name="Incrustor" partnerLinkType="IncrustLT" myRole="incrustingRequestor"
partnerRole="incrustingService"/>
</partnerLinks>
```

Nous déclarons ensuite les variables pour stocker les différentes requêtes / réponses échangées entre les partenaires.

```
 <variables>
<!-- entrée du processus BPEL -->
<variable name="userparameters"    messageType="UserParameters"/>
<!-- entrée des adaptateurs -->
<variable name="imageURL"    messageType="ImageURL"/>
<variable name="textURL"    messageType="TextURL"/>
<variable name="adaptedparametersURL"    messageType="AdaptedParametersURL"/>
<!-- sortie des adaptateurs -->
<variable name="adaptedimageURL"   messageType="AdaptedImageURL"/>
<variable name="translatedtextURL"   messageType="TranslatedTextURL"/>
<!-- sortie du processus BPEL -->
<variable name="adaptedmsURL"   messageType="AdaptedMSURL"/>
</variables>
```

Enfin arrive la spécification des étapes du processus. Tout d'abord, le processus BPEL attend la requête initiale venant de l'utilisateur puis invoque les trois services Web en respectant l'ordre d'appel de ces adaptateurs.

```
<sequence> < !--décrit le séquencement entre l'appel de l'utilisateur et la réponse du processus-->
<!-- Recevoir la requête initiale de l'utilisateur->
<receive partnerLink="user" portType="AdaptationMultimediaScenePT"
operation="sendURLs"  variable="userparameters"  createInstance="yes" />
<!--createInstance= "yes"  signifie qu'il s'agit d'une nouvelle instance du processus-->

  <sequence> < !-- l'incrusteur est appelé en séquence des autres adaptateurs sont appelés en séquence-->
     <flow>  <!-- Invocations parallèles du réducteur et du traducteur-->
          <assign> <copy>
          <!--Récupérer l'entrée du réducteur depuis l'entrée du processus-->
          <from>$userparameters.IMAGEURL</from>
             <to>$imageURL</to>
           </copy></assign>
          <invoke partnerLink="Reducer"  portType="ReduceImagePT" operation="reduce"
          inputVariable="imageURL" />
          <receive partnerLink="Reducer"  portType="ReduceImageCallbackPT"
          operation="sendAdaptationResult"  inputVariable="adaptedimageURL"/>

          <!-- Récupérer l'entrée du réducteur depuis l'entrée du processus-->
          <assign><copy>
             <from>$userparameters.TEXTURL</from>
             <to>$textURL</to>
           </copy></assign>
          <invoke partnerLink="Translator"  portType="TranslateTextPT" operation="translate"
          inputVariable="textURL" />
          <receive partnerLink="Translator"  portType="TranslateTextCallbackPT"
          operation="sendAdaptationResult"  inputVariable="translatedtextURL"/>
    </flow>
          <assign><copy>
          <!--Concaténer les variables résultant des deux adaptateurs afin d'obtenir la variable d'entrée de
          l'inscrusteur-->
             <from>*[name()=$adaptedimageURL]/*[name()=$translatedtextURL]
            </from>
             <to> $adaptedparametersURL</to>
           </copy></assign>
          <invoke partnerLink="Incrustor"  portType="IncrustPT" operation="incrust"
          inputVariable="adaptedparametersURL" />
          <receive partnerLink="Incrustor"  portType="IncrustCallbackPT"
          operation="sendAdaptationResult"  inputVariable="adaptedmsURL"/>
```

```
  </sequence>
<!--Envoyer une réponse à l'utilisateur-->
<reply partnerLink="user" portType="AdaptationMultimediaScenePT"
operation="sendURLs"  variable="userparameters" variable="adaptedmsURL"/>

</sequence>
</process>
```

3.4 Serveurs BPEL

Un serveur BPEL fournit un environnement d'exécution aux processus d'entreprise BPEL. BPEL est très relié aux services Web et aux plates-formes qui les supportent telles que J2EE ou .Net de Microsoft.

Plusieurs serveurs commerciaux ont vu le jour tels que :

- Oracle BPEL Process Manager :

  http://www.oracle.com/technology/products/ias/bpel/index.html

- Microsoft BizTalk : http://microsoft.com/biztalk/

Il existe également des serveurs BPEL open source tels que :

- ActiveBPEL Engine : http://activebpel.org
- Apache Agila précedemment connu sous le nom de Twister :

  http://wiki.apache.org/agila/

4 Extensibilité, modularité et passage à l'échelle (scalabilité) avec les services Web

Nous avons vu dans les sections précédentes que les services Web sont auto-descriptifs et offrent des mécanismes d'annonce, de recherche et de découverte de services grâce à WSDL, SOAP et UDDI. Bien que la technologie des services Web soit similaires à ses prédécesseurs (par exemple les systèmes à objets répartis tels que Corba [Corba], plusieurs aspects les différencient.

Les services Web représentent la première technologie distribuée supportée par la plupart des vendeurs de logiciels et permettent une interopérabilité universelle entre des applications déployées sur des plates-formes différentes. Ainsi, de plus en plus d'entreprises préconisent l'utilisation des services Web afin d'implémenter leurs applications métier afin qu'elles gagnent en extensibilité, en évolutivité, en facilité d'intégration, et en réutilisabilité.

Dans le cadre de PAAM, les caractéristiques des services Web permettent :

- De développer et de déployer des systèmes d'adaptation PAAM sur plusieurs plate-formes, éventuellement différentes, en garantissant une communication facile et standard entre les différents systèmes d'adaptation PAAM.
- Aux adaptateurs utilisés par PAAM de s'annoncer, d'être recherchés, d'être découverts et de dialoguer avec le système d'adaptation PAAM via des interfaces communes. Cela n'empêche pas ces adaptateurs d'être implémentés avec des langages différents et déployés sur des plates-formes différentes.
- Le passage à l'échelle de PAAM. En effet, compte tenu des deux points précédents, l'utilisation des services Web pour implémenter les systèmes d'adaptation PAAM ou les adaptateurs facilite leur déploiement à large échelle.

Nous verrons par la suite que les services Web offrent des fonctionnalités spécifiques telles que l'envoi de messages synchrones et asynchrones, de la gestion de la continuité de session, deux fonctionnalités nécessaires à PAAM pour qu'elle puisse gérer la dynamicité et la tolérance aux disparitions des adaptateurs. Nous invitons le lecteur à regarder l'annexe 2 qui résume les caractéristiques que doivent respecter les services Web d'adaptation afin d'être conformes aux spécifications de PAAM.

5 Conclusion

Dans ce chapitre, nous avons donné un aperçu des standards de services Web permettant notamment de décrire, d'annoncer, de découvrir, de composer et d'orchestrer les services Web.

Avant de présenter notre prototype PAAM implémenté à l'aide des services Web, nous présentons, dans le chapitre suivant, notre contribution concernant la gestion des adaptateurs distribués et la gestion de la dynamicité.

Chapitre 5

Gestion des adaptateurs distribués et de la dynamicité

1 Introduction

Ce chapitre présente la contribution concernant la description et l'identification de manière unique d'une ressource d'adaptation (ou adaptateur). Nous présentons également la contribution concernant la description sémantique d'un service d'adaptation pour décrire, entre autres, ses paramètres d'entrée et de sortie.

Nous exposons par la suite le protocole de négociation et d'acceptation établi entre un gestionnaire d'adaptation et un adaptateur. Ce protocole permet au gestionnaire d'adaptation de choisir un adaptateur parmi d'autres.

Enfin, PAAM gérant l'adaptation distribuée sur différents nœuds du réseau, susceptibles de se déconnecter à chaque instant, nous proposons des solutions pour gérer les déconnexions dans PAAM afin de lui procurer un aspect dynamique.

2 Catégorisation et identification des adaptateurs

L'avènement des architectures orientées composants et services a inspiré des travaux sur des langages de description de services. IDL, SDL, OWL ou encore OWL-S ont tous été proposés par des organismes de normalisation dans le but de décrire un service, ses interfaces et/ou sa sémantique [IDL] [SDL] [OWL] [OWLS]. Une de nos contributions est la proposition d'un moyen unique d'identifier un adaptateur multimédia. Pour ce faire, il a d'abord fallu répertorier les classes principales des adaptateurs en catégories. Nous présentons cette classification dans la section qui suit.

2.1 Catégories d'adaptateurs

Nous distinguons trois catégories principales d'adaptation : le transmodage, le transcodage, et la transformation.

Nous rappelons les définitions des classes d'adaptateurs, précédemment présentées dans le paragraphe 3.6 du chapitre 2 :

- Le ***transmodage*** est le changement de la modalité d'un média. Il s'agit, par exemple, de transformer un texte en image pour un client qui ne dispose pas de la police de caractère permettant l'affichage de ce texte.
- Le ***transcodage*** consiste à changer le format d'un média donné, par exemple, en passant du format vidéo MOV au format vidéo AVI.
- La ***transformation*** d'un média donné ne change ni sa modalité ni son format. Ce processus transforme un contenu en réduisant sa taille par exemple. D'autres exemples d'adaptateurs tels que l'incrustation de sous-titres dans une vidéo font également partie de cette famille d'adaptateurs. Nous pouvons également considérer des adaptateurs spécifiques qui reconstruisent un document multimédia de synchronisation de type SMIL ou MPEG-21 par exemple. Ils prennent en entrée un document multimédia et le **transforment** en un autre document multimédia.

2.2 Identification uniforme des noms des adaptateurs

Afin d'assurer des fonctions de recherche et de sélection d'adaptateurs de médias, il est nécessaire d'associer une sémantique précise à un ensemble d'adaptateurs. Une méthode efficace consiste à établir une ontologie de ce domaine. A titre d'ébauche de cette ontologie, nous proposons ci-dessous la typologie des adaptateurs qui nous sert dans l'implémentation actuelle de PAAM.

Nous utilisons pour cela la notation URN (Uniform Resource Names) [URN]. Cette notation permet d'identifier une ressource indépendamment de sa localisation et d'une façon qui décrit une partie de ses caractéristiques. Ainsi, *« urn:paam:transcoder:video:video: videoFormatConverter »* décrit un processus d'adaptation *« videoFormatConverter »* qui réalise un transcodage vidéo d'un format vers un autre.

L'utilisation des URN facilite la recherche des adaptateurs en fonction de ce qu'ils réalisent comme opérations d'adaptation. Notons que la recherche peut se faire sur un seul critère tel que la catégorie d'adaptation ou sur plusieurs critères tels que la catégorie d'adaptation et les formats d'entrée et de sortie pris en compte.

Nous présentons maintenant quelques exemples d'identification URN de ressources d'adaptation. Les deux premiers champs de l'URN signifient qu'il s'agit d'un URN relatif à l'architecture PAAM. Le troisième et le quatrième champ représentent respectivement la modalité d'entrée et la modalité de

sortie prises en compte par l'adaptateur. Le dernier champ correspond au nom mnémonique que nous avons attribué aux différents processus d'adaptation.

Le tableau suivant résume quelques identifications URN des adaptateurs que nous avons recensés :

Identification URN	Description
Transcodage	
urn:paam:transcoder:video:video:videoFormatConverter	Convertit une vidéo d'un format vers un autre
urn:paam:transcoder:image:image:imageFormatConverter	Convertit une image d'un format vers un autre
urn:paam:transcoder:audio:audio:audioFormatConverter	Convertit un contenu audio d'un format vers un autre
urn:paam:transcoder:text:text:textFormatConverter	Convertit un texte d'un format vers un autre
Transmodage	
urn:paam:transmoder:image:audio:imageReader	Convertit une image vers un contenu audio
urn:paam:transmoder:image:text:alternateTextProducer	Convertit une image vers un texte
urn:paam:transmoder:video:image:snapshot	Prélève une image d'une vidéo
urn:paam:transmoder:video:slideshow:snapshotSelection	Prélève une succession d'images d'une vidéo
urn:paam:transmoder:video:text:alternateTextProducer	Convertit une vidéo vers un texte
urn:paam:transmoder:video:audio:audioExtractor	Extrait un contenu audio d'un contenu audio visuel
urn:paam:transmoder:video:video:videoExtractor	Extrait une vidéo d'un contenu audio visuel
urn:paam:transmoder:text:image:text2Image	Convertit un texte vers une image (SVG par ex.)
urn:paam:transmoder:text:slideshow:text2Slides	Convertit un texte vers une suite d'images
urn:paam:transmoder:text:audio:textReader	Convertit un texte vers un contenu audio
urn:paam:transmoder:audio:text:audioTranscription	Convertit un son vers un texte
urn:paam:transmoder:audio:text:alternateTextProducer	Convertit un son vers un texte
Transformation	
urn:paam:transformer:video:video:videoResizer	Retaille une vidéo
urn:paam:transformer:video:video:colorRestorer	Restaure la couleur d'une vidéo
urn:paam:transformer:video:video:colorAdjuster	Règle la couleur d'une vidéo
urn:paam:transformer:video:video:blurer	Rend une vidéo floue
urn:paam:transformer:video:video:noiseReducer	Réduit le bruit d'une vidéo
urn:paam:transformer:video:video:colorCurveAdjuster	Règle la courbe de couleur d'une vidéo
urn:paam:transformer:video:video:sharpener	Affûte la qualité d'une vidéo
urn:paam:transformer:video:video:fader	Affaiblisseur d'intensité de lumière d'une vidéo
urn:paam:transformer:video:video:watermarker	Insérer des données cachées sur la vidéo
urn:paam:transformer:video:video:frameRateModifier	Modifier le nombre d'images par seconde

	d'une vidéo
urn:paam:transformer:video:video:videoCompressor	Compresse une vidéo
urn:paam:transformer:video:video:videoAbstract	Remplace la vidéo par son résumé
urn:paam:transformer:video:videoSpatialResolutionChanger	Change la résolution d'une vidéo
urn:paam:transformer:audio:audio:quantifier	Fait correspondre une valeur discrète à l'amplitude d'un échantillon de voix par rapport à des valeurs étalons appelées niveaux de quantification.
urn:paam:transformer:audio:audio:audioCompressor	Compresse un audio
urn:paam:transformer:audio:audio:audioAbstract	Remplace un contenu audio par son résumé
urn:paam:transformer:image:image:imageResizer	Retaille une image
urn:paam:transformer:image:image:imageCompressor	Compresse une image
urn:paam:transformer:image:image: spatialResolutionChanger	Change la résolution d'une image
urn:paam:transformer:text:text:Translation	Traduit du texte
urn:paam:transformer:text:text:textAbstract	Remplace le texte par son résumé
urn:paam:transformer:text:text:spaceCleaner	Enlève des espaces d'un contenu textuel
urn:paam:transformer:text:text:textSizeChanger	Retaille la police du texte
urn:paam:transformer:multimedia:multimedia:remap	Reconstruit un document multimédia

Tableau 9 : Tableau représentant les identifications URN des adaptateurs

Notons que d'autres médias tels que Graphique2D, Graphique2D animé, Graphique3D, Graphique3D animé, texte enrichi, texte avec timing, son naturel synthétique, son naturel 3D peuvent être pris en compte par ce type de codage. Notons également que certaines adaptations telles que la transformation d'un document multimédia PowerPoint en une page html sont également à considérer.

Après avoir proposé notre moyen d'identifier les adaptateurs de manière unique, il nous faut maintenant trouver un moyen pour décrire un adaptateur sémantiquement, c'est-à-dire une description qui contient des informations telles que son URN et ses paramètres d'entrée. La section qui suit détaille cette description.

2.3 Description sémantique d'un service Web d'adaptation

Dans la logique PAAM, un service web d'adaptation constitue l'entité qui adapte les médias en les récupérant depuis le fournisseur de contenus.

Nous rappelons dans ce qui suit le scénario d'adaptation de base de PAAM :

Après analyse des descriptions du contexte et du document multimédia composé, le planificateur de PAAM prend des décisions d'adaptation et construit un graphe d'adaptation logique. Ce graphe est transmis au gestionnaire d'adaptation qui doit rechercher les adaptateurs répondant aux besoins fixés par le planificateur.

Ce scénario ne peut aboutir que si les ressources d'adaptation sont bien décrites. Plus précisément, conformément à la logique PAAM, le gestionnaire d'adaptation doit chercher les adaptateurs :

- possédant la bonne identification URN,
- acceptant en entrée le mimetype du média source indiqué par le gestionnaire d'adaptation,
- fournissant en sortie le mimetype attendu par le gestionnaire d'adaptation,
- gratuits (si l'utilisateur ne veut pas payer), et
- au meilleur prix (si l'utilisateur accepte de payer).

Nous optons pour l'utilisation de WSDL car il décrit les interfaces d'un service Web et définit comment accéder à ce service. WSDL informe également sur la localisation du service et sur le protocole à utiliser pour le transport des messages.

Cependant, nous avons mentionné la nécessité pour un gestionnaire d'adaptation d'avoir des informations supplémentaires relatives à l'identification de la catégorie du service d'adaptation et aux mimetypes d'entrée et de sortie supportés par cet adaptateur.

Nous proposons d'étendre la description WSDL d'un service Web d'adaptation en insérant des balises additionnelles pour qu'un adaptateur puisse faire partie du système d'adaptation distribué de PAAM et puisse être analysé par le gestionnaire d'adaptation.

La Figure 41 est un exemple d'une description PAAM partielle d'un service d'adaptation.

