Une approche « qualité de service » dans les systèmes multimédias distribués

Besma Zeddini

LITIS, UFR des Sciences et Techniques, 25 rue Philippe Lebon, BP 540, F-76058 Le Havre Cedex

Claude Duvallet

LITIS, UFR des Sciences et Techniques, 25 rue Philippe Lebon, BP 540, F-76058 Le Havre Cedex

claude.duvallet@univ-lehavre.fr

Bruno Sadeg

LITIS, UFR des Sciences et Techniques, 25 rue Philippe Lebon, BP 540, F-76058 Le Havre Cedex

bruno.Sadeg@univ-lehavre.fr

Résumé

Les applications multimédias gèrent de grandes quantités de données, dont l'exploitation doit se faire en respectant les contraintes temporelles pour permettre de "jouer" les paquets vidéo de manière fluide. Lorsque les contraintes temporelles ne sont pas respectées, la qualité de service (QdS) fournie aux utilisateurs diminue. Notre approche consiste à exploiter les travaux sur la QdS dans les SGBD temps réel afin de les appliquer aux systèmes multimédias. Dans cet article, nous présentons une architecture pour la gestion de la QdS dans les systèmes multimédias distribués, qui utilise une boucle d'ordonnancement contrôlé par rétroaction.

Abstract

Multimedia applications manage great quantities of data. These data must be managed while respecting temporal constraints in order to allow “to play” the video packages in a fluid way. If the temporal constraints are not respected, then the quality of service (QoS) provided to the users decreases. Our work in this paper consists in exploiting a QoS approach used in real-time database systems and applying it to multimedia systems. To this purpose, we present an architecture for the management of QdS in distributed multimedia systems, where the frames scheduling is controlled by a feedback loop.

Mots-clés

architecture de contrôle par rétroaction, applications multimédias, vidéo à la demande, SGBD temps réel

Keywords

feedback control scheduling, multimedia applications, on demand video, real-time database.

1. Introduction

Ces dernières années, les applications informatiques traitent des volumes de plus en plus importants de données. Beaucoup de ces applications nécessitent que les traitements soient effectués avant des dates fixes (échéances). Ces contraintes sont habituellement prises en compte par les systèmes temps réel, qui sont bien adaptés pour le respect des contraintes temporelles, mais qui ne gèrent pas de manière efficace les données en quantités importantes. La gestion de grands volumes de données est effectuée efficacement par des systèmes de gestion de bases de données (SGBD). Afin de tenir compte à la fois des contraintes temporelles des applications et de la présence de quantités importantes de données sont apparus les SGBD temps réel (SGBDTR) (Ramamritham, 1993).
Beaucoup d'applications basées sur les SGBDTR doivent très souvent faire face à des charges d'utilisation imprévisibles qui causent la surcharge du système. Durant ces périodes, les transactions temps réel chargées d'exploiter les données de l'application peuvent manquer leur échéance ou être amenées à utiliser des données obsolètes (non fraîches). Plusieurs travaux utilisant une boucle de rétroaction pour l'adaptation des paramètres de qualité de service (QdS) ont donc été menés (Kang et al., 2002) (Lu et al., 2002) (Amirijoo et al., 2006). Dans ces travaux, l'objectif est de faire varier dynamiquement les paramètres de la qualité de service afin d'arriver à une stabilisation du comportement du SGBD temps réel et d'éviter ainsi la surcharge du système ou sa sous-utilisation (gaspillage des ressources).
Parmi ces applications, on peut citer les applications multimédias dont les données sont présentes en quantités importantes et leur traitement est temporellement contraint puisque les paquets vidéos doivent être joués avant des dates fixes pour permettre une présentation fluide d'un film par exemple et maintenir ainsi une certaine qualité de service. Plus particulièrement, nous prendrons l'exemple des serveurs de vidéo à la demande où les utilisateurs, dont le nombre varie dans le temps, effectuent des demandes de visualisation de films localisés dans la base du serveur vidéo.
Dans cet article, nous allons présenter l'approche pour la gestion de la qualité de service dans les systèmes multimédias que nous avons proposée. Dans la section 2, nous présentons la problématique de nos travaux avant de poursuivre par la présentation des travaux relatifs dans la section 3. Puis, dans les sections 4 et 5, nos développons notre méthode pour la conception d'applications basée sur une approche « qualité de service ». Ensuite, dans la section 6, nous présentons les simulations que nous avons effectuées pour tester la qualité de notre approche. Enfin, nous concluons cet article par un bilan sur ce travail et les perspectives que nous comptons lui donner.

2. Problématique

Les applications multimédias se multiplient et envahissent notre quotidien. Elles nécessitent de gérer des quantités de données extrêmement volumineuses. Ces applications peuvent, à notre avis, bénéficier des avancées effectuées dans le domaine de la gestion de la qualité de service dans les SGBDTR. En effet, elles possèdent des caractéristiques similaires aux applications visées par les travaux sur les SGBDTR.

- Des données en quantités importantes : les données multimédias.
- Des charges de fonctionnement imprévisibles : à certains moments, le nombre d'utilisateurs (transactions utilisateurs) qui demandent des vidéo peut être important alors qu'à d'autres moments il est faible.
- Il est possible de réduire la qualité des vidéos fournies aux utilisateurs (qualité des transactions) si la charge devient trop importante en exploitant les particularités du format MPEG. De même, certains clients peuvent demander à disposer d'une qualité plus faible mais plus adaptée à leur moyen de visualisation des données multimédias.
- Le contrôleur d'admission peut réduire le nombre de vidéos qui sont transmises en refusant de servir certains utilisateurs afin de garantir une certaine qualité de service aux utilisateurs déjà servis.

