Adaptation dynamique dans une application multimédia répartie

Oussama Layaida* — Daniel Hagimont**
**Université Joseph Fourier*
BP 53 - 38041 Grenoble Cedex 9
*** Projet Sardes*
Unité de recherche INRIA Rhone-Alpes
INRIA, 655 av. de l'Europe, 38334 Saint Ismier Cedex, France

{Oussama.Layaida, Daniel.Hagimont}@inrialpes.fr

RÉSUMÉ. *Dans cet article, nous présentons une architecture basée sur un proxy pour l'adaptation dynamique d'une application multimédia. Ce proxy effectue des adaptations en temps réel des données multimédias et permet de prendre en considération les capacités des clients et les ressources réseau existantes.*

ABSTRACT. *In this article, we present a proxy based architecture for dynamic adaptation in distributed multimedia applications. This proxy performs adaptations of multimedia data in real time and makes it possible to take into account the capacities of clients and the resources network existing.*

MOTS-CLÉS : *Multimédia, Adaptation de contenu, Qualité de service, Transcodage.*

KEYWORDS: *Multimedia, Content Adaptation, Quality of Service (Qos), Transcoding.*

1. Introduction

Les dernières années ont vu l'émergence d'une multitude d'équipements mobiles permettant de se connecter à l'Internet (téléphones, assistants personnels …). Ces équipements sont très hétérogènes, en termes de capacités mémoire, de calcul ou d'affichage. Ils peuvent être connectés à l'Internet à travers des réseaux très différents, comme GPRS, Ethernet, des modems ou des cartes infrarouges. De plus, les performances de l'Internet sont très variables.

La conséquence de cette évolution est qu'il devient très difficile de faire des hypothèses sur l'environnement dans lequel une application répartie sera exécutée. Il devient alors nécessaire d'adapter dynamiquement une application répartie en fonction des caractéristiques de l'environnement d'exécution.

Ce besoin d'adaptation aux capacités de l'environnement est particulièrement sensible dans le cas des applications multimédia réparties, qui se caractérisent par une utilisation intensive des ressources de l'environnement, en termes de mémoire, charge de calcul, capacité d'affichage ou débit du réseau.

Dans cet article, nous présentons une expérimentation qui consiste à adapter une application multimédia répartie. Nous montrons qu'il est possible de réaliser dynamiquement des adaptations de données multimédia, et que cela permet de mieux gérer les ressources utilisées par l'application lorsque celles-ci sont limitées. Cette adaptation est réalisée sur un site proxy, un nœud intermédiaire entre le client qui reçoit et le serveur qui émet les données multimédia. L'adaptation sur un site proxy permet d'effectuer l'adaptation de façon transparente par rapport aux applications s'exécutant sur les sites clients et serveurs. L'adaptation est exécutée dynamiquement dans le sens où elle traite les données multimédia à la volée. Nous évaluons les gains de performance que l'on peut obtenir grâce à ces adaptations dynamiques.

2. Travaux apparentés

Nous comparons notre expérimentation à d'autres travaux effectués dans des projets qui utilisent des techniques d'adaptation pour mieux gérer des présentations multimédia réparties.

De nombreux travaux se sont intéressés à la gestion de la Qualité de Service (Qos) de bout en bout dans des architectures spécifiquement conçues pour traiter ce problème de QoS (Hafid 98 et al.). Ces travaux s'intéressent plus particulièrement à l'utilisation des ressources du réseau et se basent souvent sur des techniques de réservation de bande passante comme dans (Bianchi et al. 2000)(Margatidis et al. 2000). Ces propositions sont difficiles à intégrer dans les architectures actuelles. En effet, il est nécessaire de caractériser les besoins des applications et les caractéristiques du trafic au moment de la réservation. Malgré l'existence de

nombreux travaux dans ce domaine, il est évident qu'il est difficile de connaître ces paramètres à l'avance. Si la définition de ces paramètres n'est pas réussie, les ressources peuvent être sous-utilisées ou indisponibles. En plus, dû au grand nombre de problèmes qui se présentent dans le contrôle d'admission, la signalisation, la coordination d'allocation et de dés-allocation de ressources, la résolution des conflits de réservation, ces schémas deviennent peu maniables et s'averrent difficile pour la majorité des applications multimédia qui n'exigent pas ce type de garanties de services. D'un autre coté, la réservation de ressources ne peut être la seule solution car elle ne permet pas de prendre en considérations les capacités et les préférences des utilisateurs.