```
<definitions …..>
<paam:PAAM xmlns:paam="http://www.perso.enst.fr/~kazi/paam.xsd">

<paam:urn>
        urn:paam:transmoder:video:audio:audioExtractor
</paam:urn>

<paam:input_mimetypes>
        video/mpeg;video/quicktime
</paam:input_mimetypes>

<paam:output_mimetypes>
        audio/mpeg
</paam:output_mimetypes>

<paam:price>
        0
</paam:price>

<paam:Adaptor_URL>
        http://137.194.160.42:8080/audioExtractor
</paam:Adaptor_URL>

</paam:PAAM>

<!-- Corps du WSDL-->
...
```

```
</definitions>
```

Figure 41: Description PAAM du service d'adaptation audioExtractor

L'expression des contraintes impliquées par cette description peut paraître simpliste. En effet, les mimetypes d'entrée, de sortie ainsi que l'identification URN peuvent, dans certains cas, ne pas être suffisant pour décrire toutes les caractéristiques d'un adaptateur. Par exemple, un adaptateur peut vouloir dire qu'il ne sait prendre en compte que des images dont les dimensions ne dépassent pas 1600 x 1200 pixels. Ainsi, certaines solutions préconisent l'utilisation de OWL afin de donner plus de sémantique à une telle description. Dans le cadre de PAAM, nous optons pour la description présentée en Figure 41 et nous exprimons plus de contraintes grâce à la politique d'acceptation de l'adaptateur que nous détaillons un peu plus loin.

2.4 La liste temporaire des descriptions des adaptateurs

La liste des adaptateurs résulte de l'analyse des descriptions sémantiques que nous avons présentées dans la section précédente. Chaque adaptateur souhaitant participer à la logique d'adaptation de PAAM doit fournir une description PAAM et la stocker dans un dépôt distant connu de PAAM qui pourrait, par exemple, être un annuaire UDDI, une base de données locale ou distante, distribuée ou pas, ou tout autre type de dépôt (voir Annexe 2).

Au moment du déploiement du système d'adaptation PAAM et avant toute requête utilisateur, le gestionnaire d'adaptation analyse et constitue la liste. Il en récupère les informations suivantes :

- L'identification « URN » de l'adaptateur : ce champ correspond au contenu de l'élément « paam:urn ». Dans l'exemple de la Figure 41, cela équivaut ***urn:paam:transmoder:video:audio:audioExtractor***
- Les mimetypes d'entrée supportés : ce champ correspond au contenu de l'élément « paam:input_mimetypes ». Dans l'exemple de la Figure 41, cela équivaut ***video/mpeg;video/quicktime***
- Les mimetypes de sortie supportés : ce champ correspond au contenu de l'élément « paam:outputput_mimetypes ». Dans l'exemple de la Figure 41, cela équivaut ***audio/mpeg***
- Le prix du service d'adaptation : ce champ correspond au contenu de l'élément « paam:price ». Dans l'exemple de la Figure 41, cela équivaut à 0 € (service gratuit).
- L'URL de l'adaptateur : Ce champ correspond au contenu de l'élément « paam:Adaptor_URL ». Dans l'exemple de la Figure 41, cela équivaut à ***http://localhost:8080/audioExtractor***

3 Choix d'un adaptateur

Nous avons, dans le chapitre 3, présenté un scénario de bout en bout, concernant l'adaptation d'un document multimédia composé. Nous avons vu qu'après les phases d'analyse du contexte et du document, de prise de décision et de construction du graphe d'adaptation logique, le gestionnaire d'adaptation choisit, dans la liste temporaire des adaptateurs, le « bon » adaptateur dans le cas où plusieurs d'adaptateurs répondent à ses demandes. Nous explorons dans ce qui suit quelques solutions au problème de choix du meilleur adaptateur.

Une des solutions est de gérer de manière répétitive la liste temporaire des adaptateurs de sorte qu'elle soit triée suivant la charge et la disponibilité de chaque adaptateur. Le gestionnaire de la liste envoie à des intervalles réguliers des messages *« getState »*[7] à tous les adaptateurs de la liste et trie ainsi cette liste. Bien que cette solution soit facile à réaliser, nous l'avons vite abandonnée car elle génère un trafic de messages important.

Une autre solution est de ne pas envoyer des messages *« getState »* à tous les adaptateurs, mais seulement à ceux qui sont présents dans la liste des adaptateurs et qui, lors de la construction du graphe d'adaptation physique, répondent en plus aux premiers critères de choix (URN, mimetypes d'entrée et mimetypes de sortie). Cette solution est une solution acceptable lorsque le nombre d'adaptateurs concernés n'est pas important (le cas des adaptateurs rares) mais devient aussi gourmande en bande passante que la première solution dans le cas où le nombre d'adaptateurs concernés est élevé (cas des adaptateurs courants tels que les transcodeurs d'images).

Une dernière solution consiste à n'envoyer des messages *« getState »* qu'à une partie des adaptateurs candidats. Le nombre de ces adaptateurs peut varier suivant l'heuristique que l'on utilise. Par exemple, si l'heuristique fixe le nombre d'adaptateurs à 1, le gestionnaire d'adaptation choisit un adaptateur au hasard parmi la sous liste des adaptateurs candidats et leur envoie le message *« getState »*. L'opération est répétée jusqu'à ce que la charge de l'adaptateur lui permette de réaliser l'adaptation et que la durée de l'adaptation estimée par l'adaptateur soit acceptable par le gestionnaire d'adaptation. Cette solution réduit de manière significative le nombre de messages envoyés et par conséquent allège le trafic réseau et évite les goulots d'étranglement au niveau du système d'adaptation PAAM. Cette solution suggère également l'implémentation d'un protocole de négociation et d'acceptation entre le gestionnaire d'adaptation et l'adaptateur que nous introduisons dans ce qui suit.

3.1 Définition

Une application qui désire utiliser le réseau pour transporter du trafic doit d'abord demander une connexion au réseau en l'informant sur ses besoins en matière de trafic et de qualité de service. Ces informations sont stockées dans un « contrat de trafic ». Le réseau juge s'il possède assez de ressources disponibles pour accepter la connexion, et ainsi accepte ou refuse la requête de connexion. Ceci est connu sous l'appellation de « contrôle d'admission ».

7 getState est une méthode implémentée par chaque adaptateur servant à connaître son état (par ex. sa charge CPU, mémoire, etc.)

Le contrôle d'admission dans le cadre de PAAM est également une négociation entre deux entités : le gestionnaire d'adaptation et l'adaptateur. Il ne s'agit pas de négocier une connexion Internet mais plutôt de choisir l'adaptateur en fonction de sa disponibilité et de ses capacités.

3.2 Protocole de négociation et d'acceptation de PAAM

Le protocole de négociation et d'acceptation facilite le choix d'un adaptateur par le gestionnaire d'adaptation. Il s'agit d'un mécanisme de négociation dynamique entre le gestionnaire d'adaptation et un adaptateur donné, préalablement choisi dans la liste temporaire des adaptateurs. Cette négociation suppose que les adaptateurs fonctionnent de manière asynchrone afin de pouvoir dialoguer avec le gestionnaire d'adaptation pendant le processus d'adaptation.

La Figure 42 illustre ce protocole :

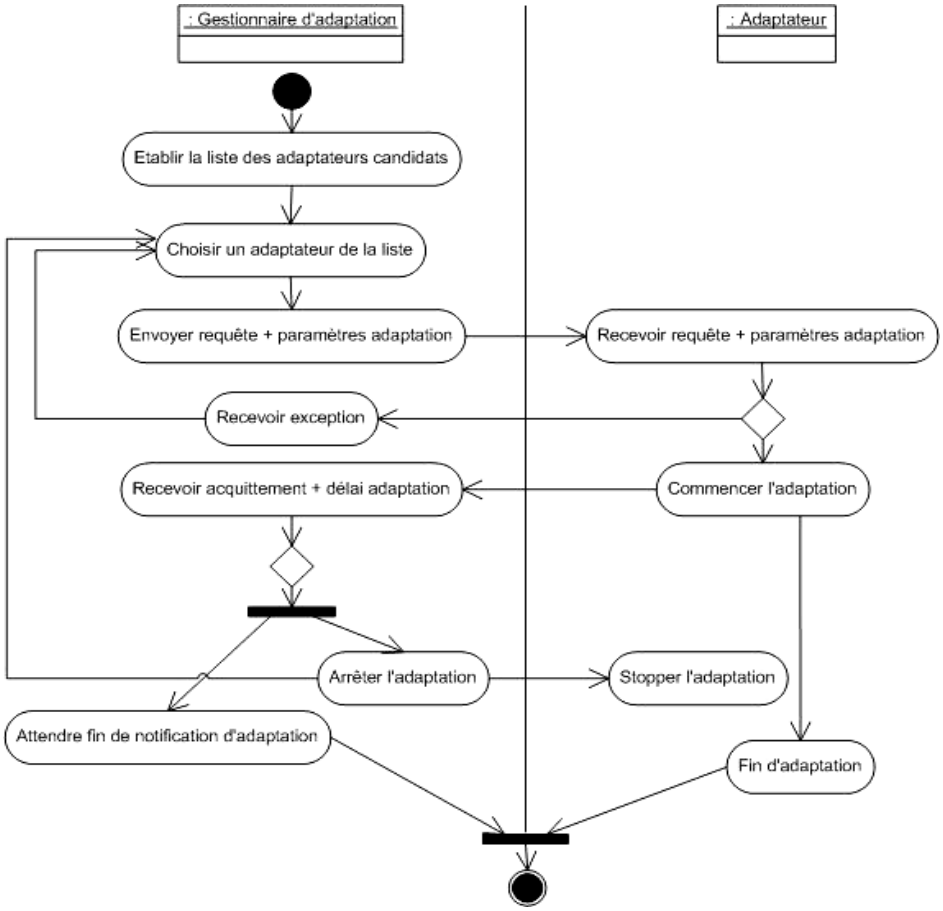

Figure 42 : Protocole de négociation et d'acceptation

Nous présentons un scénario simple résumant les étapes du protocole de négociation et d'acceptation schématisé dans la figure ci-dessus :

1. Le gestionnaire d'adaptation effectue un appel d'un service d'adaptation
2. Soit le service Web d'adaptation refuse le traitement et envoie une exception au gestionnaire d'adaptation ; soit le service Web d'adaptation démarre (de façon

asynchrone), il renvoie donc un message d'acquittement qui contient des informations sur la durée estimée de traitement,

3. Si le gestionnaire d'adaptation reçoit une exception, il cherche un autre adaptateur de la liste temporaire des adaptateurs.
4. Si le gestionnaire d'adaptation reçoit une réponse et qu'elle lui convient, il laisse le traitement se faire ; si la réponse ne lui convient pas, il envoie un message pour interrompre le traitement.

Ce scénario, bien que simplifié, introduit une utilisation complexe des services Web d'adaptation. En effet, lorsque le gestionnaire d'adaptation négocie avec l'adaptateur, le message envoyé doit non seulement contenir les paramètres de négociation nécessaires à l'algorithme de prise de décision (que nous présentons dans la section qui suit) mais également les paramètres d'appel du service Web. De cette manière, nous évitons d'envoyer un nouveau message dans le cas où la réponse de l'adaptateur est positive.

Nous formalisons dans ce qui suit le protocole de négociation et d'acceptation :

Début

1. Établir liste adaptateurs candidats tels que
   - urn_adaptateur = = urn_recherché
   - input_adaptateur_recherché = = input_adaptateur
   - output_adaptateur_recherché = = output_adaptateur
2. N = heuristique_choix ( ) : *détermine le nombre d'adaptateurs candidats avec lesquels on applique le protocole de négociation et d'acceptation. Par défaut, N = 1.*
3. Pour i allant de 1 à N Faire :
   - Gestionnaire_Adaptation.Envoyer_requête (paramètres_adaptation, paramètres_protocole) ➔ Adaptateur (i)
   - Résultat = politique_acceptation (Adaptateur (i)) : détermine si l'adaptateur peut ou non réaliser l'adaptation.
   - **Si** (Résultat = = False) **Alors** { Adaptateur(i).Envoyer_réponse (exception) ➔ Gestionnaire_Adaptation }
   - **Si** (i < N) **Alors {** i = i + 1 **&** Aller à 3 } : entamer protocole de négociation avec un autre adaptateur
   - **Sinon** Aller à 4 : le gestionnaire d'adaptation a testé tous les adaptateurs de la liste, il faut en selectionner N autres.
   - **Si** (Résultat = = True) **Alors** { Adaptateur(i).Envoyer_réponse (ACK , Délai) ➔ Gestionnaire_Adaptation **&** Aller à 5 }
4. Rafraîchir_Liste & Aller à 3 : *consiste à enlever de la liste des adaptateurs les N adaptateurs qui ne peuvent pas réaliser l'adaptation cible*
5. **Si** (Délai = = acceptable) **Alors** Aller à Fin
   - **Sinon Si** (i < N) **Alors** { i = i + 1 **&** Gestionnaire_Adaptation.Envoyer_requête (Stopper(adaptateur(i))) **&** Aller à 3 }
   - **Sinon** Aller à 4

Fin

Les paramètres nécessaires à l'algorithme de prise de décision sont de deux types : des paramètres relatifs au média et des paramètres relatifs au fournisseur du média. Ces paramètres sont :

- Les dimensions d'une vidéo ou d'une image (largeur et hauteur en pixels).
- La taille d'une vidéo, d'une image ou d'une bande son (en méga-octets).
- Le débit qui caractérise le fournisseur du média.

Notons que les adaptateurs qui n'offrent pas de politique d'acceptation ou qui ne sont pas implémentés de manière asynchrone ne sont pas considérés par PAAM (voir Annexe 2). Nous présentons cette politique dans ce qui suit.

3.3 Politique d'acceptation d'un adaptateur

Un adaptateur peut être implémenté sur n'importe quelle plate-forme avec n'importe quel langage de programmation. Cependant, chaque adaptateur souhaitant participer à la logique d'adaptation de PAAM doit implémenter, entre autres, une politique d'acceptation, exposé via une interface de service Web. Cette politique a pour but de déterminer si un adaptateur donné peut satisfaire ou pas une requête tout en fournissant une estimation du temps d'adaptation et éventuellement un délai du début de l'adaptation.

Chaque adaptateur implémente ainsi sa propre politique d'acceptation.

Chaque adaptateur implémente une politique d'acceptation avec les outils qu'il désire. Il peut, par exemple, considérer quelques-uns des éléments suivants :

- Le poids de l'adaptation entrante : chaque opération d'adaptation a un poids donné en fonction de la durée estimée de l'adaptation et des charges CPU et mémoire nécessaires à la réalisation de cette adaptation. Ces paramètres sont déterminés grâce à la taille du média à adapter, à la catégorie de l'adaptation et aux caractéristiques de l'adaptateur. Par exemple, un transcodage de vidéo possède un poids plus élevé qu'un re-dimensionnement d'image (image resizing) car, dans la plupart des cas, la taille d'une vidéo est plus grande que la taille d'une image et le transcodage d'une vidéo nécessite plus de ressources que le re-dimensionnement d'une image. Notons cependant que le poids de l'adaptation d'un même média peut varier suivant les caractéristiques de l'adaptateur (par ex : le langage d'implémentation et les ressources machine allouées à cet adaptateur).
- La charge actuelle de l'adaptateur : chaque adaptateur possède une capacité de traitement limitée. La charge actuelle de l'adaptateur est fonction du nombre d'adaptations en simultané. Ce paramètre devient complexe à calculer si l'adaptateur réalise différentes adaptations, auquel cas, il faut tenir compte du poids de chaque adaptation.
- La charge CPU et mémoire de la machine hébergeant l'adaptateur : un adaptateur peut avoir la totalité ou une partie des ressources CPU et mémoire d'une machine.

Cette liste, bien que non exhaustive, montre qu'il est difficile d'avoir des règles générales à appliquer par chaque adaptateur. Ainsi, chaque adaptateur peut implémenter sa propre politique d'acceptation allant des heuristiques rudimentaires à des analyses plus complexes intégrant l'utilisation d'outils mathématiques. Nous ne présentons pas de solution générique que chaque adaptateur pourrait appliquer. Nous présentons cependant des exemples de politiques d'acceptation.

3.4 Exemples de politiques d'acceptation

3.4.1 Exemple 1

Un adaptateur peut implémenter une politique d'acceptation qui ne l'autorise à adapter qu'un seul média à la fois. Ainsi, lorsque cet adaptateur reçoit une requête d'adaptation depuis le gestionnaire d'adaptation, il retourne une exception s'il est en train d'adapter un autre média, sinon, il retourne un message d'acquittement ainsi qu'une estimation de la durée d'adaptation.

Cette politique d'acceptation est simple à réaliser. Des variantes sont possibles :

- Autoriser deux adaptations en simultané.
- Autoriser une seule adaptation si et seulement si l'adaptateur ne réalise aucun autre processus (adaptation ou autre).

3.4.2 Exemple 2 : WESEMAC

Ce deuxième exemple est le résultat de tests réalisés sur un adaptateur appelé WESEMAC (voir Annexe 1), le but étant de fournir quelques estimations sur la charge d'un adaptateur particulier.

WESEMAC est un acronyme désignant un service Web multimédia d'adaptation réalisé au sein de notre groupe de recherche. Les tests ont été réalisés avec une machine dotée d'un processeur Intel Pentium M 760, avec une fréquence de 2.0 GHz dans des conditions idéales (aucun autre processus ne tourne) et 1Go de RAM.

WESEMAC permet de réaliser, entre autres, des opérations de transcodage de vidéos et d'images et de réduction d'image. Nous choisissons de tester le transcodage de vidéos. Afin de calibrer la taille des médias, nous avons évalué la taille moyenne des vidéos relatives à des séries télévisées, particulièrement populaires et disponibles en streaming. Il résulte de cette étude que la taille moyenne de ces vidéos varie entre 340 et 360 méga-octets et que le format le plus utilisé pour encoder ces vidéos est le format AVI.

Les tests réalisés montrent que WESEMAC peut traiter jusqu'à 15 adaptations en même temps lorsqu'il s'agit de transcoder une vidéo de 360 méga-octets.

Tenant compte de ces résultats et dans le cas où toutes les adaptations concernent des transcodages de vidéos de tailles variant de 340 à 360 méga-octets, la politique d'acceptation de WESEMAC pourrait être de n'accepter au maximum que 15 adaptations en simultané.

À noter que les tests réalisés avec WESEMAC sont à prendre avec précaution car ils ont été réalisés dans des conditions idéales dans lesquelles aucune autre application ne tournait et nous avons dû en tenir compte.

4 Dynamicité et gestion des déconnexions

Un système dynamique est un système qui adapte son exécution en fonction des changements perçus dans son contexte d'exécution ou dans un contexte influençant le résultat de son exécution.

Dans le cas de PAAM, le changement des caractéristiques logicielles ou matérielles d'un terminal, le changement des préférences de présentation d'un utilisateur ou la disparition d'un adaptateur sont autant d'événements susceptibles de déclencher une adaptation dynamique. La boucle d'adaptation réalisée par le système PAAM doit tenir compte de ces événements.

Nous distinguons deux types de changements qui nécessitent chacun un traitement approprié : le changement dans le contexte de l'utilisateur et le changement dans le contexte de la plate-forme des adaptateurs.

4.1 Changements dans le contexte de l'utilisateur

Le contexte de l'utilisateur est l'ensemble des informations relatives à un utilisateur et à son environnement d'exécution. Dans le cadre de PAAM, nous supposons que le contexte utilisateur est composé des préférences utilisateur d'affichage et de consommation des médias (par ex. les langues souhaitées et le format préféré d'une vidéo) et des capacités logicielles (par ex. les lecteurs multimédia disponibles) et matérielles (par ex. taille du tampon, taille de l'écran ou résolution) de son terminal.

L'utilisateur peut, par exemple, à tout moment, faire parvenir ses nouvelles préférences à son planificateur ou changer de terminal. L'utilisateur peut également opter pour un autre format vidéo ou passer d'un ordinateur portable à un PDA. Ainsi, ces événements sont déclenchés par l'utilisateur ou détectés par un des équipements de sondage et doivent être interprétés par le planificateur.

L'interaction *utilisateur / PAAM* peut être réalisée grâce à des échanges de messages entre l'utilisateur ou l'équipement de sondage et le système PAAM lorsque un événement modifie le contexte utilisateur.

L'interaction *PAAM/adaptateurs* peut également être un moyen d'informer l'utilisateur sur l'état et la durée des adaptations en cours. L'utilisateur peut alors changer ses préférences en fonction de ces informations. Par exemple, l'utilisateur peut vouloir passer à une autre modalité ou changer de format d'affichage lorsque le temps d'adaptation du média concerné est trop long.

La durée d'une adaptation ne peut être récupérée par le planificateur et transmise à l'utilisateur qu'une fois l'adaptateur instancié et lorsque le gestionnaire d'adaptation interroge l'adaptateur sur l'état (par exemple la durée) de l'adaptation en cours grâce au message « *getAdaptationState* ».

Lorsqu'un changement, quel qu'il soit, s'opère dans le contexte de l'utilisateur, le système d'adaptation PAAM doit être implémenté d'une manière à gérer les suivis de sessions. Le planificateur tient compte de ce(s) changement(s) et applique de nouveau pour le ou les média(s) concerné(s) les algorithmes de prise de décision en réévaluant les politiques d'adaptation ; il en résulte un nouveau graphe d'adaptation logique qui est transmis au gestionnaire d'adaptation et instancié. Le gestionnaire d'adaptation a alors

pour mission de libérer les ressources du ou des adaptateur(s) supprimé(s) du graphe d'adaptation et d'instancier les nouveaux adaptateurs.

4.2 Changements dans le contexte des adaptateurs

Cette deuxième catégorie de changement de contexte concerne l'état des adaptateurs. Un système où les ressources sont mises à disposition par des nœuds présents dans le réseau dans une logique P2P est fréquemment confronté aux déconnexions. Ainsi, si un nœud offrant un adaptateur disparaît, le service d'adaptation qu'il fournit disparaît avec lui ainsi que le média ou une partie de ce média adapté jusque-là.

La gestion de ce type de changement commence par la détection d'une déconnexion. Cette détection peut être réalisée de différentes façons :

- Au niveau du pair, qui, à des intervalles réguliers, envoie un message au gestionnaire d'adaptation pour donner des informations sur son état
- À l'initiative du gestionnaire d'adaptation qui envoie des messages de détection de connexions aux nœuds adaptateurs.

En résumé, la détection de la déconnexion se traduit par des échanges de messages entre le gestionnaire d'adaptation et le ou les nœuds concernés par les opérations d'adaptation (ceux qui constituent le graphe d'adaptation physique).

Dans le cas de PAAM, nous avons opté pour la deuxième solution. Le gestionnaire d'adaptation envoie à des intervalles réguliers des messages *« getState »* afin de connaître la charge d'un adaptateur, et dans le cas où le gestionnaire d'adaptation ne reçoit pas de réponse d'un adaptateur, il conclut qu'il a disparu et agit en conséquence en cherchant dans la liste un autre adaptateur.

Le nombre des messages « *getState* » ne doit être trop élevé afin de ne pas saturer le trafic réseau. Ce nombre dépend de la taille du média et du temps d'adaptation estimé. Nous verrons cela avec plus de détail dans le chapitre 7.

4.3 Tolérance aux disparitions des adaptateurs

[Gar99] définit une faute comme un état non voulu (mais cependant possible) d'un processus. Nous pouvons ainsi associer la disparition d'un nœud de la plate-forme d'adaptation de PAAM qui réalise une adaptation à la notion de faute.

La tolérance aux fautes désigne la capacité d'un système à continuer à fournir un service acceptable malgré la présence de quelques fautes [LRA04]. La tolérance aux fautes (disparitions) est indispensable dans un système comme PAAM où des nœuds distribués offrant des ressources d'adaptation peuvent à tout moment disparaître.

Les garanties que doit assurer la tolérance aux fautes dépendent du type de l'application, allant des applications critiques aux applications commerciales [BDM93]. Dans le cadre de PAAM, la tolérance

aux fautes (disparitions) doit garantir un meilleur service à valeur ajoutée en évitant les pertes d'informations et de temps.

Dans le cadre de PAAM et de son environnement distribué d'adaptateurs, tolérer les disparitions revient à mettre en place des mécanismes de sauvegardes supplémentaires et d'instanciations multiples. Dans la littérature, cela est appelé redondance.

La redondance, d'une façon générale, se rapporte à la qualité ou à l'état d'être en sur-nombre, par rapport à la normale. Ce qui peut avoir la connotation de superflu, mais aussi un sens positif quand cette redondance est voulue afin de prévenir un dysfonctionnement [8].

[Gar99] affirme que toute solution tolérante aux fautes doit proposer au moins une forme de redondance et doit implémenter un mécanisme de détection des erreurs.

Plusieurs types de redondances existent : la redondance d'information, la redondance temporelle et la redondance spatiale.

La redondance d'information consiste à dupliquer les données. Elle est souvent utilisée dans le monde des bases de données. Cette catégorie de redondance est utilisée afin de maintenir l'intégrité des données dans un système.

La redondance temporelle consiste à dupliquer les calculs et les traitements. Cette forme de redondance implique des coûts en temps pour corriger l'erreur, typiquement en relançant l'opération qui a produit l'erreur.

La redondance spatiale implique la duplication de composants matériels et/ou logiciels. Elle est utilisée pour détecter les erreurs (en comparant les résultats fournis par les composants dupliqués), corriger les fautes (par exemple en isolant le composant fautif), recouvrir les erreurs (par exemple en adoptant le résultat produit par la plupart des composants)

Notons qu'une autre notion se rapproche de la notion de la redondance spatiale. Il s'agit de la réplication.

La réplication consiste à dupliquer de manière cohérente les mêmes unités logicielles et matérielles sur plusieurs nœuds physiques ou logiques pour mettre en œuvre un aspect de la sûreté de fonctionnement.

La différence entre la redondance spatiale et la réplication est que la redondance spatiale est souvent utilisée pour gérer des défaillances graves telles que les pannes byzantines[9], tandis que la réplication est utilisée pour gérer des pannes moins graves telles que les pannes en omission[10].

La réplication des adaptateurs et/ou des adaptations correspond le mieux aux besoins de tolérances aux disparitions des adaptateurs dans PAAM.

8 http://fr.wikipedia.org/wiki/Redondance

9 Panne byzantine : tout comportement d'un système ne respectant pas ses spécifications, en donnant des résultats non conformes.

10 Panne en omission : lorsque des messages sont perdus en entrée ou en sortie ou les deux

Nous présentons dans ce qui suit le mécanisme de détection des disparitions des adaptateurs. Nous décrivons par la suite deux cas gérant les disparitions des adaptateurs. Contrairement au second cas, le premier cas ne réplique les adaptations que lorsqu'une disparition est détectée.

4.3.1 Détection des disparitions des adaptateurs

La détection des disparitions des adaptateurs implique des envois supplémentaires de messages. Hormis le protocole de négociation et d'acceptation, le gestionnaire d'adaptation doit intégrer un mécanisme de détection des adaptateurs. Ce mécanisme se traduit par l'envoi de messages, pendant les adaptations des médias, afin de vérifier l'état de l'adaptation et la présence ou non du nœud offrant l'adaptateur.

À l'issue du protocole de négociation et d'acceptation, le gestionnaire d'adaptation reçoit une estimation de la durée d'adaptation de chaque adaptateur. Afin de vérifier l'état de chaque adaptateur, le gestionnaire d'adaptation agit comme suit :

- **Si** (durée adaptation < 30 s) **Alors** le gestionnaire n'envoie pas de massage *getState()* pendant l'adaptation.
  - *** Si** aucun message de notification de fin n'est reçu après l'écoulement du double du temps estimé, **Alors** le gestionnaire d'adaptation conclut qu'il s'agit d'une déconnexion.
- **Si** (durée adaptation > 30 s), **Alors** le gestionnaire d'adaptation envoie messages *getState()* toutes les 30 s.
  - *** Si** l'adaptateur ne répond pas à un de ces messages intermédiaires **OU Si** pas de message de notification de fin, **Alors** le gestionnaire d'adaptation détecte une déconnexion de l'adaptateur.

Notons que la déconnexion d'une machine réalisant l'adaptation d'un média composant le document est moins grave que la déconnexion de la machine centralisée qui réalise l'adaptation de tous les médias du document composé ce qui corrobore une solution d'adaptation distribuée.

4.3.2 Premier cas de la gestion des disparitions des adaptateurs

Le plus simple et le moins coûteux en terme d'implémentation est de considérer que lorsqu'un nœud adaptateur disparaît, la partie du média adaptée jusque-là est perdue. Une fois que le gestionnaire d'adaptation détecte la déconnexion du nœud fournissant l'adaptateur, il cherche un ou plusieurs adaptateurs (en cas de composition) réalisant le même processus d'adaptation et les instancie.

Dans ce cas, la réplication des adaptations n'est réalisée que lorsqu'une disparition est détectée (une seule adaptation pour le même média à la fois). Ceci entraîne une perte de temps mais n'implique pas forcément l'échec de l'opération d'adaptation (à condition de trouver un ou plusieurs adaptateurs qui réalisent le même traitement).

Cette solution, bien que facile à mettre en œuvre, augmente la durée de l'adaptation globale, surtout dans le cas où l'adaptateur disparu réalisait des processus d'adaptation longs tels que le transcodage vidéo. La figure ci-dessous illustre ce cas :

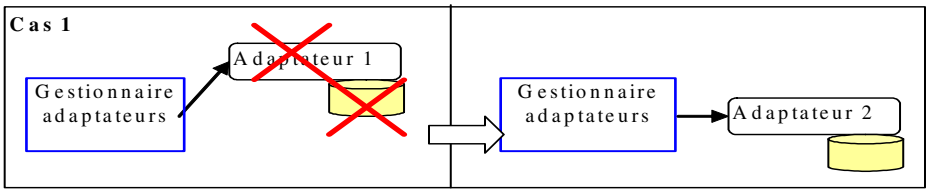

Figure 43 : Cas 1 : Chercher un autre adaptateur pour remplacer l'adaptateur disparu

4.3.3 Deuxième cas de la gestion des disparitions des adaptateurs

La réplication des adaptations dès la première instanciation est une deuxième solution. Le gestionnaire d'adaptation instancie deux adaptateurs identiques pour adapter la même source si c'est possible (présence dans le réseau de deux adaptateurs répondant aux mêmes critères). Le média est ainsi adapté à deux endroits différents. Ainsi, dès qu'un adaptateur finit d'adapter, ses ressources sont libérées ainsi que celles de l'autre adaptateur même s'il n'a pas fini l'adaptation : le média adapté est donc récupéré depuis l'adaptateur le plus rapide.

Si nous souhaitons considérer plusieurs niveaux de tolérance aux déconnexions, dès la disparition d'un adaptateur, l'opération de recherche et d'instanciation d'adaptateur est répétée, afin d'avoir en permanence, à un instant t, deux adaptateurs pour un média et un graphe d'adaptation donnés.

La figure ci-dessous illustre ce deuxième cas :

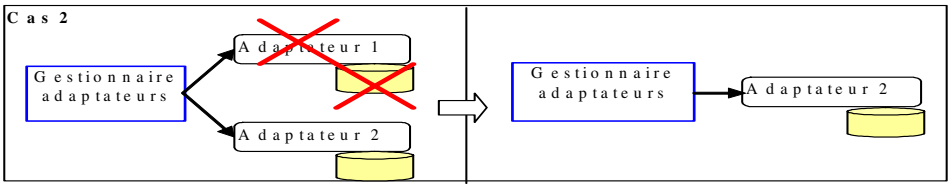

Figure 44 : Cas 2 : Deux adaptateurs pour le même média sont instanciés en même temps

4.3.4 Remarque

La réplication des adaptations dans les deux cas n'a de sens que si les adaptateurs existent dans le réseau. Ainsi, la tolérance aux disparitions des adaptateurs n'est assurée que s'il existe des adaptateurs identiques offerts par des nœuds distribués différents. Si un adaptateur disparaît et que le gestionnaire d'adaptation ne trouve aucun autre adaptateur (ou composition d'adaptateurs) pouvant réaliser la même chose, il le signale au planificateur qui décide de supprimer le média concerné du document multimédia final.

4.4 Reconstruction du graphe d'adaptation

Comme nous venons de le voir, la gestion de la dynamicité implique la reconstruction du graphe d'adaptation physique par le gestionnaire d'adaptation. Ceci suppose que ce graphe doit être modifiable

à la volée, soit à la suite de la modification du graphe d'adaptation logique, soit à la détection d'une déconnexion d'un pair fournisseur d'adaptateur(s).

Suivant les cas, la reconstruction implique soit la recherche et éventuellement la composition et l'instanciation de nouveaux adaptateurs, soit l'arrêt d'autres adaptateurs réalisant des, processus d'adaptation supprimés du graphe.

Par exemple, lors d'un changement dans le contexte de l'utilisateur, une nouvelle phase de prise de décision, relative au média concerné par ce changement, est réalisée. Le graphe d'adaptation logique est partiellement reconstruit ce qui conduit à une reconstruction partielle du graphe d'adaptation physique.

Par contre, lors d'un changement dans le contexte des adaptateurs, il n'est pas nécessaire de reprendre des décisions ni de reconstruire le graphe d'adaptation logique. En effet, lors de la disparition d'un adaptateur, le gestionnaire d'adaptation cherche un adaptateur similaire qui remplacera celui qui a disparu dans le graphe d'adaptation physique.

5 Conclusion

Dans ce chapitre, nous avons présenté la stratégie utilisée dans PAAM pour l'identification et la description des adaptateurs. Nous avons, par la suite, présenté un protocole de négociation et d'acceptation et une politique d'acceptation, utiles lors du choix des adaptateurs par le gestionnaire d'adaptation pendant la phase d'instanciation du graphe d'adaptation physique.

Nous avons présenté plusieurs manières de gérer les déconnexions des adaptateurs distribués. Les mécanismes de détection des déconnexions demeurent les mêmes. Chacune de ces solutions a ses avantages et ses inconvénients.

Le premier cas requiert le mécanisme de détection de la déconnexion d'un adaptateur et le mécanisme de la recherche et d'instanciation d'un adaptateur. Ce dernier mécanisme est le même que celui qui sert à la première recherche et instanciation. Cette solution a pour inconvénient l'augmentation de la durée d'adaptation.

Le deuxième cas est également assez facile à mettre au point. L'inconvénient est qu'il consomme deux fois plus de ressources des nœuds du réseau.

Le chapitre suivant présente notre implémentation d'une chaîne d'adaptation PAAM complète.

Chapitre 6

Implémentation d'une chaîne d'adaptation complète

1 Introduction

Dans ce chapitre, nous présentons et motivons nos choix technologiques relatifs à l'utilisation des services Web. Nous décrivons notre implémentation de la chaîne d'adaptation complète correspondant aux spécifications de PAAM, présentées dans le chapitre 3.

Nous présentons dans la Figure 45 les phases principales d'un cycle d'adaptation complet, inspirées de la Figure 2 du chapitre 2. L'implémentation de PAAM a pour objectif de réaliser la totalité de ces fonctionnalités.

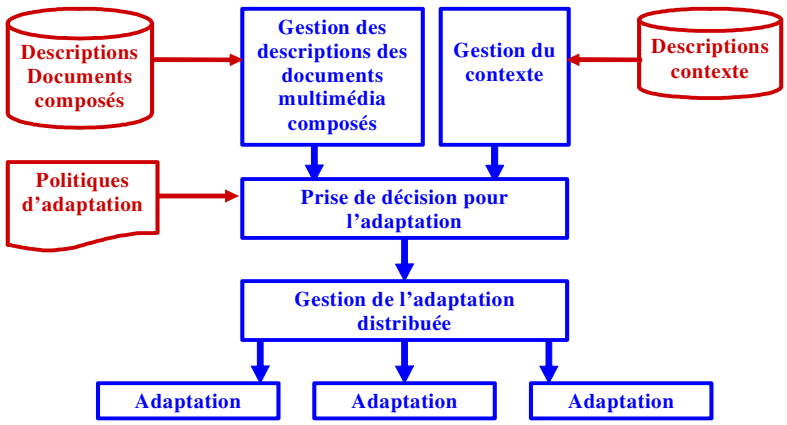

Figure 45 : Les éléments de base à implémenter pour réaliser une plate-forme d'adaptation basée sur PAAM

Le reste du chapitre est organisé comme suit : avant de justifier nos choix technologiques et nos moyens de description du contexte utilisateur et du document multimédia composé, nous présentons un aperçu de la chaîne d'adaptation complète implémentée. Nous exposons par la suite notre algorithme de prise de décision en illustrant quelques-unes de nos politiques d'adaptation.

2 Aperçu de la chaîne d'adaptation complète de PAAM

La Figure 46 montre toutes les étapes de la chaîne d'adaptation que nous avons implémentée suivant l'architecture décrite dans le chapitre 3.

Les quatre gestionnaires et les adaptateurs sont implémentés sous forme de services Web. Nous justifions notre choix pour cette technologie dans la section 3

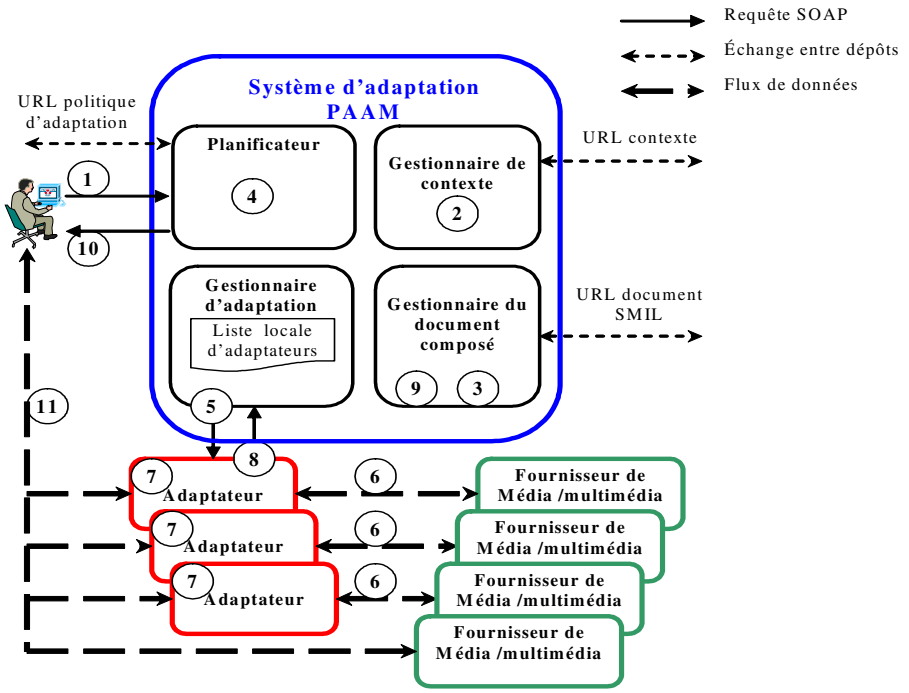

Figure 46 : Une chaîne d'adaptation complète

Nous rappelons qu'une requête typique se déroule comme suit :

L'utilisateur invoque le planificateur en envoyant une requête SOAP (1) contenant l'URL du document SMIL ainsi qu'un lien vers la description du contexte de l'utilisateur.

Le planificateur envoie deux requêtes, l'une au gestionnaire de contexte et l'autre au gestionnaire de documents composés. Ces deux gestionnaires récupèrent et analysent les descriptions du contexte et du document multimédia composé ((2) et (3)) puis envoient les données agrégées au planificateur.

L'étape suivante porte sur les prises de décision concernant chaque média élémentaire composant le document original (4). Dans cette phase, le planificateur utilise les politiques d'adaptation et applique son algorithme et ses heuristiques de prise de décision. Le résultat de cette phase est la construction du graphe d'adaptation logique.

Le passage du graphe d'adaptation logique au graphe d'adaptation physique se fait au niveau du gestionnaire d'adaptation qui utilise la liste locale des adaptateurs. Cette liste contient les caractéristiques de chaque adaptateur : l'identifiant URN, l'URL du service Web lui correspondant, et les mimetypes d'entrée et de sorties supportés, et si le service est payant ou pas et facilite la recherche des adaptateurs pour le gestionnaire d'adaptation.

Il est parfois possible que le nombre d'adaptateurs du graphe d'adaptation physique soit plus grand que le nombre d'adaptations présentes dans le graphe d'adaptation logique. Pour mieux comprendre cela, considérons l'exemple d'une opération d'adaptation présente dans un graphe d'adaptation logique relative à une réduction spatiale d'une vidéo. Si après une phase de recherche, le seul adaptateur disponible ne supporte pas le mimetype d'entrée de la vidéo, le gestionnaire d'adaptation insère un adaptateur supplémentaire. Cet adaptateur réalise une opération de transcodage transformant le mimetype source de la vidéo en un mimetype supporté en entrée par ce réducteur spatiale de vidéo.

Après application du protocole de négociation et d'acceptation, le graphe d'adaptation physique est construit, et le gestionnaire d'adaptation envoie une requête SOAP à chaque URL d'adaptateur formant le graphe (5).

Chaque adaptateur récupère le média depuis les fournisseurs de média (6) et réalise l'adaptation (7). Certaines opérations d'adaptation ont été émulées, d'autres ont été implémentées. Chaque adaptateur informe le gestionnaire d'adaptation de la fin d'adaptation et envoie l'URL du nouveau média adapté (8). Une fois toutes les URL des médias adaptés reçues, le gestionnaire d'adaptation les transmet au planificateur. Ce dernier charge le gestionnaire de documents composés de reconstruire le document composé adapté (9).

Finalement, le client reçoit l'URL du document composé multimédia adapté (10), éventuellement stocké dans le cache du gestionnaire de documents composés, tandis que les médias adaptés peuvent être stockés dans les caches des adaptateurs qui les ont manipulés (11). Notons que certains médias peuvent ne pas avoir été adaptés, l'utilisateur les récupère alors directement depuis les fournisseurs des médias.

La section qui suit présente les choix technologiques qui nous ont servi à implémenter la chaîne d'adaptation complète.

3 Choix technologiques

Cette section présente et justifie nos choix concernant les outils technologiques que nous utilisons pour implémenter la plate-forme PAAM.

PAAM est un système distribué où les participants peuvent jouer les rôles de consommateurs ou producteurs de contenus multimédia, et éventuellement fournisseurs de ressources d'adaptation. Nous avons, dans le chapitre 4, motivé notre choix d'implémenter les entités PAAM à l'aide de la technologie des services Web. Rappelons que les services Web permettent d'avoir un système évolutif, facilement déployable et qui se prêtent au passage à l'échelle. Les services Web permettent le partage direct de ressources entre entités et répondent, en cela, à notre objectif.

Il faut alors choisir un serveur d'applications afin de déployer nos services Web ainsi que des outils facilitant l'implémentation des services Web.

3.1 Projets J2EE, serveurs d'applications et pile de services Web

Nous optons pour la réalisation de nos services Web à l'aide d'une technologie orientée Java. Il existe plusieurs projets open source visant à offrir des serveurs d'applications Java et des piles de services Web. Parmi les plus connus nous citons Apache et GlassFish [APACHE] [GLASS]. Nous les présentons dans ce qui suit.

3.1.1 Apache

La Apache Software Foundation (*Fondation Apache*) est une organisation à but non lucratif qui développe des logiciels libres sous la licence Apache. Elle a été créée en juin 1999 dans le Delaware aux États-Unis.

La Fondation Apache est une communauté décentralisée de développeurs qui travaillent sur ses projets open source. Les projets Apache sont caractérisés par un mode de développement collaboratif basé sur le consensus ainsi que par une licence de logiciel ouverte et pragmatique. Chaque projet est dirigé par une équipe de contributeurs auto-désignée et on ne devient membre de la fondation qu'après avoir contribué activement aux projets Apache.

Les principaux produits Apache qui nous intéressent particulièrement sont le projet Tomcat et le projet Web Services qui inclut le sous projet AXIS 2 [TOMCAT] [WSPROJ] [AXIS2].

- Apache Tomcat est un moteur de servlets. Issu du projet Jakarta, Tomcat est désormais un projet à part entière de la fondation Apache [JAKA]. Tomcat implémente les spécifications des servlets Java et des JavaServer Pages de Sun Microsystems. Il inclut des outils pour la configuration et la gestion, mais peut également être configuré en éditant des fichiers de configuration XML. Tomcat inclut un serveur HTTP interne, et est aussi considéré comme un serveur HTTP. Apache Tomcat est développé dans un environnement ouvert et participatif.
- AXIS 2 est un moteur de services Web. Ce standard est une nouvelle conception et une réécriture du très largement utilisé Apache Axis [AXIS]. L'architecture Axis 2 a vu le jour en Août 2004 à un sommet à Colombo au Sri Lanka. Axis 2 est plus flexible, efficace et configurable que l'architecture Axis 1.X. En effet, Axis 2 a été conçue pour ajouter aisément des « modules » plug-in qui ajoutent des fonctionnalités telles que la sécurité. Certains concepts tels que les handlers ont été repris par Axis 2. Axis 2 est construit autour du concept Apache AXIOM, un nouveau modèle objet très performant et basé sur XML. Les outils AXIS 2 sont : Axis2 Web Application (Web App), WSDL2WS, Service Archive Wizard et Java2WSDL.

3.1.2 GlassFish

GlassFish est un projet de développement open source d'un serveur d'application Java EE 5. Il sert de fondation au produit *Sun Java System Application Server PE 9* de Sun Microsystems. Ce projet fournit un processus structuré pour le développement d'un serveur d'applications. C'est la réponse aux

développeurs Java désireux d'accéder aux sources et de contribuer au développement des serveurs d'applications de nouvelle génération de Sun.

Au niveau des standards, GlassFish est une implémentation complète de la norme Java EE 5 qui recouvre notamment une pile de services Web : JAX-WS [JAXWS].

JAX-WS fait partie du projet JAX-WS Reference Implementation (RI). JAW-WS RI développe et inclut la base de l'implémentation de référence de Java API for XML Web Services (JAX-WS).

JAX-WS est un cadre de travail (Framework) de services Web qui fournit les outils et l'infrastructure pour développer des solutions services Web pour les utilisateurs finaux et des développeurs de middleware. JAX-WS propose les outils suivants :

- L'outil « apt » qui fournit un support pour traiter des annotations ajoutées par Java à travers la JSR[11] 175 appelée *Metadata Facility for the Java Programming Language*. En résumé, la JSR 175 permet aux programmeurs de déclarer de nouveaux types de modificateurs structurés pouvant être associés avec des éléments, des champs des méthodes et des classes de programme. Apt génère tous les artifacts[12] portables d'un service Web JAX-WS
- L'outil « wsimport » génère des artifacts JAX-WS portables utilisables par des utilisateurs et des services. Cet outil lit une description WSDL et génère tous les artifacts nécessaires pour le développement, le déploiement et l'invocation de services Web.
- L'outil « wsgen » lit la classe implémentée d'un service endpoint et génère tous les artifacts portables d'un service Web JAX-WS.

En conclusion, JAX-WS permet coté serveur de générer un package service Web à partir de sources Java, de classes Java ou depuis un fichier WSDL à l'aide de l'un des trois outils présentés ci-dessus, et coté client de générer les classes permettant d'invoquer le service Web.

3.2 Choix d'implémentation

Nous avons réalisé nos premières versions de services Web de PAAM à l'aide de la pile web services AXIS 2. Nous avons déployé ses services Web au sein du moteur Apache Tomcat.

Nous avons par la suite migré pour la pile de services Web JAX-WS tout en gardant le même serveur d'applications Apache Tomcat. Ce choix est principalement motivé par deux éléments : les outils JAX-WS génèrent moins de code que ceux de AXIS 2 et JAX-WS facilite la programmation car il repose sur JAXB (*Java Architecture for XML Binding*), un standard permettant de générer des document XML à partir des entités Java et vice versa [JAXB].

11 : JSR : Java Specifications Request est un système normalisé ayant pour but de faire évoluer la plate-forme JAVA. Il existe actuellement 328 JSR (de 1 à 300 et de 901 à 927) ;

12 : un artifact représente de manière générale un fichier (par exemple un document d'analyse des besoins ou de conception, un code source, un makefile, un script de test, etc.). Dans le cadre de JAX-WS, il s'agit de fichiers XML nécessaires pour construire l'archive .war relative au service Web (par exemple sun-jaxws.xml, web.xml)

Le projet JAX-WS Reference Implementation (RI) a mis au point une extension permettant aux développeurs de récupérer les états des objets au sein des services Web. À l'origine, JAX-WS RI prévoyait de créer une seule instance de classe du service Web qui devait répondre à toutes les requêtes entrantes ; ceci rendait impossible d'utiliser de manière significative les champs d'une instance précise. Les dernières versions de JAX-WS introduisent la notion de service Web avec état (*Stateful Web Service*) pour répondre à ce problème et permettre la manipulation de plusieurs instances de classe d'un service donné. L'utilisation de WS-Addressing [WSAD] fournit un protocole basé sur les standards et un modèle de programmation facile à utiliser notamment en intégrant des notations telles que @ stateful et @Addressing, qui, annotées au début d'une classe service Web signifient que ce service Web gère la gestion d'état et d'adressage.

WS-Addressing définit l'adressage des services Web et l'identification des messages indépendamment du transport. WS-Addressing offre aux développeurs une méthode d'adressage d'objets pour les applications de services Web, il étend les capacités des services Web en permettant les échanges de messages asynchrones et en permettant l'interaction de plus de deux services à la fois. Ce standard permet ainsi l'asynchronisme et les mécanismes de Call-Back.

4 Éléments de la chaîne d'adaptation

Cette section a pour but de présenter les entrées et sorties de chaque maillon de la chaîne d'adaptation. Nous commençons par décrire nos solutions pour décrire le contexte d'un utilisateur et la description d'un document multimédia composé. Nous présentons par la suite un exemple d'algorithme de prise de décision et de construction du graphe d'adaptation logique et physique.

4.1 Description du contexte utilisateur

Le contexte d'après [Dey00] est toute caractéristique ou capacité d'une entité de l'environnement (utilisateur, terminal, réseau, environnement naturel de l'utilisateur, état et caractéristiques des modules réalisant l'adaptation) pouvant influer sur la manière de consommer un service donné, et susceptible de changer dans le temps suite à des événements. A cet effet et dans le cadre de PAAM, nous avons décidé de tenir compte des informations contextuelles suivantes :

- Informations personnelles de l'utilisateur
- Préférences utilisateur de présentation des médias
- Capacités logicielles et matérielles du terminal de l'utilisateur

La Figure 47 présente un exemple d'une description d'un contexte utilisateur compris par notre système d'adaptation PAAM.

Notons que la conception des politiques d'adaptation repose sur cette description du contexte et que nous nous sommes inspirés des outils DIA du standard MPEG-21.

```
<?xml version="1.0" encoding="UTF-8"?>
<UserContext>
<User>
        <UserPreferences>
                <PreferredLang>text/plain:en</PreferredLang>
                 <PreferredPlayer>RealOne</PreferredPlayer>
                <ServiceMaxPrice type="free">0</ServiceMaxPrice>
                <HandicapCategory>NONE</HandicapCategory>
                <VideoQuality>INCHANGED</VideoQuality>
                <AudioQuality>ADAPTED</AudioQuality>
                <ImageQuality>ADAPTED</ImageQuality>
                <TextQuality>INCHANGED</TextQuality>
                <VideoMaxHeight>200</VideoMaxHeight>
                <VideoMaxWidth>100</VideoMaxWidth>
                <ImageMaxHeight>50</ImageMaxHeight>
                <ImageMaxWidth>80</ImageMaxWidth>
                <VideoMaxSize>10240</VideoMaxSize>
                <ImageMaxSize>1200</ImageMaxSize>
                <AudioMaxSize>5120</AudioMaxSize>
        </UserPreferences>
</User>
<Terminal>
        <TerminalCapability>
                 <Player type="multimediaPlayer" name="QuickTime">
                 <SupportedMimetypes>
                      …
                 </SupportedMimetypes>
                 </Player>
                 <ScreenResolution>
                 <Height>300</Height>
                 <Width>250</Width>
                 </ScreenResolution>
                <ColorSupport value="true"/>
        </TerminalCapability>
</Terminal>
</UserContext>
```

Figure 47 : Description du contexte d'un utilisateur du système d'adaptation PAAM

Nous expliquons dans ce qui suit chacun des champs du contexte utilisateur.

4.1.1 Préférences de l'utilisateur

Les préférences utilisateur sont décrites à l'intérieur de la balise *« UserPreferences »* qui comprend les balises suivantes :

- *PreferredLang* : représente la ou les langue(s) préférée(s) de l'utilisateur suivant le format « *mimetype : lg* » où *mimetype* et *lg* correspondent respectivement au mimetype du texte (par exemple « text/plain ») et à l'abréviation de la langue (par exemple « fr » pour français, « en » pour anglais, etc.).
- *PreferredPlayer* : décrit la préférence de l'utilisateur en matière de lecteur multimédia (par exemple realOne ou QuickTime)
- *ServiceMaxPrice* : détermine si l'utilisateur accepte de payer ou non pour un service d'adaptation grâce à l'attribut « *type* » (free ou not free). Si l'utilisateur ne désire pas payer, la valeur de cet élément est égale à « 0 », sinon, elle représente (en €) le prix maximum que l'utilisateur payera pour un service d'adaptation donné.
- *HandicapCategory* : détermine la valeur du handicap de l'utilisateur ; les valeurs possibles sont *NONE* (aucun handicap), *BLINDNESS* (malvoyance) et *HEARINGIMPAIREMENT* (surdité). La valeur par défaut est fixée à *NONE*.
- *VideoQuality*, *AudioQuality* , *ImageQuality* et *TextQuality* : L'utilisateur peut, à travers ces balises, définir ses choix sur la qualité des médias vidéo, audio, image et texte. Les valeurs possibles sont : *INCHANGED* (si l'utilisateur ne souhaite aucun changement concernant ce média), *NONE* (si l'utilisateur ne souhaite pas ce média dans son document composé) et *ADAPTED* (si l'utilisateur souhaite adapter ce média, l'opération d'adaptation est fixée). Les valeurs par défaut sont *INCHANGED.*
- *VideoMaxHeight* et *VideoMaxWidth* : représentent les préférences utilisateur relatives à la largeur et la hauteur (en méga pixels) maximum du média vidéo.
- *ImageMaxHeight* et *ImageMaxWidth* : représentent les préférences utilisateur relatives à la largeur et la hauteur (en méga pixels) maximum du média image.
- *VideoMaxSize*, *ImageMaxSize*, *AudioMaxSize* : représentent les préférences utilisateur relatives à la taille (en kilo-octets) des médias vidéo, image et audio.

4.1.2 Capacités du terminal

Les capacités du terminal sont décrites à l'intérieur de la balise *« TerminalCapability »* qui comprend les balises suivantes :

- *Player* : une description de capacités de terminal peut avoir plusieurs balises *Player* car un terminal peut avoir plus d'un lecteur. Cet élément contient deux attributs : *type* (par exemple « multimediaPlayer ») et *name* (par exemple « QuickTime » ou « RealOne »)

- *SupportedMimetypes* : cet élément fils de l'élément « Player » énumère tous les mimetypes supportés par ce lecteur. À noter que d'un terminal à un autre, les valeurs des mimetypes d'un même lecteur peuvent changer. Cela peut être dû à la version du lecteur ou aux codecs installés.
- *ScreenResolution* : cet élément décrit la résolution du terminal à travers ses deux fils *Height* et *Width* qui représentent respectivement la résolution (en pixels) verticale et horizontale du terminal.
- *ColorSupport* : détermine si le terminal supporte ou pas la couleur. La valeur par défaut est *true.*

4.2 Description du document multimédia composé

Nous avons utilisé SMIL, présenté dans la section 7.3 du chapitre 2, afin de décrire certaines caractéristiques des médias utilisés (mimetype, langue), l'organisation spatiale et la synchronisation temporelle d'un document multimédia composé.

Les éléments que nous avons besoin de décrire sont :

- Les mimetypes : SMIL n'offre pas de balise spéciale pour décrire les mimetypes. Ainsi, nous avons utilisé l'attribut *metadata*, que l'on retrouve dans l'entête d'un document SMIL et qui contient des descriptions RDF.
- La langue du texte : le même attribut *metadata* nous sert à déterminer la langue du texte (par exemple *fr* pour francais, *en* pour anglais, etc.)
- L'URL de chaque média : l'attribut *src*, que l'on retrouve dans les balises média du corps d'un document SMIL et qui permet de décrire l'URL de chaque média.
- La taille (en méga pixels) du média : l'attribut *métadata* nous sert également à décrire la taille de chaque média composant le document SMIL.

La figure qui suit représente une entête d'un document SMIL représentant la balise *metadata* comprenant les éléments cités ci-dessus.

```
<?xml version="1.1" ?>
<smil>
…
<metadata id="description_PAAM">
  <rdf:RDF
     xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
     xmlns:dc = "http://purl.org/dc/elements/1.1/">
  <!-- Méta-données à propos de la vidéo 1 -->
  <rdf:Description rdf:about="#video1"
```