D'autres approches utilisent des comportements adaptatifs pour offrir aux applications la configuration la mieux appropriée en fonction de la disponibilité des ressources. Une première technique consiste à adapter dynamiquement le débit de transmission de la source en fonction des conditions des récepteurs (Schulzrinne et al. 1999). Cette approche est limitée car dans une transmission multipoint avec récepteurs hétérogènes, le serveur ne peut pas déterminer un niveau de qualité qui satisfait l'ensemble des clients. Il est donc préférable d'adapter la vidéo sur un proxy pour prendre en compte les capacités de chaque client. Une deuxième possibilité est de gérer plusieurs versions de la même vidéo sur le serveur, chacune de ces versions répondant à différentes contraintes de QoS (Noble et al. 1997) (Cheung et al. 1996). L'avantage est la simplicité de l'approche, mais elle manque grandement de flexibilité, car le serveur ne peut gérer toutes les versions correspondant à tous les cas possibles. Une nouvelle contrainte (par exemple un nouveau type de terminal) nécessite une prise en compte sur tous les serveurs. Un autre inconvénient de cette approche est lié à la mauvaise utilisation des ressources réseau à cause de la transmission des données redondantes, puisque les différents flux correspondent à la même séquence vidéo. Dans notre approches, les différentes versions sont produites dynamiquement sur le proxy en fonction des besoins de chaque client, sans qu'il soit nécessaire de dupliquer le contenu des vidéos sur le serveur.

Une autre technique est l'encodage en multicouches, qui consiste à diviser la vidéo en plusieurs couches cumulatives, chacune correspond à un incrément de qualité (McCanne et al. 1996). Cette approche permet de mieux gérer le problème de l'hétérogénéité étant donné que chaque client décide localement du nombre de couches qu'il souhaite recevoir sans affecter les autres clients. Cependant, certains clients ne sont pas capables d'implémenter les techniques de codage et de transmission employées par le serveur dû aux limitations de leurs capacités de traitement (on cite le cas des PDA). De plus, ces techniques de codage sont incompatibles avec les applications multimédia existantes. Dans ce cas, notre architecture basée sur un proxy peut tirer profit de cette technique en filtrant ces couches dynamiquement.

Plusieurs travaux ont déjà adressé l'adaptation à la volée du flux vidéo sur des proxy, en effectuant des transformations sur le contenu de cette vidéo (Angin et al.

1998) (Rejai et al. 1999). Ces travaux reposent essentiellement sur des simulations et se sont intéressés à des adaptations au niveau des flots médias qui consiste soit à jeter des images, soit à réduire la qualité de la vidéo à l'encodage. Nos travaux visent à adapter la vidéo à un niveau quelconque. En particulier, ces adaptations peuvent changer le format d'encodage de la vidéo, la re-dimensionner ou la passer en noir et blanc.

3. Motivations

Les applications multimédias réparties expriment le besoin d'un transfert de données efficace avec des garanties de qualités. Des adaptations de ces applications, sous des contraintes variables de ressources, permettent de maintenir un niveau de qualité optimal. Dans notre approche, un client communique à travers un site proxy, qui joue le rôle d'une entité logique placée entre le client et le serveur auquel il accède. Dans un réseau hétérogène, ce proxy devrait être placé sur la frontière, entre les grandes et les faibles connectivités.

Les bénéfices d'effectuer ces adaptations sur un proxy consistent, d'une part à déporter l'exécution d'un traitement d'un site client vers le proxy, et de diminuer la quantité de données transférées vers le client. Dans les deux cas, le principe de cette adaptation est de modifier à la volée sur le proxy la vidéo transférée. Ces modifications peuvent avoir un impact non négligeable sur la charge de calcul sur le client qui présente cette vidéo, ou sur la consommation de la bande passante sur le réseau. Considérons deux exemples illustrant ces deux motivations.

Une présentation multimédia peut être jouée sur différents types de terminaux, pouvant aller de la station de travail au PDA (*Personal Digital Assistant*). Ces terminaux diffèrent en termes de capacité de calcul et d'affichage (taille et qualité). Restituer sur un PDA une vidéo encodée pour être présentée sur un écran de station de travail est souvent inefficace parce qu'un PDA dispose d'un écran bien plus petit. Le PDA doit re-dimensionner la vidéo alors qu'il ne possède pas la capacité de calcul permettant de re-dimensionner la vidéo à la volée. Dans ce cas, on peut effectuer une adaptation sur le proxy, qui dispose d'une puissance de calcul, pour retailler la vidéo à la volée.

Le second exemple illustre l'adaptation à la variation de bande passante du réseau. La vidéo peut être transférée sur des réseaux ayant des bandes passantes différentes, voire variables. Une adaptation à ce niveau peut être effectuée par la dégradation de la qualité des données transmises vers le client, par exemple la passer en noir et blanc ou diminuer son facteur de qualité et augmenter ainsi le taux de compression. Cette adaptation réduit significativement le volume de données à émettre.

Un autre aspect important de ces scénarios est que nos adaptations peuvent être effectuées sur les sites proxy par lesquels passe le flux vidéo, sans qu'il ne soit

nécessaire d'ouvrir les sites serveurs à ces adaptations. L'intérêt ici est de réaliser des adaptations dynamiquement sans reconsidérer les logiciels installés sur les sites clients et serveurs. Lors de la connexion d'un client à un serveur, les composants logiciels d'adaptation peuvent être installés dynamiquement sur le site proxy choisi, en fonction des contraintes de l'environnement d'exécution et des besoins d'adaptation.

4. Expérimentations

Dans cette section, nous décrivons les environnements matériel et logiciel que nous avons utilisé pour nos expérimentations, afin de valider notre démarche.

4.1. Architecture de l'application

Dans notre expérimentation, les deux machines serveur et proxy sont des PC Pentium III à 700 MHz avec 256 Mo de mémoire, interconnectés par un réseau Ethernet à 100 Mb/s. La machine cliente est un PDA IPaq munie d'un processeur à 206 MHz avec 32 Mo de mémoire et un écran de 320x240, elle est connectée par un réseau sans fils à 2 Mb/s.

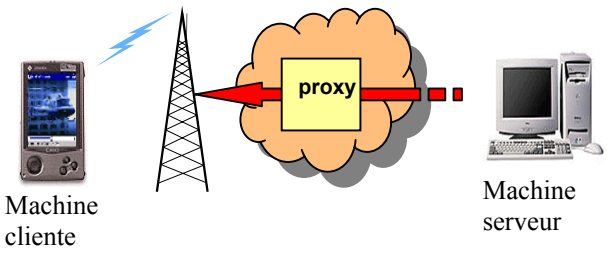

Figure 1. *Architecture de l'application*

Cet environnement matériel nous a permis de mettre au point nos scénarios de l'application multimédia répartie, et également d'évaluer les bénéfices de l'adaptation de l'application. Comme nous le verrons par la suite, nous avons pu notamment mesurer la réduction du volume de données transféré et la réduction de la charge de calcul sur la machine cliente, pour les adaptations que nous avons sélectionnées. Nous avons également pu évaluer la capacité d'adaptation de la vidéo à la volée sur le site proxy.

4.2. *Architecture logicielle*

Les logiciels que nous avons utilisés sur le site serveur et le site client sont des logiciels standard. Il s'agit respectivement d'un serveur Web Apache et des clients de visualisation multimédia sur PDA (PocketTV pour les données en MPEG et VVP pour les données en H.261).