```
      dc:Format="video/mpg"
      dc:Language="en">
</rdf:Description>
<!-- Méta-données à propos de l'image 1 -->
<rdf:Description rdf:about="#image1"
      dc:Format="image/jpeg"
      dc:Language="en" >
</rdf:Description>
<rdf:Description rdf:about="#texte1"
      dc:Format="text/plain"
      dc:Language="fr" >
</rdf:Description>
</rdf:RDF>
 </metadata>
…
</smil>
```

Figure 48 : balise metadata d'un document SMIL correspondant aux spécifications de PAAM

4.3 Implémentation de la prise de décision

Le processus de prise de décision repose sur l'interprétation des descriptions du contexte, des descriptions du document multimédia composé et des politiques d'adaptation. Avant de décrire l'algorithme de prise de décision, nous commençons par donner quelques notions sur le graphe d'adaptation et sur les politiques d'adaptation utilisées par l'algorithme de prise de décision que nous avons implémenté.

4.3.1 Définition du graphe d'adaptation

Un graphe d'adaptation décrit les adaptations que doivent subir les médias élémentaires composant un document multimédia composé, en séquence ou en parallèle, afin que ce document puisse être consommé par un utilisateur doté d'un contexte particulier. La Figure 49 illustre un exemple de graphe d'adaptation.

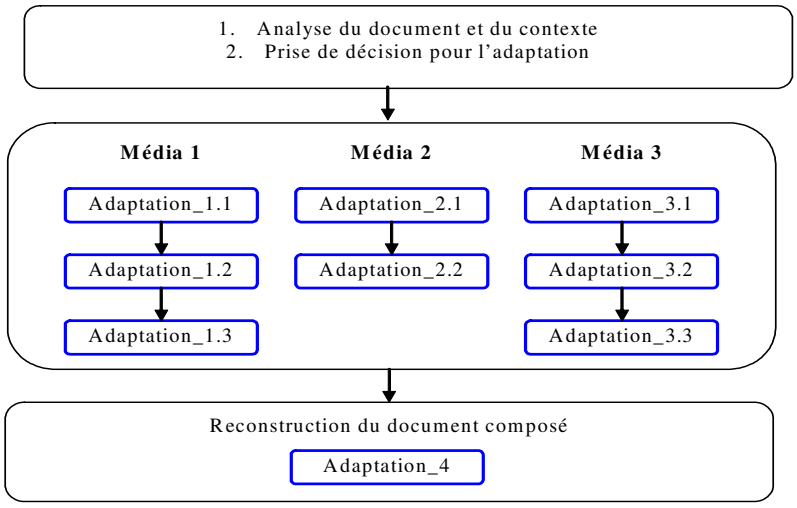

Figure 49 : Un graphe d'adaptation dans le cadre de PAAM

4.3.2 Politiques d'adaptation

Les politiques d'adaptation de PAAM sont des documents XML reposant sur la règle déclarative suivante l'expression : *Si condition(s) alors action*. Nous présentons deux principales catégories de politiques d'adaptation, celles relatives au handicap de l'utilisateur et celles relatives aux médias.

4.3.2.1 Politiques d'adaptation relatives au handicap

Ces politiques concernent le handicap potentiel d'un utilisateur. Les règles sont les suivantes :

- **Si** HandicapCategory = = "BLINDNESS" **Et** média = = "video" **Alors** adaptateur = "urn:paam:transformer:audiovideo:audio:audioExtractor"
- **Si** HandicapCategory = = "BLINDNESS" **Et** média = = "image" **Alors** adaptateur = "urn:paam:transmoder:image:audio:imageReader"
- **Si** HandicapCategory = = "BLINDNESS" **Et** média = = "text" **Alors** adaptateur = "urn:paam:transmoder:text:audio:textReader"
- **Si** HandicapCategory = = "HEARIMPAIREMENT" **Et** média = = "video" **Alors** adaptateur = " urn:paam:transformer:audiovideo:video:videoExtractor"
- **Si** HandicapCategory = = "HEARIMPAIREMENT" **Et** média = ="audio" **Alors** adaptateur = " urn:paam:transmoder:audio:text:audioTranscription"

4.3.2.2 Politiques d'adaptations relatives aux médias

Cette catégorie de politiques sert de support aux prises de décision pour chaque type de média (vidéo, image, audio et texte).

- **Si** VideoQuality = = "ADAPTED" **Et** média = = "video" **Alors** adaptateur = PlannerVideoAdaptation
- **Si** VideoQuality = = "NONE" et média = "video" alors adaptateur = "none"
- **Si** AudioQuality = = "ADAPTED" **Et** média = = "audio" **Alors** adaptateur = PlannerAudioAdaptation
- **Si** AudioQuality = = "NONE" et média = = "audio" alors adaptateur = "none"
- **Si** ImageQuality = = "ADAPTED" **Et** média = = "image" **Alors** adaptateur = PlannerImageAdaptation
- **Si** ImageQuality = = "NONE" et média = = "image" Alors adaptateur = "none"
- **Si** TextQuality = = "ADAPTED" **Et** média = = "text" **Alors** adaptateur = PlannerTextAdaptation
- **Si** TextQuality = = "NONE" **Et** média = = "text" **Alors** adaptateur = "none"

L'expression PlanerMediaAdaptation où Media = {Video, Image, Audio, Text} signifie que le choix de l'adaptateur est laissé au planificateur. Ainsi, chaque planificateur implémente ses propres politiques d'adaptation pour choisir les adaptations à réaliser, le rendant ainsi éventuellement différent d'autres planificateurs.

4.3.2.1 Remarque

Ces politiques d'adaptation sont propres au système d'adaptation PAAM que nous avons implémenté. D'autres implémentations d'un système d'adaptation PAAM peuvent compléter ces politiques d'adaptation, voire en définir d'autres afin de répondre à d'autres besoins. Elles doivent cependant suivre la même structure « Si condition alors action ».

4.3.3 Algorithme de prise de décision

Nous présentons dans la Figure 50 notre algorithme de prise de décision et de construction du graphe d'adaptation.

L'action *x.op(y)* ➔ *z* signifie que x réalise l'opération op ayant pour paramètre y au destinataire z.

L'action *x.Send(y)* ➔ *z* signifie que x envoie le message y au destinataire z.

Le message *x(y)* signifie le message avec le paramètre y.

L'opérateur *//* dans *action_1 // action_2* signifie que les deux actions action_1 et action_2 se font en parallèle.

4.4 Discussion sur l'algorithme de prise de décision

L'algorithme simplifié présenté ci-dessus est un exemple montrant l'utilisation des politiques d'adaptation. D'autres implémentations d'algorithmes sont possibles appliquant chacune des politiques

d'adaptation différentes de celles que nous avons présentées. Ainsi, chaque système d'adaptation possède des caractéristiques qui le différencient des autres systèmes d'adaptation PAAM.

Afin qu'un planificateur réponde aux spécifications de PAAM, il faudra qu'il offre des interfaces interopérables avec, d'un côté, les spécifications du contexte et du document multimédia composé, et d'un autre côté, le gestionnaire d'adaptation qui récupère un graphe d'adaptation. Le gestionnaire d'adaptation peut également gérer les adaptateurs en utilisant différentes méthodes. Il peut par exemple implémenter un moteur BPEL capable de gérer une composition de services Web. Dans le cas de notre implémentation, l'instanciation des services Web d'adaptation et la gestion de la composition sont gérées sans intégrer de moteur BPEL.

Début

1. **Établir la liste locale des adaptateurs**

Adaptors_List = {Adaptors / Adaptors = (U, In_MT, Out_MT, URL)} où :

- U : urn de l'adaptateur
- In_MT : mimetypes d'entrée de l'adaptateur
- Out_MT : mimetypes de sortie de l'adaptateur
- URL : URL de l'adaptateur

2. **envoi requête au gestionnaire du contexte et au gestionnaire du document**

planificateur.Send (*Analyse (URL_contexte)*) ➔ gestionnaire_contexte //

planificateur.Send (*Analyse (URL_document)*) ➔ gestionnaire_document

Où

- URL_contexte est l'adresse à laquelle est stocké le contexte utilisateur
- URL_document est l'adresse à laquelle est stocké le document multimédia demandé par l'utilisateur

3. **analyse de la description du contexte**

gestionnaire_contexte.Send(end_analyse( )) ➔ planificateur

4. **analyse de la description du document**

gestionnaire_document.Send(end_analyse( )) ➔ planificateur

5. **appliquer les politiques d'adaptation relatives au handicap**

Pour chaque média du document composé faire :

planificateur.Apply_handicap_policies( )

6. **appliquer les politiques d'adaptation relatives aux médias**

Pour chaque média du document composé faire :

planificateur.Apply _media_policies( )

7. **Construire le graphe d'adaptation logique**

Appliquer certaines heuristiques telles que :

- **Si** média = = vidéo **Alors**
  - Si video_size > video_size_maximum alors PlannerVideoAdaptation = urn:paam: transformer:video:video:videoCompressor

- Si video_mimetype n'est pas supporté alors transcoder = urn:paam:transcoder:video:video:videoFormatConverter

- **Si** média = = image **Alors**
  - **Si** image_size > image_size_maximum **Alors** PlannerImageAdaptation = urn:paam: transformer:image:image:imageCompressor
  - **Sinon Si** image_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder: image:image:imageFormatConverter

- **Si** média = = audio **Alors**
  - **Si** audio_size > audio_size_maximum **Alors** PlannerAudioAdaptation = urn:paam: transformer:audio:audio:audioCompressor
  - **Sinon Si** audio_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder:audio:audio:audioFormatConverter

- **Si** média = = texte **Alors**
  - **Si** text_lang ≠ PreferredLang **Alors** PlannerTextAdaptation = urn:paam:transformer:text:text:Translation
  - **Sinon Si** text_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder: text: text: textFormatConverter

8. Rechercher les adaptateurs physiques

Chercher dans la liste les adaptateurs correspondant au graphe d'adaptation logique.

- **Si** média = = vidéo **Alors**
  - **Si** video_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder:video:video:videoFormatConverter

- **Si** média = = image **Alors**
  - **Si** image_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder: image:image:imageFormatConverter

- Si média = = audio **Alors**
  - **Si** audio_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder:audio:audio:audioFormatConverter

- Si média = = texte **Alors**
  - **Si** text_mimetype n'est pas supporté **Alors** transcoder = urn:paam:transcoder:text:text:textFormatConverter

9. Appeler les adaptateurs

Pour chaque adaptateur faire

callAdaptor(URL_Media, Input_MimeType, Additional_Parameters, Output_Transcoder_MimeType, Output_Mimetypes) où

- URL_Media représente l'URL du media
- Input_MimeType représente le mimetype d'entrée du média
- Additional_Parameters représentent des paramètres additionnels nécessaires à la négociation avec le gestionnaire d'adaptation et à l'appel du service d'adaptation. Ils varient d'un média à un autre, par exemple pour la vidéo, la taille maximum (en méga-octets), la largeur et la hauteur maximum de la vidéo (en méga pixels), et un booléen représentant la possibilité d'afficher ou non de la couleur sont envoyés.
- Output_Transcoder_MimeType représente le mimetype attendu en sortie d'un transcodeur
- Output_Mimetypes représentent les mimetypes tolérés en sortie.

10. Superviser les adaptateurs

Pour chaque adaptateur instancié

TimeOut = 10 s

I = 30 s

Faire à chaque intervalle I

Send(getState()) ➔ Adaptor

Si pas de réponse après TimeOut **Alors** aller à 9

11. Reconstruction du document multimédia composé

planificateur.Send(Reconstruct(Graph_Media)) ➔ document_reconstructor

document_reconstructor.Send (URL_Adapted_Document) ➔ planificateur

12. Envoyer l'URL du document adapté à l'utilisateur

Fin

Figure 50 : Algorithme de prise de décision

5 Conclusion

Nous avons présenté dans ce chapitre une chaîne d'adaptation complète (de bout en bout). L'utilisateur envoie une requête à travers la plate-forme d'adaptation PAAM et reçoit un document multimédia composé adapté à son contexte. Nous avons opté pour la technologie des services Web car ils sont basés sur des standards largement utilisés à travers la toile tels que XML et HTTP. Les services Web représentent ainsi une solution :

- Indépendante des langages de programmation, des plates-formes d'exécution et de déploiement
- Facilement déployable et paramétrable
- Extensible et tolérante à d'éventuelles évolutions technologiques
- Permettant le passage à l'échelle

Nos choix se sont également portés sur une solution personnalisée inspirée des standards tels que MPEG-21 afin de décrire un contexte aussi particulier que Suzy dans le scénario « *Suzy and her PDA* ». Nous avons également présenté une version simplifiée de l'algorithme de prise de décision implémenté.

Nous présentons dans le chapitre suivant les tests et étude de performance ayant pour but de valider notre solution et de montrer qu'aucun surcoût significatif n'est induit par l'utilisation de PAAM en comparaison avec d'autres solutions d'adaptation existantes.

Chapitre 7

Étude des coûts et tests

1 Introduction

Ce chapitre est consacré à l'étude des coûts des fonctionnalités de PAAM que nous avons implémentées (c.f. chapitre 6). Nous présentons dans ce chapitre les tests réalisés afin de valider l'architecture PAAM. Ces tests montrent que les coûts induits par la mise en place d'une solution d'adaptation distribuée utilisant les services Web sont compensées par la parallélisation des adaptations. Ainsi, il n'y a pas ou peu de surcoût par rapport aux solutions existantes d'adaptation de contenus multimédia.

Ce chapitre s'organise comme suit : nous commençons par comparer PAAM avec une architecture d'adaptation de documents multimédia composés au contexte d'un utilisateur, et qui, contrairement à PAAM, ne distribue pas l'adaptation. Cette architecture de référence nous sert, par la suite, à identifier les fonctionnalités additionnelles de PAAM, telles que la gestion de l'adaptation distribuée, susceptibles d'induire des coûts additionnels et de mesurer ces coûts. Ainsi, nous réalisons une étude des coûts de ces fonctionnalités dans la section 3 dans laquelle nous montrons l'apport de la parallélisation des adaptation.

2 PAAM vs architecture d'adaptation centralisée de contenus multimédia

PAAM est basée sur le concept de l'intermédiation et de la distribution de l'adaptation sur plusieurs sites. Nous ne comparons pas PAAM avec une solution sans adaptation ou une solution basée sur la sélection[13]. En effet, il est rare qu'un producteur / fournisseur de contenus mette à disposition un grand nombre de versions d'un même média car cela demanderait plus d'espaces de stockage, plus de traitements, sans avoir la garantie que ces versions soient adaptés aux différents contextes utilisateurs pouvant exister.

Dans cette section, nous comparons PAAM avec une architecture qui adapte un document multimédia composé au contexte d'un utilisateur et qui ne distribue pas l'adaptation. Le but de cette comparaison

13 Le choix de l'utilisateur de la bonne version parmi celles offertes par le fournisseur du contenu

est d'identifier les fonctionnalités additionnelles de PAAM susceptibles d'induire des coûts additionnels afin de mesurer ces coûts par la suite.

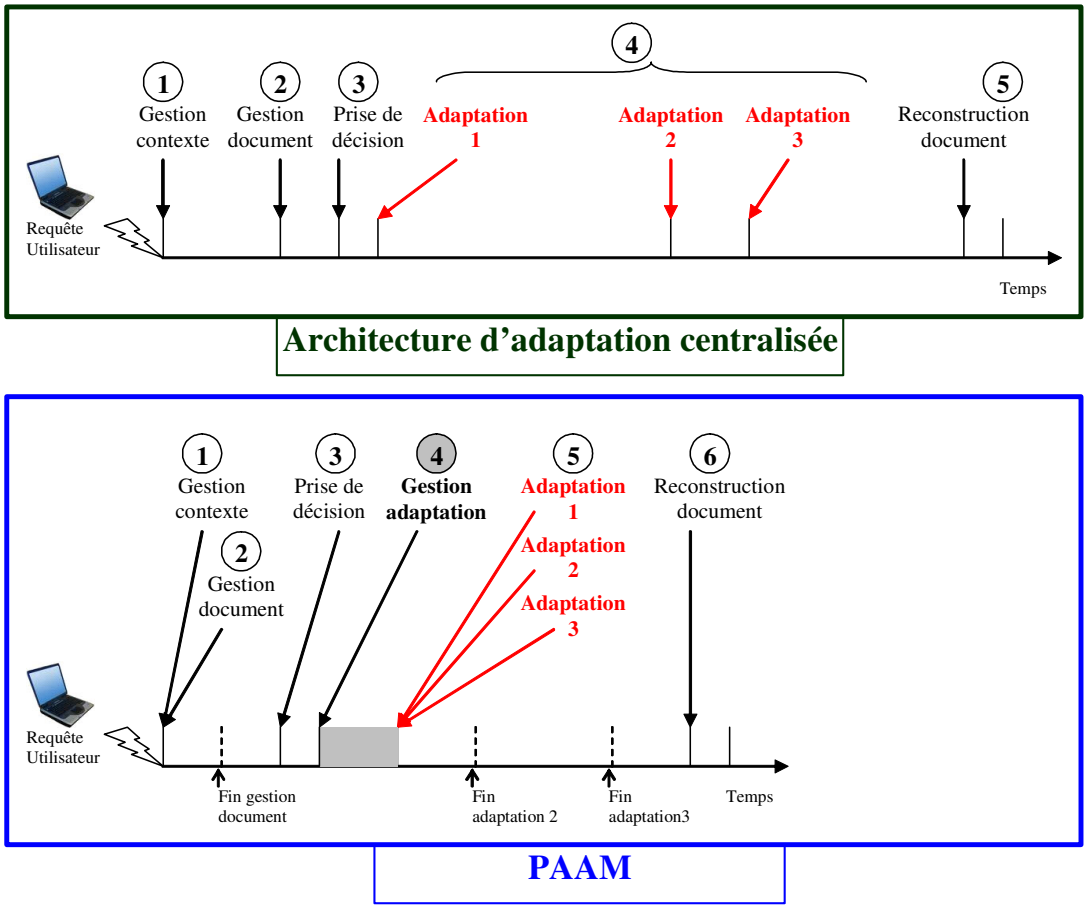

Figure 51 : PAAM vs architecture d'adaptation centralisée

2.1 Analyse et comparaison

La figure ci-dessus illustre les principales étapes d'adaptation au sein de deux différentes architectures : une architecture qui ne gère pas la distribution de l'adaptation, et PAAM. Les deux schémas utilisent la même échelle de temps.

Les numéros représentent les différentes étapes dans chaque architecture. PAAM compte une étape en plus : la gestion de l'adaptation, qui inclut la constitution de la liste des adaptateurs, l'implémentation du protocole de négociation et d'acceptation et la gestion des disparitions d'adaptateurs, trois fonctions nécessaires pour mener à bien la gestion des adaptateurs distribués. Néanmoins, la figure montre, de manière schématique, que la durée de traitement d'une requête d'adaptation d'un document multimédia nécessitant trois adaptations dure moins longtemps dans le cas de l'architecture de PAAM. Ceci est dû à la parallélisation des adaptations résultant de la distribution de l'adaptation, contrairement à ce qui se

produit dans le cadre d'une adaptation centralisée. Ainsi, la durée des adaptations dans le premier cas est égale à la somme des durées de chaque adaptation. À l'inverse, la durée des adaptations, dans le second cas, est égale à la durée de la plus longue adaptation.

Notons que PAAM permet de paralléliser également la gestion du contexte et la gestion du document multimédia composé, ce qui n'est pas forcément le cas des autres architectures d'adaptation de contenus multimédia.

La phase de gestion de la prise de décision et la phase de reconstruction du document multimédia composé sont communes aux deux architectures comparées.

2.2 Conclusion

PAAM sépare les fonctions de prise de décision de l'adaptation proprement dite, ce qui la distingue de la plupart des architectures, telles que ADMITS, ISIS, NAC et UMA, qui réalisent les deux fonctions au sein du même nœud. L'adaptation est distribuée sur différents nœuds, même si elle reste orchestrée par le système d'adaptation de PAAM. L'introduction des nœuds adaptateurs au sein de la plate-forme d'adaptation PAAM implique des coûts supplémentaires. Par conséquent, les fonctionnalités de PAAM susceptibles de générer des coûts additionnels par rapport aux autres architectures sont :

- La constitution de la liste temporaire des adaptateurs disponibles
- L'implémentation du protocole de contrôle d'admission
- La gestion des déconnexions des adaptateurs

Notons que PAAM peut également être comparée aux architectures qui adaptent des médias élémentaires tels que des vidéos ou des flux audio, ce qui implique pour PAAM un coût supplémentaire dû à l'analyse et à la reconstruction du document multimédia composé.

Nous présentons et mesurons dans ce qui suit les coûts induits par ces quatre fonctionnalités additionnelles.

3 Études des coûts des fonctionnalités additionnelles de PAAM

Le but de cette section est de mesurer les coûts des fonctionnalités de PAAM que n'implémentent pas les autres architectures de fourniture de contenus multimédia.

La section qui suit présente les conditions dans lesquelles nous avons réalisé nos expérimentations.

3.1 Conditions d'expérimentations et hypothèses

Les tests ont été réalisés sur une machine dotée d'un processeur Intel Pentium M 760, avec une fréquence de 2.0 GHz et d'une mémoire RAM de 1 Go. Le réseau utilisé est Fast Ethernet et possède un débit de 100 Mo/s.

Concernant les adaptateurs, certains ont été émulés tandis que d'autres tels que les transcodeurs vidéo ont été utilisés dans les tests (c.f. Annexe 1).

Les mesures que nous présentons dans les prochaines sections ont été réalisées en terme de délai (durée) et de nombre de messages (SOAP) envoyés. Structurellement parlant, un message SOAP est un contenu XML encapsulé dans une enveloppe HTTP et envoyé dans le réseau. Le tableau suivant présente les mesures relatives à l'échange de messages SOAP dans les conditions d'expérimentations décrites ci-dessus.

Éléments de services Web	Taille	Durée d'échange
Requête SOAP	800 bytes	60 ms
Réponse SOAP	600 bytes	

Tableau 10 : Temps d'envoi de requêtes SOAP

Notons que l'utilisation de JAX-WS implique, pour chaque première requête d'un service Web, deux requêtes/réponses SOAP, les premières servant à récupérer la description WSDL et les deuxièmes réalisant réellement l'appel de la méthode distante. Ainsi, si « NA » est le nombre d'adaptateurs à appeler, le nombre de messages SOAP échangés (sans gestion de disparitions ni prise en compte des états d'un adaptateurs) est égal à *« 4 x NA »*

Nous présentons, dans ce qui suit, les mesures relatives aux fonctionnalités additionnelles de PAAM. Par la suite, nous verrons que ces coûts sont compensés par la parallélisation des adaptations.

3.2 La constitution de la liste temporaire des adaptateurs disponibles

La liste temporaire des adaptateurs résulte de l'analyse des descriptions des adaptateurs présents dans le voisinage réseau du gestionnaire d'adaptation. Elle est utilisée lors du protocole de négociation et d'acceptation.

Afin de construire cette liste, il faut rechercher ces descriptions, les analyser et construire la liste, qui est, éventuellement, stockée localement.

Dans le cadre de PAAM, la phase de recherche des descriptions PAAM n'est pas prise en compte. En effet, un nœud fournisseur d'adaptateurs souhaitant faire partie du réseau d'adaptateurs d'un système d'adaptation PAAM annonce et stocke sa ou ses description(s) d'adaptateur(s) PAAM dans un dépôt connu de ce système d'adaptation PAAM. Ainsi, le coût induit par la constitution de la liste ($T_{TRAITEMENT}$) dépend des coûts relatifs à la récupération des fichiers descriptifs WSDL des adaptateurs ($T_{TRANSFERT}$) et à l'analyse des descriptions de ces adaptateurs ($T_{PARSING}$), donc :

$$T_{TRAITEMENT} = \mathbf{T_{TRANSFERT} + T_{PARSING}}$$

Avec :

$T_{TRANSFERT}$ = NA x TR où :

NA : nombre de descriptions d'adaptateur à transférer.

TR : temps de transfert d'une description WSDL étendue.

A titre d'exemple, nous avons réalisé des mesures sur la constitution d'une liste temporaire relative à 1000 descriptions d'adaptateurs.

En considérant le cas où les descriptions PAAM des adaptateurs sont stockées dans un dépôt distant, nous avons mesuré la durée de transfert des descriptions en utilisant le protocole d'échange de fichiers : FTP (File Transfer Protocol). Une description PAAM d'un adaptateur, présentée dans la section 2.3 du chapitre 5, possède une taille moyenne de 2400 octets et son temps de transfert vers le cache du gestionnaire d'adaptation varie entre 20 et 40 millisecondes.

Nous avons, par la suite, réalisé des tests visant à évaluer la durée de traitement (parsing) d'un ensemble de descriptions d'adaptateurs PAAM. Les tests ont été faits dans le cas où les descriptions sont stockées en local. Nous avons fait varier le nombre de descriptions à analyser et obtenu le graphe suivant :

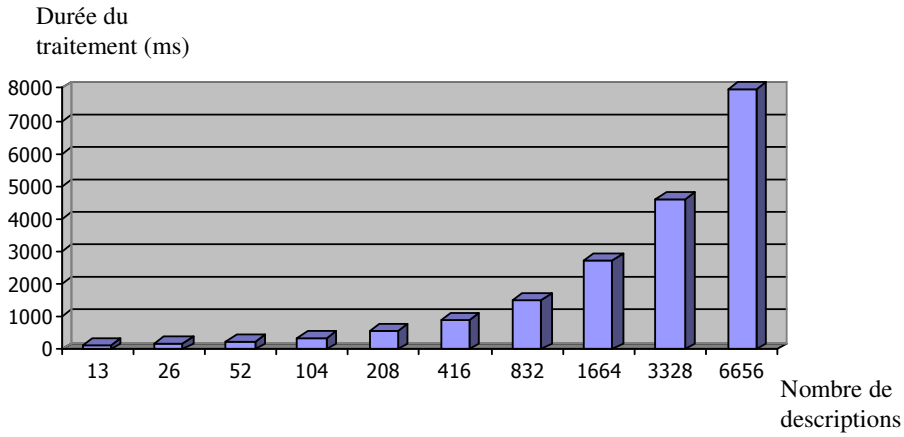

Figure 52 : Variation du temps de parsing des descriptions PAAM des adaptateurs suivant leur nombre

La Figure 52 montre que la durée de traitement des descriptions des adaptateurs ne dépasse pas les 3 secondes pour 1000 descriptions.

Ainsi, si nous considérons que TR a une valeur moyenne de 30 ms, alors le temps moyen nécessaire pour constituer la liste temporaire de 1000 adaptateurs est égal à :

$T_{TRAITEMENT\ (1000\ adaptateurs)} = 1000 \times 30 + 3000 = 33000$ millisecondes = 33 secondes.