Notre but est de réaliser des adaptations de la vidéo sur le proxy pour modifier le contenu de la vidéo en temps réel. Notre logiciel qui joue le rôle du proxy repose sur l'environnement DirectShow de Microsoft (DirectX 8.0). Cet environnement fournit une interface de programmation pour la manipulation des données multimédia par l'assemblage de composants logiciels, facilitant ainsi la configuration d'un logiciel proxy. Le traitement qu'effectue le proxy consiste à transformer dynamiquement la vidéo d'un certain format en un autre. Pour cela, la vidéo en entrée est décodée en une représentation intermédiaire, qui sera ensuite transformée et délivrée à l'encodeur qui produit un autre format en sortie. Le passage dans une représentation intermédiaire nous permet d'effectuer plusieurs modifications sur la vidéo, comme le re-dimensionnement ou le passage en noir et blanc.

5. Evaluations

Le scénario de l'expérimentation consiste à comparer deux cas de figure. Dans le premier, le client visualise une vidéo au format MPEG à partir du serveur. Aucune adaptation n'est effectuée. Nous utilisons des vidéos de tailles différentes afin d'observer le comportement du client qui doit re-dimensionner la vidéo pour la présenter sur le PDA.

Dans le deuxième cas, le client communique avec le proxy qui récupère la vidéo en MPEG à partir du serveur, réduit sa taille et modifie son format d'encodage au format H261 (format connu pour être moins coûteux en termes de volume de données et de charge de calcul pour le décodage).

Lorsque les données multimédia envoyées au PDA sont encodées au format MPEG, elles sont transférées en utilisant le protocole HTTP (Berners Lee ET al. 1999) en mode streaming, alors que lorsqu'elles sont encodées au format H261, elles sont transférées en utilisant le protocole RTP (IETF). Ces deux protocoles de transfert sont respectivement ceux utilisés pour transférer des données multimédia encodées dans ces deux formats. Toutes les mesures ont été effectuées dans les conditions réelles de l'architecture précédente.

5.1 Besoin d'adaptation pour le client

Afin d'évaluer les capacités de traitement en temps réel du client (le PDA) qui doit re-dimensionner la vidéo pour la présenter sur le PDA, nous avons mesuré la

cadence d'images affichées sur le site client lorsque celui ci reçoit une vidéo au format MPEG. Nous avons effectué ce premier test en utilisant la même vidéo avec des tailles différentes, encodées à la même cadence de 25 images/secondes. Le tableau suivant montre bien que les performances du PDA sont fortement dépendantes de la taille de la vidéo.

Format	**Cadence d'image** (images/secondes)
MPEG 640x480	1.76
MPEG 320x240	8.12
MPEG 176x144	17.7

Tableau 1. *Cadence d'images en MPEG*

On observe que le client peut difficilement afficher une vidéo de grande taille, car il doit décoder et re-dimensionner cette vidéo à la volée. Cependant lorsqu'on diminue la taille, on voit que la cadence d'affichage augmente, particulièrement pour le format QCIF (176x144) qui est probablement le format approprié pour un terminal mobile comme un PDA.

5.2 Influence de l'adaptation sur le client

Pour montrer l'impact d'une adaptation par un proxy sur les performances de la machine cliente, nous avons mesuré la cadence d'images affichées[1] sur le client lorsque le proxy re-dimensionne les deux plus grandes vidéos précédentes (640x320 et 320x240) vers le format QCIF (176X144) et change le format d'encodage en H.261 (Figure 2). Cette figure montre que le client dans ce cas est capable de visualiser la vidéo à une cadence proche de la cadence de la vidéo originale. On remarque aussi que la cadence est pratiquement la même pour les deux vidéos car le proxy maintient la cadence originale de la vidéo même si sa taille est plus grande.

[1] Il ne s'agit pas du nombre d'images reçues, car les clients temps réel peuvent ignorer les données qui arrivent en retard.

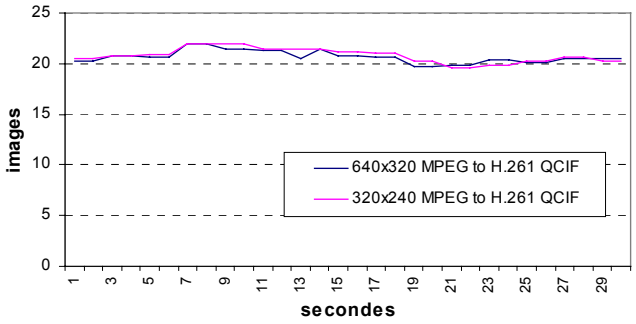

Figure 2. *Cadence d'images en H.261*

5.3 Impact sur le volume des données émis sur le réseau

Un autre bénéfice de l'adaptation sur le site proxy peut concerner le volume des données transmises sur le réseau. Dans le scénario précédent, nous avons changé le format d'encodage en H.261 et nous avons réduit la taille des images. Notons ici que l'encodage en H.261 réduit considérablement le volume de la vidéo sachant que la vidéo obtenue est de moins bonne qualité. La réduction de la taille des images a naturellement le même effet. Le tableau suivant donne les moyennes du volume des données reçues par le client pour trois des mesures précédentes. On note que la conversion au format H.261 QCIF a un effet significatif sur l'utilisation des ressources du réseau :

MPEG 640x320	2300 Kbit/s
MPEG 320x240	700 Kbit/s
H.261 QCIF	250 Kbit/s

Tableau 2. *Volume de données émises sur le réseau*

La réduction du volume des données peut aussi être obtenue par la dégradation de la qualité de la vidéo à l'encodage. Ceci peut être effectué soit en diminuant le facteur de qualité pour augmenter le taux de compression ou en diminuant la cadence d'image générée par l'encodeur, ce qui revient à jeter des images. Cette dégradation peut être bénéfique lorsque le réseau est congestionné ou bien pour alléger la charge sur le client. La figure suivante montre l'impact de cette dégradation sur la quantité de données reçues par le client. Dans ce scénario, le proxy commence par envoyer la vidéo avec un débit et une qualité élevés, ensuite il

réduit dynamiquement le débit des données produit par l'encodeur en diminuant le nombre d'images encodées.

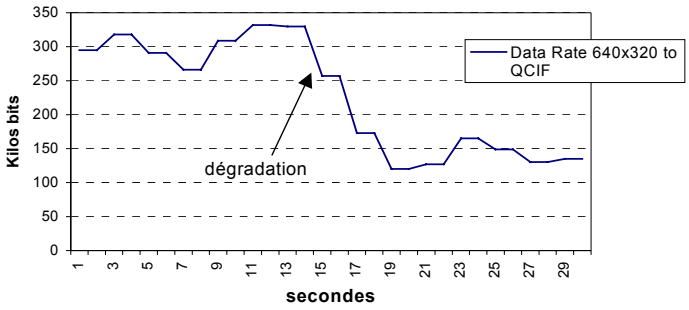

Figure 3. *Impact de la dégradation de qualité sur le volume des donnés.*

5.4 Performances de l'adaptation sur le proxy

Dans le scénario précédent, nous avons utilisé un seul effet sur le proxy pour adapter la taille de la vidéo. D'autres effets peuvent être ajoutés sur le proxy pour effectuer des traitements sur le contenu de la vidéo. Afin d'évaluer l'influence de ces effets sur les performances du proxy, nous avons ajouté trois autres codecs, le premier passe la vidéo en noir et blanc et le deuxième insert un logo sur la vidéo. Nous utilisons ici le format intermédiaire YUY2 car il permet un traitement sur la vidéo plus simple et plus rapide que sur le format RGB24. Le troisième effet change la représentation des données de l'image au format RGB24 pour la passer à l'encodeur. Nous gardons les facteurs de qualité et de compression de l'encodeur H.261 au même niveau que le premier test. La figure suivante montre la cadence d'image affichée par le client pour la première vidéo (taille originale 640x320).