En conclusion, le délai induit par la phase de construction de la liste locale varie d'un cas à un autre et suivant le mode de transfert des descriptions WSDL étendues. Notons cependant que cette construction est réalisée à une fréquence relativement basse : tout d'abord au déploiement du système d'adaptation PAAM et ensuite à des intervalles assez espacés (par exemple toutes les heures). Des mécanismes peuvent être mis en place, lors de la mise à jour de la liste locale, afin d'éviter qu'une même description PAAM ne soit transférée puis analysée si elle n'a pas été modifiée.

3.3 Implémentation du protocole de négociation et d'acceptation

En instanciant le graphe d'adaptation physique, le gestionnaire d'adaptation démarre un protocole de négociation et d'acceptation avec chacun des adaptateurs du graphe. Le protocole que nous avons proposé dans la section 3.2 du chapitre 5 vise à minimiser le nombre de messages envoyés. Ainsi, le premier message que le gestionnaire d'adaptation envoie à un adaptateur contient les paramètres d'appel ainsi que les paramètres de négociation.

Deux cas possibles résultent de l'application du protocole de négociation et d'acceptation, en fixant, par exemple, à 1 le nombre d'adaptateurs à tester de la liste temporaire des adaptateurs :

- L'adaptateur peut réaliser l'adaptation et commence à adapter : dans ce cas, un message est envoyé au gestionnaire d'adaptation contenant une estimation du temps d'adaptation. À la fin de l'adaptation, un message de notification de fin d'adaptation contenant l'URL du média adapté est envoyé au gestionnaire d'adaptation. Dans ce cas, **5** messages sont nécessaires pour gérer chaque adaptateur du graphe : deux messages requête / réponse pour récupérer la description WSDL, un message requête contenant les paramètres d'appel et de négociation, un message d'acquittement et un message de notification de fin d'adaptation.
- L'adaptateur ne peut pas réaliser l'adaptation : dans ce cas, un message d'exception est envoyé au gestionnaire d'adaptation qui choisit un autre adaptateur et lui envoie un autre message. Dans ce cas, le nombre de messages est égal à ***« (NME x 4) + 5 »***, où *«NME »* est le nombre d'adaptateurs candidats pour une adaptation donnée, qui ne peuvent pas réaliser l'adaptation et envoient ainsi des messages d'exception. Notons que le nombre total de messages envoyés entre le gestionnaire d'adaptation et un adaptateur ne pouvant pas réaliser l'adaptation est égal à **4** car il n'y a pas de message de notification de fin.

En résumé, si *« NA »* est le nombre d'adaptateurs à gérer, le nombre de messages envoyés par le gestionnaire d'adaptation, en appliquant le protocole de négociation et d'acceptation, est égal à :

$$NA \times 5 + \sum (NME_i \times 4)$$

Où « NME_i » est le nombre d'adaptateurs ayant envoyé un message d'exception correspondant au processus d'adaptation *« i »* du graphe d'adaptation physique.

3.4 Gestion des déconnexions des adaptateurs

Hormis les messages envoyés pour gérer le protocole de négociation et d'acceptation, le gestionnaire d'adaptation envoie d'autres messages, pendant les adaptations des médias, afin de vérifier l'état de l'adaptation et la présence ou pas du nœud offrant l'adaptateur. Le but de cette section est de quantifier le nombre de messages envoyés suivant les cas. Nous rappelons dans ce qui suit le mécanisme de détection des disparitions que nous avons présenté dans la section 4.3.1 du chapitre 5, tout en quantifiant, dans chaque cas, le nombre de messages envoyés :

- **Si** la durée estimée ne dépasse pas un seuil « S » propre à chaque gestionnaire d'adaptation (par exemple 30 s), **Alors** il n'envoie pas de messages pour vérifier l'état. **Si** aucun message de notification de fin n'est reçu après l'écoulement du double du temps estimé, **Alors** le gestionnaire d'adaptation conclut qu'il s'agit d'une déconnexion, cherche un autre adaptateur et l'instancie. Ainsi, **Si** *« NAD_i »* est le nombre d'adaptateurs disparus, **Alors** le nombre de messages supplémentaires échangés entre le gestionnaire d'adaptation et les adaptateurs disparus correspond à ***« $\sum(NAD_i \times 4)$ »***. Ceci s'explique par le fait que l'adaptateur disparu n'envoie pas de message de notification de fin d'adaptation.

- **Si** la durée estimée dépasse le seuil « S », **Alors** le gestionnaire d'adaptation envoie, toutes les « S » secondes, *« ME »* messages pour vérifier l'état avec *« ME = ($\lceil (DE / S) \rceil - 1$) »* [14], *« DE »* représentant la durée estimée. Ainsi, le nombre maximum de messages échangés entre le gestionnaire d'adaptation et un adaptateur est ***« ($\lceil (DE / S) \rceil - 1) \times 2$ »*** en supposant que l'adaptateur réponde à chaque fois par un message d'acquitement. **Si** le service Web destinataire ne répond pas à un de ces messages intermédiaires **OU Si** aucun message de notification de fin n'est reçu après l'écoulement du temps estimé, **Alors** le gestionnaire d'adaptation conclut à une déconnexion de l'adaptateur, cherche un autre adaptateur et l'instancie.

D'une manière générale le nombre total de messages échangés lors de la gestion des disparitions des adaptations est égal à :

$$\sum\left(\left(\left(\lceil (DE_i / S) \rceil - 1\right) \times 2\right) + NAD_i \times 4\right)$$

Avec :

« S » : seuil à partir duquel il est nécessaire d'envoyer des messages de vérification de l'état d'un adaptateur.

« NAD_i » : nombre d'adaptateurs disparus en réalisant l'adaptation i.

« DE_i » : durée estimée de l'adaptation i.

Notons que cette formule suppose que la détection de la disparition est réalisée lors du premier envoi du message d'état. Dans le cas contraire, le nombre total de messages échangés lors de la gestion des disparitions des adaptations augmente.

3.5 Gestion des documents multimédia composés

Deux étapes sont à considérer dans l'analyse des coûts de traitement d'un document multimédia composé : l'analyse du document multimédia composé et la reconstruction de ce document.

14 $\lceil (DE / 30) \rceil$ signifie la partie entière supérieure (ceiling en anglais) du résultat de la division DE / 30. Par exemple si DE = 50 alors $\lceil (DE / 30) \rceil = 2$

L'analyse d'un document multimédia composé implique l'analyse d'un contenu XML contenant des descriptions d'au moins un média.

La recomposition (reconstruction) du document composé suit la fin et la synchronisation de toutes les adaptations. Nous considérons la recomposition comme l'ultime adaptation avant d'envoyer le lien du document adapté à l'utilisateur. Il s'agit alors d'appliquer d'autres algorithmes d'adaptation au sein de cet adaptateur « spécial » qui recompose et qui, dans le cas de notre implémentation, est le gestionnaire de document multimédia.

Afin de calculer le coût de ces traitements relatifs aux documents multimédia composés, nous calculons le temps nécessaire à l'analyse et à la décomposition d'un document composé ainsi que le temps global de sa reconstruction. Le tableau suivant résume les coûts estimés de ces étapes :

	Taille du document multimédia composé	Durée de l'analyse
Analyse et décomposition d'un document (3 médias)	De Kb 1,34 à 2,28 Kb	0,3 ms
Temps de la recomposition d'un document (3 médias)	De Kb 1,34 à 2,28 Kb	0,034 ms

Tableau 11:Durée de temps de traitement d'un document multimédia composé

Ces mesures montrent que le coût résultant de l'analyse et de la reconstruction du document multimédia composé ne dépasse pas les 0,34 s. Rappelons que ces deux fonctionnalités consistent à analyser (parser) des documents XML (les documents SMIL), à en extraire tout ou partie des éléments les constituant et à traiter les informations extraites ce qui constitue une complexité dépendant de la taille des documents XML à analyser.

Notons également que d'après la Figure 51, l'analyse du document se fait en parallèle de l'analyse du contexte qui dure plus longtemps. Le coût induit par l'analyse du document est ainsi masqué par l'analyse du contexte.

3.6 Apport de la parallélisation des adaptations

Nous avons présenté, pour chacun des fonctionnalités spécifiques de PAAM, une étude des coûts en terme de nombre de messages envoyés et de temps de traitement.

Contrairement à la constitution de la liste temporaire des adaptateurs, le protocole de négociation, la gestion des documents multimédia composés et la gestion des disparitions des adaptateurs impliquent des coûts pendant le processus d'adaptation. Ceci nous amène à montrer dans cette section l'apport de la parallélisation des adaptations par rapport aux coûts induits par ces trois fonctionnalités. Le tableau qui suit est un récapitulatif du nombre de messages échangés lors de l'exécution du protocole de négociation et de la gestion des disparitions des adaptateurs :

	Protocole de négociation	Gestion des états / disparitions des adaptateurs
Nombre de messages	$NA \times 5 + \sum (NME_i \times 4)$	$\sum ((\lceil (DE_i / 30) \rceil - 1) \times 2) + NAD_i \times 4)$

Tableau 12 : Récapitulatif du nombre de messages échangés lors de la gestion des adaptateurs

où :

« NME_i » : nombre d'adaptateurs ayant envoyé un message d'exception correspondant au processus d'adaptation *« i »* du graphe d'adaptation physique.

« NAD_i » : nombre d'adaptateurs disparus en réalisant l'adaptation i.

« DE_i » : durée (en secondes) estimée de l'adaptation du média i.

« NA » : nombre d'adaptateurs à instancier (d'après le graphe d'adaptation).

Afin d'illustrer l'avantage de la parallélisation des adaptateurs, nous donnons l'exemple d'un document multimédia composé d'une vidéo, d'une image et d'un texte et nécessitant trois adaptations. Le tableau suivant résume ces caractéristiques.

	Type média	Taille média	Type adaptation	Temps adaptation (ms)
Composition du document multimédia	Vidéo (AVI)	487 MO	Transcodage (MOV)	153520
	Image (JPEG)	1024 x 768 pixels	Réduction (400 x 300 pixels)	1000
	Vidéo (AVI)	250 MO	Transcodage (MOV)	80950

Tableau 13 : Caractéristiques du document multimédia composé servant à illustrer l'apport de la parallélisation

En appliquant les formules présentées précédemment, nous donnons dans ce qui suit le nombre de messages nécessaires lors du déroulement du protocole de négociation et d'acceptation dans les deux cas : avec ou sans exception. Nous calculons également le nombre de messages échangés lors de la gestion des disparitions suivant les deux cas : aucune disparition d'adaptateurs et une disparition par adaptation. Le tableau suivant illustre le nombre de messages envoyés et montre la durée de chaque fonctionnalité (si nous considérons que la durée d'envoi et de réception des messages SOAP est égale à 60 ms) :

	Nombre de messages	**Durée d'envoi et de réception des messages SOAP (ms)**
Protocole de négociation et d'acceptation (sans exception)	3 x 5 = 15	450
Protocole de négociation et d'acceptation (avec une exception par adaptation)	3 x 5 + 4 + 4 + 4 = 27	810
Gestion des états (sans disparition)	10 + 4 + 0 = 14	420
Gestion des états (avec une disparition par adaptation)	(10 + 4) + (4 + 4) + 4 = 26	780

Tableau 14 : Le nombre de messages échangés lors du protocole de négociation et de la gestion des états des adaptateurs

La figure qui suit compare l'architecture d'adaptation centralisée (de référence) avec PAAM. La figure schématise la durée des fonctionnalités deux architectures. La gestion de l'adaptation distribuée ne concerne que PAAM et comprend le protocole de négociation et d'acceptation (cas d'une exception par adaptation) ainsi que la gestion des états et des disparitions des adaptateurs (cas d'une disparition par adaptation).

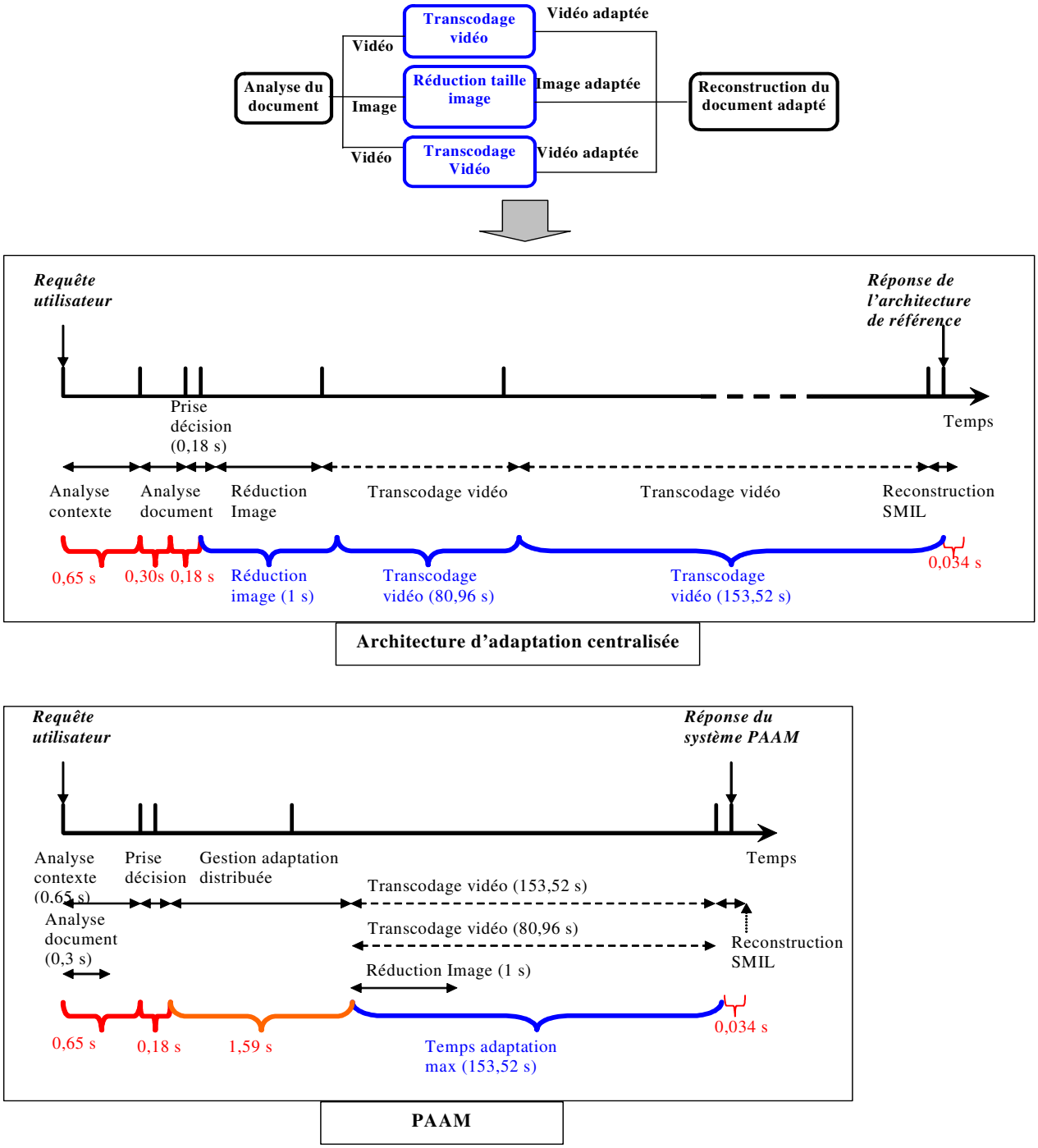

Figure 53 : Illustration des durées des différentes étapes du processus d'adaptation dans PAAM et dans l'architecture de référence

La Figure 53 montre que la parallélisation des adaptations réalisée dans PAAM réduit le temps total d'adaptation de « 80,96 + 1 » s par rapport à l'architecture d'adaptation centralisée. Ainsi, ce gain en temps masque les coûts impliqués par la gestion distribuée des adaptateurs en utilisant les services Web.

En effet, le coût de l'adaptation distribuée (1,59 s) est très petit par rapport au gain obtenu grâce à la parallélisation (81,96 s) (de l'ordre de 1 sur 50).

À noter également que le coût de la gestion des documents multimédia composés est également masqué par la parallélisation des adaptations.

4 Conclusion

Ce chapitre a montré qu'une adaptation distribuée des médias est avantageuse, et que dans ce cas, le nombre de messages nécessaire à la distribution est compensé par le gain obtenu en parallélisant les processus d'adaptation. Ceci évite de surcharger, voire d'effondrer une seule machine. Bien que le nombre de messages envoyés dans le cas d'une adaptation distribuée est sensiblement plus élevé que le nombre de messages envoyés dans le cas d'une adaptation centralisée, la première solution passe à l'échelle, ce qui compense cette différence. En effet, le système d'adaptation centralisé supporte un nombre limitée d'utilisateurs, et donc d'adaptations. Si ce nombre est dépassé, il n'est pas possible de déléguer le traitement à d'autres machines, ce qui est, au contraire, possible dans une solution d'adaptation distribuée comme PAAM

Ce chapitre a également montré que les fonctionnalités de PAAM telles que la gestion du contexte, la gestion des documents multimédia composés, la gestion de la prise de décision et la gestion de l'adaptation possèdent des coûts très petits par rapport aux coûts induits par certaines adaptations telles que le transcodage de vidéo de grande taille.

Chapitre 8

Conclusion et perspectives

1 Rappel des objectifs

L'échange de documents multimédia composés de médias tels que des vidéos, des images ou du texte, est l'une des applications P2P les plus populaires de l'Internet. Notre objectif principal est que chaque utilisateur puisse consommer ces contenus quel que soient ses préférences (par exemple : sa langue parlée, son handicap ou son lecteur multimédia préféré) ou les capacités de son terminal (par exemple : la taille de l'écran du terminal ou les lecteurs multimédia présents) : en d'autres termes, quel que soit son contexte. Idéalement, tout usager d'Internet devrait pouvoir accéder à ces contenus et les recevoir dans un format adapté au contexte dans lequel il travaille.

Il faut donc un système qui adapte les contenus multimédia par rapport au contexte des usagers.

1.1 Adapter…oui mais où ?

Les usagers possèdent de plus en plus des terminaux à capacités réduites telles que les assistants personnels (par exemple les blackberry). De ce fait, nous excluons une adaptation côté client (ou consommateur final).

L'adaptation côté serveur, où à la source du document multimédia, nécessite l'implémentation de modules supplémentaires qui n'est pas toujours possible (capacité des serveurs limitées parfois) et qui peut créer une charge supplémentaire indésirable sans rapport avec la production de contenus multimédia. L'adaptation côté serveur est ainsi à son tour exclue.

1.2 L'adaptation distribuée

L'adaptation par un ou plusieurs intermédiaires répond le mieux aux besoins de passage à l'échelle et d'extensibilité. L'intermédiation ainsi réalisée apporte une valeur ajoutée en évitant de charger l'utilisateur final et la source du document de tâches spécifiques consommatrices de ressources sans rapport direct avec le service final offert.

Ainsi, notre but est de concevoir et implémenter une architecture de fourniture de contenus multimédia intégrant une adaptation distribuée au niveau de plusieurs nœuds intermédiaires présents dans le réseau afin de répondre à ces besoins : passage à l'échelle, extensibilité, robustesse et évolutivité.

2 Contributions et bilan scientifique

Dans ce rapport de thèse, nous avons présenté PAAM, une architecture qui adapte, dynamiquement, des documents multimédia composés au contexte des utilisateurs. PAAM est une architecture orientée services. Les ressources d'adaptation sont mises à disposition du système d'adaptation PAAM par des nœuds librement répartis sur le réseau.

Sur la base d'une requête utilisateur contenant un lien vers son contexte et vers le document multimédia composé qu'il souhaite consommer, le gestionnaire de contexte et le gestionnaire de documents composés analysent respectivement le contexte utilisateur et le document multimédia composé. Le planificateur récupère les informations analysées par les deux gestionnaires et décide des adaptations à effectuer pour adapter le document multimédia au contexte de l'utilisateur. Il en résulte un graphe d'adaptation qu'il transmet au gestionnaire d'adaptation. Ce dernier recherche les adaptateurs utiles, les compose pour réaliser des opérations d'adaptation complexes et pilote l'adaptation. Pour chaque document, le gestionnaire d'adaptation permet de redéfinir l'ordonnancement des tâches entre des adaptateurs choisis sur le réseau ; cela facilite la prise en compte des changements dans l'environnement des usagers et dans celui des adaptateurs.

Notre première contribution est ainsi la conception et la réalisation d'un tel système. Nous avons choisi d'implémenter le système d'adaptation PAAM ainsi que les adaptateurs à l'aide de la technologie des services Web. Ceci facilite le déploiement de *PAAM* à large échelle, l'extensibilité et l'évolutivité.

Notre deuxième contribution est la gestion des adaptateurs qui se décline de la façon suivante :

- Nous avons, tout d'abord, catégorisé un ensemble d'adaptateurs et proposé une identification unique pour chacun d'eux. Cette identification repose sur des adresses URN. Nous avons proposé, par la suite, une version étendue des descriptions WSDL des services Web d'adaptation afin d'y rajouter certains paramètres tels que les « mimetypes » d'entrée et de sortie des adaptateurs, leur identification unique et le prix du service.
- Nous avons également proposé un protocole de négociation et d'acceptation qui établit une communication entre le gestionnaire d'adaptation et le ou le(s) adaptateur(s) à instancier. Ce protocole permet au gestionnaire d'adaptation de choisir le meilleur adaptateur parmi plusieurs répondant aux mêmes critères. La description des adaptateurs permet également de bien choisir un adaptateur.
- Nous avons identifié la nécessité pour un adaptateur d'implémenter une politique d'acceptation lui permettant d'accepter ou de refuser une adaptation entrante. Nous avons donné quelques exemples de ces politiques.

- Nous proposons également des solutions pour gérer les éventuelles disparitions d'adaptateurs.

Pour conclure, nous avons réalisé des tests ainsi qu'une étude des coûts induits par la distribution de l'adaptation et par l'utilisation des services Web. Les tests et études réalisés ont montré que la parallélisation des adaptations masque les coûts induits par l'utilisation des services Web et par la gestion des adaptateurs distribués.

3 Perspectives et travaux futurs

La modularité de l'architecture PAAM permet d'envisager de nombreuses améliorations indépendamment les unes des autres. Nous présentons dans cette section les perspectives de nos travaux.

L'architecture de PAAM comprend plusieurs blocs fonctionnels. Nous nous intéressons plus particulièrement au planificateur, au gestionnaire d'adaptation et au service Web d'adaptation.

3.1 Travaux futurs relatifs au planificateur

La version courante du planificateur implémente un algorithme de prise de décision qui repose sur des politiques d'adaptation et qui génère un graphe d'adaptation qui est envoyé au gestionnaire d'adaptation.

Nous envisageons de fournir des outils permettant de générer, à partir de nouvelles descriptions d'adaptateurs, des politiques d'adaptation et de doter le planificateur d'un moteur d'inférence intégrant de manière automatique ces politiques.

3.2 Travaux futurs relatifs au gestionnaire d'adaptation

Le gestionnaire d'adaptation peut être amélioré en intégrant un moteur BPEL capable de composer et d'orchestrer les services Web d'adaptation. Le graphe d'adaptation doit alors être traduit en processus BPEL et donné en entrée du moteur BPEL. L'utilisateur de ce standard permet plus d'automatisation dans l'exécution du graphe.

À noter qu'un moteur BPEL ne sera utilisé par le gestionnaire d'adaptation qu'après avoir choisi les bons adaptateurs en appliquant le protocole de négociation et d'acceptation.

3.3 Travaux futurs relatifs à un adaptateur

Un adaptateur dans PAAM est implémenté à l'aide des services Web. Nous avons vu dans le chapitre 5 que nous avons étendu les descriptions WSDL afin de décrire un service Web d'adaptation de telle sorte qu'il puisse être facilement recherché et instancié. Il est ainsi intéressant que les générateurs, qui prennent en entrée des classes afin de fournir les descriptions WSDL correspondantes (par exemple l'outil WSGEN fournit avec la pile services Web JAX-WS [JAXWS]), puissent être modifiés afin de produire des descriptions WSDL étendues répondant aux spécifications de PAAM. Cette perspective

ne peut être envisagée que si nous soumettons notre contribution à l'organisme de standardisation des services Web : le W3C.

Une autre amélioration des adaptateurs concerne les politiques d'acceptation qu'ils implémentent. En effet, nous avons proposé dans ce manuscrit des politiques d'acceptation simplifiées qui nous ont permis d'implémenter notre prototype. Nous envisageons par la suite de proposer des politiques d'acceptation plus complexes qui tiennent compte de plusieurs critères tels que le poids de l'adaptation, la charge CPU de l'adaptateur, la taille du média à adapter et le nombre d'adaptations en cours.

3.4 Autres travaux envisagés