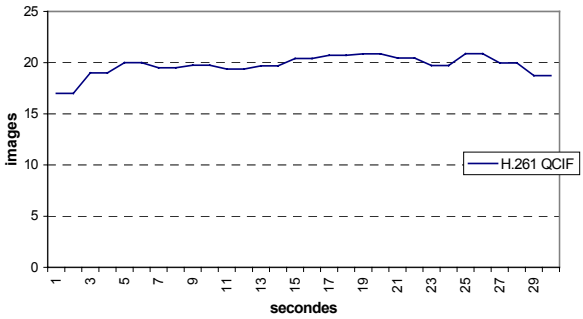

Figure 4. *Impact de l'ajout des effets*

On observe sur cette figure que l'ajout de codecs sur le canal de traitement du proxy n'influe pas négativement sur la qualité de la vidéo reçue par le client, étant donnée que le proxy maintient sa cadence d'émission au même niveau. Ces codecs permettent d'ajouter d'autres types d'adaptation, comme la personnalisation de service par exemple en insérant un message publicitaire ou un sous-titrage sur la vidéo.

Dans notre architecture, plusieurs clients peuvent ouvrir une session avec le proxy en même temps. Donc, une autre évaluation consiste à déterminer ses capacités à maintenir plusieurs sessions en parallèle. Pour cela nous avons mesuré la charge du processeur sur le proxy en fonctions du nombre des clients (figure 5). Nous rappelons ici qu'une session sur le proxy consiste à recevoir le flux du serveur, décoder, transformer et re-encoder la vidéo, ensuite la transmettre au client. Cette figure montre qu'une session consomme en moyenne 13 % des ressources processeur sur le proxy. Ce taux nous paraît acceptable dans la mesure où nous avons utilisé un PC moyen de gamme pour exécuter notre logiciel proxy. On voit aussi que lorsque le proxy crée une nouvelle session, l'utilisation du processeur est maximale ('a' sur la figure) sur une durée inférieure à 2 secondes, cette phase correspond à la création et la configuration de tous les codecs qui seront utilisés dans cette session (communication, codage, décodage et transformations).

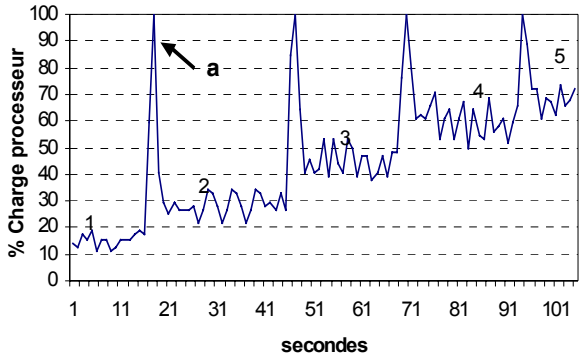

Figure 5. *Variations de la charge du processeur sur le proxy*

6.Conclusion

Dans cet article, nous avons présenté une expérience consistant à adapter une application multimédia répartie. Dans cette expérimentation, une vidéo est transmise depuis un site Web vers une machine cliente qui présente la vidéo. Le flot vidéo

passe à travers un site intermédiaire appelé *proxy*. Le proxy effectue dynamiquement des transformations sur le flot vidéo transmis à la machine cliente. Le proxy peut être configuré dynamiquement afin d'y installer les composants logiciels permettant de réaliser les adaptations requises.

Afin de valider cette approche, nous avons effectué des mesures dans le cas où l'adaptation vise l'amélioration de performance. L'adaptation sur le proxy a consisté à transformer une vidéo MPEG de grande taille en vidéo H.261 de taille QCIF (de plus petite taille). Les mesures montrent que le proxy (matérialisé par un PC) peut effectuer cette adaptation en temps réel et que cette adaptation a une influence bénéfique et significative sur le volume de données transmises sur le réseau et sur la charge de calcul nécessaire au décodage de la vidéo sur la machine cliente (matérialisée par un PDA).

Nos travaux sur l'adaptabilité dans les applications multimédia réparties se poursuivent. Nos travaux futurs concernent la configuration dynamique de la machine proxy en fonction des besoins d'adaptation ainsi que la mise en œuvre de scénarios d'adaptation plus complexes.