D'autres travaux futurs concernent le déploiement de PAAM à très large échelle et la réalisation de tests visant à vérifier que l'utilisation des services Web réponde bien à la problématique de la montée en charge et du passage à l'échelle.

Pour conclure, nous envisageons, à plus long terme, d'étendre PAAM au streaming.

Liste des publications

Z. Kazi-Aoul, I. Demeure et J-C. Moissinac. *PAAM: A Web Services Oriented Architecture for the Adaptation of Composed Multimedia Documents*, dans les actes de la conference PDCN'08, 12-14 Février 2008.

J-C. Moissinac, Z. Kazi-Aoul, Isabelle Demeure. *Sémantique pour la composition de Web Services d'adaptation multimédia*, dans les actes de la conférence NOTERE'2007 Marrakech, Maroc, Juin 4-8, 2007.

Z. Kazi-Aoul, I. Demeure et J-C. Moissinac, *Vers un système d'adaptation de documents multimédia dans un environnement P2P*. Conférence sur les Nouvelles Technologies de la Répartition (NOTERE'06), Toulouse, France 6-9 Juin 2006.

Z. Kazi-Aoul, I. Demeure et J-C. Moissinac, *Towards a Peer-to-peer Architecture for the provision of Adaptable Multimedia Composed Documents*. DFMA (Distributed Frame for Multimedia Applications), Penang, Malaysie, 14-17 Mai 2006.

Zakia Kazi-Aoul, Isabelle Demeure, J-C. Moissinac, *Une architecture pour la fourniture de services multimédia adaptables : Illustration par un scénario*, UBIMOB'04 (Ubiquité et mobilité), Nice, Juin 1-3 2004

Isabelle Demeure et al,*« Adaptable Service architectures »*, Délivrable 1.4.2 du projet IST-2001-38835 ANWIRE, 31 Août 2004.

Z. Kazi-Aoul, S. Ferraz, O. Fouial and I. Demeure. *CAAS: an architecture for component-based adaptable service provision*. 2nd ANWIRE Workshop on Wireless, Mobile & Always Best Connected. Mykonos, Grèce, 25-26 Septembre 2003.

Bibliographie

[Ambient] Site officiel du projet Ambient Networks http://www.ambient-networks.org/.

[APACHE] Projet Apache http://www.apache.org/.

[AXIS] Projet Apache AXIS : http://ws.apache.org/axis/.

[AXIS2] Projet Apache AXIS 2 : http://ws.apache.org/axis2/.

[BDM93] Michael Barborak, Anton Dahbura, Minoslaw Malek : The consensus problem in fault-tolerant computing, ACM Computing Surveys (CSUR), Volume 25 , Issue 2, Juin 1993, pages: 171 - 220

[BHK03] Böszörményi L., Hellwagner H., Kosch H., Libsie M., Podlipnig S., « Metadata driven adaptation in the ADMITS project », EURASIP Signal Processing : Image Communication Journal, vol 18, n° 8, Septembre 2003, pages : 749-766.

[BVH03] Ian Burnett, Ric Van de Walle, Keith Hill, Jan Bormans, Fernando Pereira, MPEG-21 : Goals and Achievements, dans IEEE Multimedia, Octobre-Novembre 2003, vol 10 N°6, pages 60-70.

[CCPP] Site officiel de CC/PP : http ://www.w3.org/Mobile/CCPP/.

[Chauv02] Jean-Marie Chauvet : Services Web avec SOAP, WSDL, UDDI, ebXML..., Eyrolles, Mars 2002.

[Corba] Site officiel de Corba : http://www.corba.org/.

[Dey00] Dey, A.K., *Enabling the Use of Context in Interactive Applications,* in the CHI 2000 Doctoral Consortium, in the Proceedings of the 2000 Conference on Human Factors in Computing Systems (CHI 2000), 1-6 Avril 2000, pages : 79-80.

[DKo03] Mario Doller et Harald Kosch, MPEG-7 Multimedia Data Cartridge, dans Proceedings of the SPIE Conference on Multimedia Computing and Networking 2003 (MMCN03), Santa Clara, CA, 29-31 Janvier, 2003.

[DMa02] Neil Day and José M. Martinez, Introduction to MPEG-7 (v4.0), ISO/IEC JTC1/SC29/WG11 N4675, Jeju, Mars 2002.

[DOM] Document Object Model (DOM) Level 3 Core Specification, Version 1.0, W3C Recommendation 07 April 2004, http://www.w3.org/TR/DOM-Level-3-Core.

[DUBLIN] Site officiel de Dublin Core : http://dublincore.org/.

[ECMA] ECMAScript Language Specification, 3rd edition (December 1999), Standard ECMA-262, http://www.ecma-international.org/publications/standards/Ecma-262.htm.

[EJB] Site de Enterprise Java Beans : http://java.sun.com/products/ejb/.

[Emule] Site officiel de Emule : http://www.emule-project.net/home/perl/ general.cgi?l=13.

[Erl04] Thomas Erl, Service Oriented architecture : A field Guide to Integrating XML and Web Services, Prentice Hall.

[Gar99] Felix C. Gärtner : Fundamentals of fault-tolerant distributed computing in asynchronous environments, ACM Computing Surveys (CSUR), Volume 31 , Issue 1, Mars 1999, pages : 1-26.

[GBP05] G. Berhe, L. Brunie, JM. Pierson : Distributed Content Adaptation for Pervasive Systems. Dans les actes de IEEE International Conference on Information Technology, ITCC 2005, 4-6 Avril 2005, Las Vegas, Nevada, USA, Vol.2 pages : 234-241.

[GCK04] Gioia P., Cotarmanac'h A., Kamyab K., Goulev P., Mamdani E., Wolf I., Graffunder A., Panis G., Hutter A., Difino A., Negro B., Kimiaei M., Concolato C., Dufourd J-C., DiGiacomo T., Joslin C., Thalmann N., « ISIS : Intelligent Scalability for Interoperable Services », Actes de la 1st European Conference on Visual Media Production (CVMP), Londres, 15 et 16 Mars, 2004, pages : 295-304.

[GLASS] Projet GlassFish https://glassfish.dev.java.net/.

[GTB02] Balazs Goldschmidt, Roland Tusch, et László Böszörményi, A mobile Agent-based Infrastructure for an Adaptive Multimedia Server, 4ème workshop autrichien – hongrois sur les Distributed And Parallel SYStems (DAPSYS), 2002, pages 141 à 148.

[GTB03] Balazs Goldschmidt, Roland Tusch, László Böszörményi, Hermann Hellwagner et Peter Schojer, Offensive and Defensive Adaptation in Distributed Multimedia Systems**,** Rapport technique No TR/ITEC/03/2.03, université de Klagenfurt, Autriche, Février 2003.

[Gur04] Anura Gurugé : Web Services, Theory and Practice, ELSEVIER DIGITAL PRESS, ISBN: 1-55558-282-6, 15 Mars 2004, 371 pages.

[Her94] Jean-Marc Herellier : Le Multimédia, 791 pages, édité en Septembre 1994, éditions SYBEX.

[IDL] Document de référence sur IDL : http://www.omg.org/docs/formal/03-02-01.pdf.

[ISIS02] Site du projet ISIS http://isis.rd.francetelecom.com/

[JAKA] Projet Apache Jakarta http://jakarta.apache.org/.

[JAXB] Page d'accueil de JAXB : https://jaxb.dev.java.net/.

[JAXWS] Projet Java XML for Web Services https://jax-ws.dev.java.net/.

[JXTA] Site officiel de JXTA : https://jxta.dev.java.net/.

[Kim05a] Kimiaei-Asadi M., Dufourd J-C, « Context-aware Semantic Adaptation of Multimedia Presentations », Actes de la IEEE International Conference on Multimedia & Expo (ICME 2005), Amsterdam.

[Kim05b] Kimiaei Asadi, Mariam : Adaptation de Contenu Multimédia avec MPEG-21: Conversion de Ressources et Adaptation Sémantique de Scènes. Thèse de Doctorat soutenue le 30 Juin 2005 à l'ENST.

[LRe05] Lapayre J-C., Renard F., « Appat : a New Platform to Perform Global Adaptation », Actes de la 1st IEEE International Conference on Distributed Frameworks for Multimedia Applications (DFMA'2005), Besançon, 6-9 février 2005, pages : 351-358.

[Lem04] Lemlouma T., Layaïda N., « Context-Aware Adaptation for Mobile Devices », Actes de la IEEE International Conference on Mobile Data Management (MDM2004), Berkeley, 19-22 janvier, 2004.

[LHC03] Rainer Lienhart, Igor Kozintsev, Yen-Kuang Chen, Matthew Holliman, Minerva Yeung, Andre Zaccarin, and Rohit Puri: Challenges in Distributed Video Management and Delivery. *Handbook of Video Databases.* CRC Press, Boca Raton, Florida , pages : 961-990, 2003.

[LRA04] Jean-Claude Laprie, Carl Landwehr, Algirdas Avizienis, Brian Randell : Basic Concepts and Taxonomy of Dependable and Secure Computing, IEEE Transactions on Dependable and Secure Computing, Volume 1 , Issue 1, Janvier 2004, pages : 11-33

[Mag02] Thèse de João Miguel da Costa Magalhães, Universal Multimedia Content Based on the MPEG-7 Standard, soutenue en Juin 2002 à l'institut supérieur technique de l'université technique de Lisbonne.

[Mat06] Matjaz B.Juric, Benny Mathew, Poornachandra Sarang : Business Process Execution Language for Web Services, An architect and developer's guide to orchestrating web services using BPEL4WS, deuxième édition, Janvier 2006, Packt Publishing Ltd.

[MKP02] José M. Martínez, Rob Koenen, and Fernando Pereira. MPEG-7: The generic Multimedia Content Description Standard, part 1, dans IEEE multimedia volume 9, issue, Avril 2002, pages 78-87.

[MKS06] Bertrand Mathieu, Michael Kleis, Meng Song : A P2P Approach for the selection of Media Processinf Modules for Service Specific Overlay Networks, dans les actes de la conference AICT-ICIW 2006, 19-25 Février 2006, pages 103.

[MLS04] R. Montanari, E. Lupu, C. Stefanelli : Policy based Dynamic Reconfiguration of Mobile-code Applications, dans IEEE Computer, vol 34, Juillet 2004, pages 73-80.

[MPe04] Da Costa Magalhães J.M., Pereira F., « Using MPEG standards for multimedia customization », Signal Processing : Image Communication, vol. 19, n° 5, Mai 2004, pages : 437-456.

[MPEG21] ISO/IEC JTC1/SC29/WG11/N6168, « MPEG-21 Part7: Digital Item Adaptation », Mars 2004.

[OASIS] Site officiel de OASIS : http://www.oasis-open.org/home/index.php.

[OPE02] Site du projet OPERA http://opera.inrialpes.fr/.

[OWL] Spécification de OWL : http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[OWLS] Spécification de OWL-S : http://www.daml.org/services/owl-s/1.0/.

[Papa03] Papazoglou, M.P. : Service-oriented computing: concepts, characteristics and directions, in Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE), 2003, pages 3- 12.

[Parth06] Parthasarathy Ranganathan, Erik Geelhoed, Meera Manahan, and Ken Nicholas, Energy-aware user interfaces and energy-adaptive displays, Hewlett-Packard, publié par the IEEE Computer Society, Mars 2006.

[PURL] Le site officiel de PURL http://purl.org/.

[RBu04] Letian Rong, Ian Burnett, Dynamic Resource Adaptation in a Heterogeneous Peer-to-peer Environment, Consumer Communications and Networking Conference (CCNC 2005), Second IEEE, January 3rd to 6th, 2005, pages : 416 - 420.

[RDF] W3C, “Resource Description Framework” http://www.w3.org/TR/REC-rdf-syntax/.

[SATO06] System Design of SATO & ASI, déliverable D12-F.1, http://www.ambient-networks.org/Files/deliverables/D12-F.1_PU.pdf.

[SBH03] Peter Schojer, László Böszörményi, Hermann Hellwagner, Bernhard Penz et Stefan Podlipnig, Architecture of a Quality Based Intelligent Proxy (QBIX) for MPEG4 Videos**,** dans les proceedings du journal ACM 2003 de WWW’03 (World Wide Web Conference), Budapest, Hongrie, Mai 2003, pages : 394-402.

[SCh01] Nary Subramanian, Lawrence Chung, Software Architecture Adaptatbility: An NFR Approach, Proc., Int. Workshop on Principles of Software Evolution (IWPSE`01), 10-11 Septembre, Vienne, Austriche. ACM Press, 2001, pages : 52-61.

[SDL] Spécification de SDL : http://www.sdl-forum.org/SDL/index.htm.

[SEc07] Sokol Joachim, Eckert Klaus-Peter : MCDN: Multimedia Content Discovery and Delivery, , 2007. dans Eighth International Symposium on Autonomous Decentralized Systems (ISADS '07), 21-23 Mars 2007, pages : 411 - 420

[SMIL] Page d’accueil de SMIL : http://www.w3.org/TR/REC-smil/.

[SOAP] Spécification de SOAP : http://www.w3.org/TR/soap/.

[TOMCAT] Projet Apache Tomcat : http://tomcat.apache.org/.

[UAPROF] Forum WAP Ltd, “WAG-User Agent Profile Specification 2.0” 30 Mai 2001.

[UDDI] Site officiel de UDDI : http://www.uddi.org/.

[URN] Spécification de URN : http://www.ietf.org/rfc/rfc2141.txt.

[Vet04] Vetro A., « MPEG-21 Digital Item Adaptation : Enabling Universal Multimedia Access », IEEE Multimedia, janvier-mars 2004, vol. 11, n° 1, pages : 84-87.

[WAM03] Site du projet WAM : http://wam.inrialpes.fr/index.fr.html

[WAP] FORUM WAP Ltd, “WAP 2.0 Technical White Paper”, Août 2001.

[WSAD] Site officiel de WS-Addressing : http://www.w3.org/Submission/ws-addressing/.

[WSDL] Page d’accueil de WSDL : http://www.w3.org/TR/wsdl.

[WSPROJ] Projet Apache <Web Services/> http://ws.apache.org/.

[WURFL] Pite officiel de WURFL : http://wurfl.sourceforge.net/.

[XHTML] Spécification W3C de XHTML : http://www.la-grange.net/w3c/xhtml1/.

[ZYX] Susanne Boll, Wolfgang Klas, ZyX : A Multimedia Document Model for Reuse and Adaptation of Multimedia Content, in IEEE Transactions on Knowledge and Data Engineering, Volume 13 , Issue 3, Mai 2001, pages : 361 - 382.

Annexe 1
Wesemac : Web Services Multimédia

1 Objectif du projet WESEMAC

Le projet WESEMAC est une plateforme générique spécifiquement conçue pour l'implémentation d'appels distants sur des objets dédiés aux services multimédia (par ex. conversion de format et de présentation d'entités multimédia telles que des images et des vidéos). Afin d'être accessible depuis tout nœud sur l'Internet, ces services d'adaptation de contenu sont fournis sous forme de services Web.

2 Framework d'exécution

2.1 Environnement de développement et d'exécution

L'application WESEMAC est codée en utilisant les APIs du socle J2EE 1.4.2_05.

L'exécution de l'application WESEMAC nécessite une JRE (Java Runtime Environment) correspondant à une version au moins égale à 1.4.2_05

Des modifications peuvent être apportées au code de WESEMAC dans un environnement de développement JDK (Java Development Kit) dont la version est au moins égal à 1.4.2_05.

2.2 Logiciels

2.2.1 Tomcat 5.0.28

Le serveur Tomcat du groupe Apache sera utilisé en tant que conteneur de la servlet AXIS 2. Le serveur Tomcat sera le proxy utilisé pour recevoir les requêtes de service web destinées à la plateforme WESEMAC.

2.2.2 AXIS 2 (axis 2 0.92)

AXIS 2 sera utilisé en tant pile de services Web.

AXIS 2 a été préféré à AXIS car AXIS 2 implémente la gestion de la synchronisation et le support des échanges de messages au format MTOM.

2.3 Librairies de conversion d'entités média

La plateforme développée pourra proposer des services d'adaptation pour les objets média image, audio et vidéo. Pour cela, les services Web implémentés devront utiliser des librairies ou des utilitaires de manipulation d'entités média, entre autres JMF et JAI.

2.3.1 JMF API (version 2.2.1)

JMF (Java Media Framework) est un framework Java d'affichage d'entités média présentant une contrainte temps réel. JMF supporte les formats MPEG-1, MPEG-2, QuickTime, AVI, WAV, AU, et MIDI.

JMF fournit une architecture et un protocole d'échange pour manipuler aisément toutes sortes de contenus multimédia

On peut au choix :

- Faire abstraction des médias grâce aux classes de haut niveau de JMF : Présentation and Processing API.
- Avoir un accès direct au contrôle des média pour répondre à des besoins spécifiques. On utilisera alors les classes de bas niveau JMF Plug-in API.

L'API JMF est constitué principalement d'interfaces définissant le comportement et les interactions entre les objets définis pour capturer, traiter et présenter les données multimédia. On distingue principalement :

- Objet CaptureDeviceManager (périphériques de capture) tel que webcam, micro, etc.
- Objet DataSource (Source de données) : flux de données qui encapsule un média. Peut être instancié à partir d'un fichier local, d'un micro, d'un flux venant du net, etc.
- Objet Player : prend en entrée un flux de données (DataSource) (audio et/ou vidéo) et effectue un traitement pour pouvoir présenter ces données sur un écran ou des haut-parleurs.
- Objet Processor (Processeur) hérite de Player. Contrôle du traitement. Le flux peut être transmis à un Player ou à un autre Processor.
- Objet Datasink récupère le média d'une source de données et le redirige vers une destination. Un Datasink permet par exemple d'enregistrer un média dans un fichier.
- Objet Format décrit le format d'un media. (format d'encodage et type de données).
- Objets Manager : facilitent l'intégration des objets vu ci-dessus.
- Manager : crée des Players, DataSource, Processor et Datasink. ex : création d'un lecteur à partir d'une source de données.
- PackageManager : gère l'ensemble des packages disponibles dans les classes de JMF, à savoir des lecteurs, processeurs, sources de données et DataSinks spécifiques.

- CaptureDeviceManager : gère l'ensemble des périphériques de capture disponibles.
- PlugInManager : gère l'ensemble des composants JMF de bas niveau, comme les objets Multiplexers, Demultiplexers, Codecs, Effects, et Renderers.
- SessionManager : obligatoire pour gérer plusieurs flux.
- Utilisation de l'interface Controler : définit l'état des objets, et gère les transitions entre les différents états de ces objets ainsi que les contraintes liées au temps.

2.3.1.1 Architecture RTP

Real time Transport Protocol (RTP) est un protocole qui permet de transporter des flux de données qui ont des propriétés temps réel, il se situe uniquement au niveau de l'application et utilise des protocoles sous-jacents de transport TCP ou UDP.

La transmission de média en temps réel est assurée par l'implémentation de RTP dans l'API JMF.

Le protocole utilisé pour ce type de transfert est le RTP, s'appuyant sur UDP ou RTCP s'appuyant sur TCP.

L'objet SessionManager sert à coordonner une session RTP.

2.3.2 JAI API (version 1.1.2_01)

Pour réaliser les traitements d'images, nous avons choisi d'utiliser l'API Java de Sun Microsystems JAI (Java Advanced Imaging).

JAI présente un modèle de programmation simple. En effet, JAI encapsule les formats de données d'image et les invocations de méthodes à distance dans des objets images réutilisables, ce qui permet de traiter de la même façon un fichier ou un flux de données en temps réel.

Afin d'optimiser le temps de traitement, JAI peut utiliser du code natif pour réaliser certaines opérations sur quelques architectures matérielles. Cependant, il y a toujours la possibilité de forcer l'utilisation du code Java pour être certain d'obtenir le même résultat.

2.3.3 Architecture fonctionnelle de Wesemac

Nous présentons dans ce qui suit ce qui a servi de base pour implémenter le service Web d'adaptation générique Wesemac.

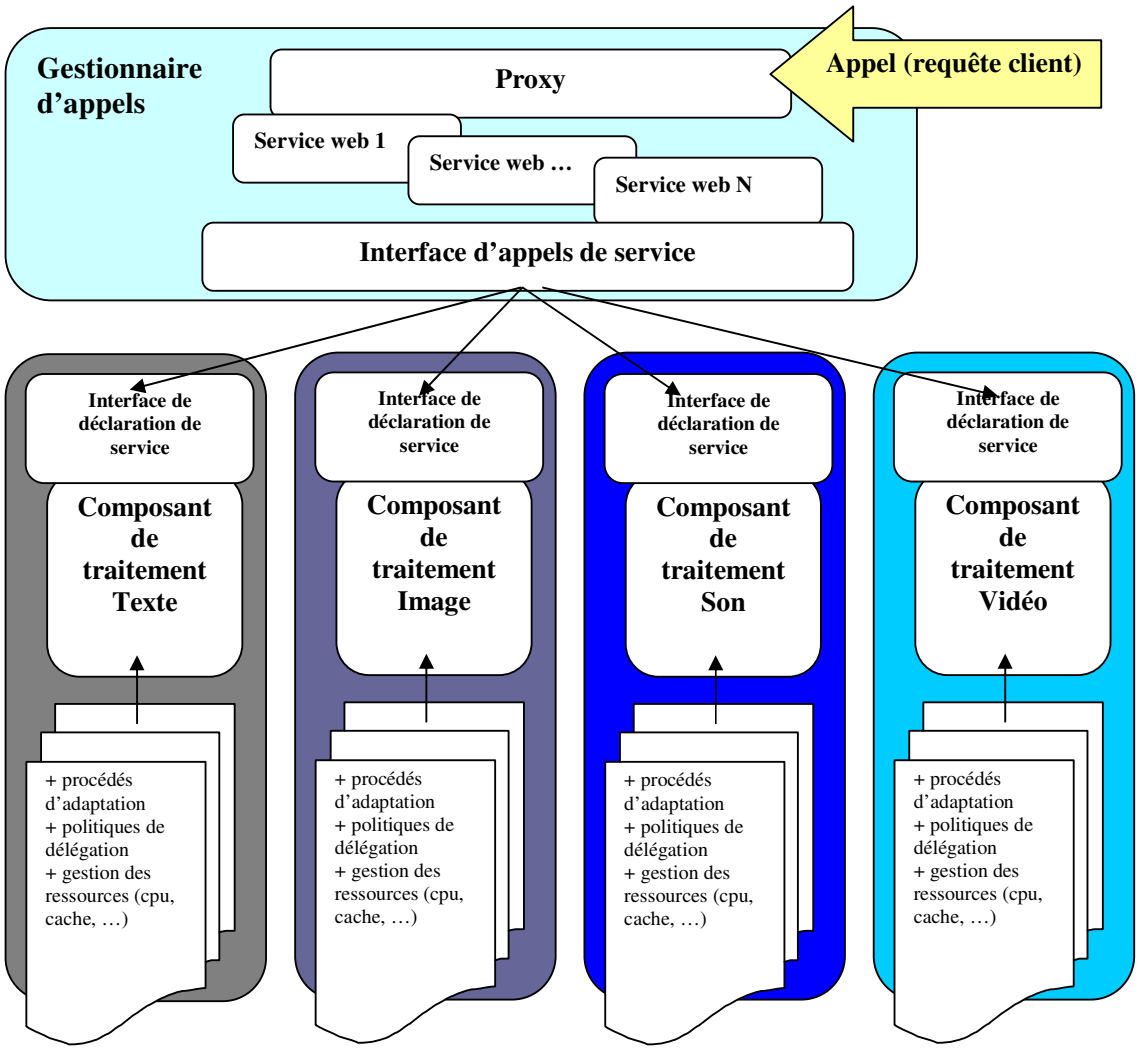

Figure 54 : architecture fonctionnelle de Wesemac

Annexe 2
Contraintes liées aux adaptateurs

1 Rappel du contexte d'utilisation des adaptateurs

PAAM est un système d'adaptation de documents multimédia composés où les adaptateurs, payants ou non, sont offerts aussi bien par des prestataires de services que par des utilisateurs. La distribution de l'adaptation permet un gain de temps en parallélisant les opérations d'adaptation et en évitant de surcharger un seul adaptateur offert par un seule machine.

Ces prestataires ou utilisateurs doivent respecter certaines contraintes et spécifications pour assurer une cohérence entre leur(s) adaptateur(s) et le système d'adaptation PAAM auquel ils sont rattachés.

2 Contraintes et spécifications des adaptateurs PAAM

Nous avons opté pour les services Web Afin d'implémenter PAAM. Les services Web permettent aux systèmes d'adaptation et aux adaptateurs de communiquer et de parler un seul langage. Cette section a pour but de spécifier les contraintes communes à tous les adaptateurs et la manière avec laquelle ils doivent se comporter pour participer à un processus d'adaptation au sein de PAAM.

2.1 Entrées / sorties des adaptateurs

Un adaptateur PAAM est décrit à l'aide d'une description WSDL étendue dans laquelle il est contraint d'indiquer les paramètres suivants :

- L'identification « URN » de l'adaptateur.
- Les mimetypes d'entrée supportés.
- Les mimetypes de sorties possibles.
- L'URL de l'adaptateur.
- Le prix du service qu'il propose.

Un adaptateur PAAM qui ne fournit pas ces informations ne peut pas être utilisé par PAAM car le système d'adaptation utilise ces paramètres pour rechercher et instancier tout adaptateur.

Les paramètres d'appel d'un service Web d'adaptation (voir la méthode *adapt* de la classe *Interface.java* présentée dans la section suivante) sont :

- L'URL du média
- Le mimetype d'entrée du média à adapter.
- Le mimetype nécessaire à la sortie d'un transcodeur (si l'adaptateur est un transcodeur) ou la langue souhaitée à la sortie d'un traducteur (si l'adaptateur est un traducteur).
- Le prix de l'adaptateur.
- La liste des mimetypes finaux supportés par le terminal et/ou préférés par l'utilisateur
- La liste des paramètres supplémentaires :
- MediaMaxSize : correspond à la taille maximum du média à adapter.
- MediaMaxHeight : correspond à la hauteur maximum du média à adapter s'il s'agit d'une image ou d'une vidéo.
- MediaMaxWidth : correspond à la largeur maximum du média à adapter s'il s'agit d'une image ou d'une vidéo.
- ColorSupport : détermine si le terminal supporte la couleur ou pas.

L'adaptateur utilise tout ou partie de ces paramètres mais doit retourner obligatoirement l'URL du média adapté.

2.2 Asynchronisme et gestion de suivi de session des adaptateurs

L'asynchronisme est un mode de communication entre deux entités. Dans le cadre de PAAM, il s'agit d'un mode de communication entre le gestionnaire d'adaptation et l'adaptateur lors de du protocole de négociation et d'acceptation. Les services Web permettent ce type de communication grâce à WS-Addressing [WSAD].

Nous rappelons que le but de la spécification WS-Addressing est de fournir les moyens d'identifier un service Web et une manière d'utiliser de tels identifiants dans des messages SOAP servant de moyen de communication entre un client et un serveur de service Web.

Un adaptateur pouvant être instancié par plusieurs gestionnaires d'adaptation à la fois, il est important de gérer l'état de session avec un adaptateur. Ce concept est très corrélé avec celui de l'asynchronisme.

En terme d'implémentation et en utilisant JAX-WS [JAXWS], chaque adaptateur doit implémenter deux classes : *Assigner.java* et *Interface.java.*

Voici à quoi ressemble la classe *Assigner.java* :