7. Bibliographie

E. Amir, S. McCanne, H. Zhang. « An Application Level Video Gateway ». Proc. of ACM Multimedia'95, San Francisco, November 1995.

O. Angin, A. Campbell, M. Kounavis, R. Liao. « The Mobiware toolkit: programmable support for adaptive mobile networking ». IEEE Personal Communications Magazine, 5(4), p. 32-43, Août 1998.

P. Bahl. « Supporting digital video in managed wireless network ». IEEE Communications Magazine, p. 94-102, Juin 1998.

T. Berners-Lee et al. « Hypertext Transfer Protocol -- HTTP/1.1 », RFC 2616, 1999.

G. Bianchi, A. Campbell. « A Programmable MAC Framework for Utility-Based Adaptive Quality of Service Support ». IEEE Journal on Selected Areas in Communications, 18(2), Février 2000.

Y. Chawathe, S. McCanne, E. Brewer. » RMX: Reliable multicast for heterogeneous networks ». Proceedings of IEEE InfoCom, 2000.

S. Cheung, M. Ammar, X. Li. On the use of destination set grouping to improve fairness in multicast video distribution. Proceedings IEEE InfoCom'96, March 1996.

A. Hafid et G.V Bochmann. « Distributed Multimedia Applications and Quality of Service : a review ». Electronic Journal on Network and Distributed Processing, n° 6, p. 1-50, 1998.

C.J Hughes, J. Srinivasan, and S.V. Adve. « Saving Energy with Architectural and Frequency Adaptations for Multimedia Applications ». Proceedings of the 34th International Symposium on Microarchitecture, December 2001.

IETF. RTP: A Transport Protocol for Real-Time Applications. Internet Draft, version 8, revision of RFC 1889, Juillet 2000.

ISO/IEC 11172-2:1993, The MPEG standard, Part 2: Video.

ITU-T Standardization Sector of ITU, « Video coding for low bitrate communication ». ITU-T Recommendation H.263, Mars 1996.

ITU-T Recommendation H.261: Video codec for audiovisual services at p x 64 kbit/s. Geneva, 1990, revised at Helsinki, March 1993.

M. Margaritidis, G. Polyzos. MobiWeb : Enabling adaptive continuous media applications over wireless links. IEEE International Conference on Third Generation Wireless Communications, Silicon Valley, San Francisco, Juin 2000.

Microsoft DirectX 8.0 Programmer's Reference http://www.microsoft.com/directx/

B. Noble, M. Satyanarayanan, D. Narayanan, J. Tilton, J. Flinn, K, Walker. « Agile application-aware adaptation for mobility ». Proceedings of the 16th ACM Symposium on Operating Systems Principles, Saint-Malo, France, Octobre 1997.

X.Wang et H.Schulzrinne, « Comparaison of Adaptive Internet Multimedia Applications », Juin 1999.

D. Sisalem et H. Schulzrinne, « The loss-delay adjustment algorithm: a TCP-friendly adaptation scheme», Juillet 1998.

J. Smith, R. Mohan, C. Li. « Transcoding Internet Content for Heterogeneous Client Devices ». IEEE Conference on Circuits and Systems, Monterey, June 1998.

S. McCanne et V. Jacobson, M. Vetterli, « Receiver-driven Layered Multicast », Août 1996.

R. Rejai, H. Yu, M. Handley et D.Estrin «Multimedia Proxy Caching Mechanism for Quality Adaptive Streaming Applications in the Internet» 1999.

PocketTV MPEG Movie Player for PocketPC, http://www.pockettv.com, 2000.

VIC viewer for PocketPC, http://www.oncoursetech.com/video/default.htm, Avril 2001.

X. Wang, H. Schulzerinne. « Comparison of Adaptive Internet Multimedia Applications ». Institute of Electronics, Information and Communication Engineers (IEICE) Transactions on Communications, Vol. E82-B, June 1999.

W. Yuan, K. Nahrstedt, X. Gu. « Coordinating Energy-Aware Adaptation of Multimedia Applications and Hardware Resource ». Proceedings of the 9th ACM Multimedia (Multimedia Middleware Workshop), October 2001.