```
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import javax.xml.ws.wsaddressing.W3CEndpointReference;
```

```
@WebService
public class Assigner
 {
    public Assigner() {
   }
      @WebMethod
    public W3CEndpointReference getRef( @WebParam(name = "id")String id) {
            return Interface.manager.export(new Interface(id));
   }
       @WebMethod
    public  void free(@WebParam(name = "id" )String id) {
    Interface.manager.unexport(new Interface(id));
   }
}
```

La classe *Assigner.java* est commune à tous les adaptateurs. Les notations @WebService et @WebMethod proviennent des spécifications de JAX-WS et signifient respectivement que cette classe est la classe référence d'un service Web et que ce service Web offre deux méthodes. La première méthode a pour but de créer une référence unique de l'adaptateur en invoquant la méthode *getRef* qui fait appel à la méthode *manager.export* que nous retrouvons dans la classe *Interface.java.* Le résultat de la méthode *getRef* est un objet *W3CEndpointReference* qui identifie de manière unique l'instance de l'adaptateur (référence du service Web). Afin de supprimer l'instance d'un adaptateur, deux solutions existent : redémarrer le serveur hébergeant le service Web d'adaptation, ou invoquer la méthode *free*, qui prend en paramètre le même identifiant que celui utilisé pour créer l'instance.

Nous présentons dans ce qui suit la méthode Interface.java d'un adaptateur (par exemple imageCompressor) :

```
import com.sun.xml.ws.developer.Stateful;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import javax.xml.ws.soap.Addressing;
import com.sun.xml.ws.developer.StatefulWebServiceManager;

@Stateful @WebService @Addressing
public class Interface {
private final String id;
String adapted_media_url="";

public static StatefulWebServiceManager<Interface> manager;
public Interface (String id)
   {
      this.id=id;
   }

 @WebMethod
  public String adapt(
        @WebParam(name = "media_url") String media_url,
        @WebParam(name = "inputMimetype") String inputMimetype,
        @WebParam(name = "transcoder_outputmimetype") String transcoder_outputmimetype,
        @WebParam(name = "outputMimetypes") String outputMimetypes,
         @WebParam(name = "price") String price,
```

```
        @WebParam(name = "additional_parameters") String additional_parameters) {

// Appeler le code qui permet de réaliser l'adaptation

return adapted_media_url;
}
   @WebMethod
   public String getState() {
      // Application de l'algorithme de contrôle d'admission, retourne l'état d'un
//adaptateur
   }
   @WebMethod
   public String getAdaptationState(@WebParam(name = "inputMediaURL") String inputMediaURL) {
      // Retourne l'état de l'adaptation d'un media donné
   }
}
```

Nous remarquons que cette classe contient en plus de la notation @WebService, les deux notations @Stateful et @Addressing. Cela signifie que cette classe supporte l'asynchronisme et la gestion de session. La déclaration *public static StatefulWebServiceManager<Interface> manager* va de paire avec la déclaration *Interface.manager.export(new Interface(id))* de la classe *Assigner.java*. Ces deux déclarations alliées avec le constructeur java *Interface* permettent d'instancier la classe d'un adaptateur à chaque fois qu'il est appelé. Pour cela, un premier appel au service Web *Assigner* est réalisé qui retourne comme résultat une référence vers l'objet correspondant au service Web *Interface*.

Interface.java doit implémenter également deux autres méthodes en plus de la méthode adapt : getState() et getAdaptationState(URL_media) qui permettent respectivement de connaître l'état d'un adaptateur lors de la phase de contrôle d'admission et de connaître l'état d'un adaptation en cours.

2.3 Algorithme de négociation et d'acceptation

Le protocole de négociation et d'acceptation facilite le choix d'un adaptateur par le gestionnaire d'adaptation. Pour cela, chaque adaptateur PAAM doit implémenter une politique d'acceptation correspondant à son mode de fonctionnement.

Nous citons dans ce qui suit les politiques les plus simples à implémenter :

2.3.1 Algorithme 1

Cet algorithme très trivial consiste pour un adaptateur donné de refuser toutes adaptations entrantes s'il est déjà en train d'adapter.

1. Appel de getState( )
2. **Si** (adaptation_en_cours = = true) **Alors** Return (Refuser_Adaptation)
3. **Sinon** Return (Accepter_Adaptation)

2.3.2 Algorithme 2

Cet algorithme est dérivé du précédent. L'adaptateur accepte d'adapter seulement et seulement si la machine (le nœud) qui l'héberge réalise aucun autre traitement.

1. Appel de getState( )
2. **Si** (nombre_traitements_machine > = 1) **Alors** Return (Refuser_Adaptation)
3. **Sinon** Return (Accepter_Adaptation)