- Les vidéos doivent être présentées en respectant des contraintes temporelles (transactions temps réel) pour obtenir une bonne qualité de vision (fluidité).

Nous cherchons dans ces travaux à appliquer les résultats obtenus sur la gestion de la qualité de service dans les SGBDTR au domaine des applications multimédias. L'objectif à terme est de permettre de concevoir des applications multimédias qui seront en mesure de fournir des garanties de qualité de service et une certaine robustesse lorsque les demandes des utilisateurs croissent rapidement. Ces travaux s'appliquent particulièrement aux applications de vidéo à la demande (VoD).
Dans la suite, nous commençons par déterminer les particularités des systèmes multimédias à prendre en considération. Dans les SGBDTR, la qualité de service est en fait traitée de façon très générale alors qu'ici elle doit être traitée de façon plus spécifique. Il est donc nécessaire de déterminer à quoi correspond la qualité des transactions, quels sont les paramètres de qualité de service lorsque l'on parle d'applications multimédias et quelles sont les contraintes temporelles du système.

3. Travaux relatifs

3.1. Les approches "qualité de service" pour les SGBDTR

Dans une application reposant sur l'utilisation de SGBDTR, des transactions en provenance des utilisateurs arrivent à des fréquences variables. Lorsque la fréquence augmente de façon considérable, l'équilibre du SGBDTR est mis en péril. Des travaux basés sur une approche qualité de service (QdS) (Kang et al., 2002) (Amirijoo et al., 2006) tentent de rendre les SGBDTR plus robustes face aux périodes d'instabilité (périodes de sous-utilisation et périodes de surcharge). Ces travaux s'appuient sur des techniques de contrôle d'ordonnancement par rétroaction[1] (Lu et al., 2002) et autorisent la manipulation de résultats imprécis (Liu et al., 1994). Dans ces approches, deux notions ont été introduites : (1) la qualité des données et (2) la qualité des transactions. Ces notions sont mises en œuvre au sein d'une architecture d'ordonnancement par rétroaction (Amirijoo et al., 2006).

3.1.1. Les données temps réel et la qualité des données

Les données temps réel représentent la capture de l'état du monde réel. Ces données doivent être remises à jour régulièrement afin de refléter au plus près le monde réel. Elles possèdent donc une durée de validité qui représente la période pendant laquelle elles peuvent être utilisées.
Amirijoo et al. ont introduit la notion de qualité des données (QdD) (Amirijoo et al., 2006) qui permet de considérer qu'une donnée stockée dans la base peut posséder un certain écart avec sa valeur dans le monde réel. On appelle cet écart "erreur sur la donnée" (noté DE[2]) et on le calcule en faisant la différence entre la donnée en base et la valeur du monde réel. Cet écart possède un seuil (MDE[3]) qui permet de déterminer si la transaction qui souhaite mettre à jour une donnée temps réel peut être écartée (DE < MDE) ou pas (DE ≥ MDE).

3.1.2. Les transactions temps réel et la qualité des transactions

Les transactions temps réel utilisées possèdent des échéances strictes non critiques. Dans ces travaux, deux types de transactions temps réel sont considérées, les transactions de mise à jour et les transactions utilisateur :

[1] Feedback Control Scheduling Architecture (FCSA).
[2] Data Error.
[3] Maximum Data Error.

- Les transactions de mise à jour ont pour tâche de mettre à jour périodiquement les données acquises depuis des capteurs.
- Les transactions « utilisateur » effectuent des opérations de lecture/écriture de données non temps réel et/ou des lectures de données temps réel.

La conception des transactions « utilisateur » est basée sur des techniques du calcul imprécis (Liu et al., 1994). Chaque transaction est composée d'une partie obligatoire et de plusieurs parties optionnelles qui sont exécutées suivant le temps disponible pour l'exécution de la transaction.

3.1.3. Le modèle global

L'architecture avec rétroaction généralement utilisée est décrite dans de nombreux articles (Kang et al., 2002) (Amirijoo et al., 2006). La tâche du *contrôleur d'admission* est de filtrer les transactions «utilisateur» qui sont acceptées ou pas dans le système. Pour cela, il considère la charge d'utilisation calculée par le *contrôleur d'utilisation et d'échéances ratées* et les paramètres de qualité de service spécifiés par le DBA(administrateur de la base de données).
Les transactions admises dans le système sont placées dans une file d'attente avant d'être envoyées au *déclencheur de transactions* pour être exécutées. Il dispose de plusieurs modules complémentaires :

- Un *gestionnaire de fraîcheur* qui vérifie la fraîcheur des données qui vont être accédées par une transaction.
- Un *contrôleur de concurrence* qui est chargé de gérer les conflits d'accès aux données qui apparaissent entre les transactions.
- Un *ordonnanceur de base*, bien souvent EDF (*Earliest Deadline First*) (Buttazzo, 1997) qui ordonnance les transactions selon leur échéance. La priorité est donnée à la transaction dont l'échéance est la plus proche.

Un *moniteur* permet de mesurer les performances du système en inspectant l'exécution des transactions (quantité de transactions terminées, abandonnées, qui ont raté leur échéance,...). Les valeurs ainsi mesurées permettent d'alimenter le *contrôleur d'utilisation et d'échéances ratées* et font partie de la boucle de contrôle par rétroaction qui va contribuer à stabiliser le système.
Le *contrôleur d'utilisation et d'échéances ratées* permet de réajuster les paramètres de QdS en fonction des valeurs déterminées par le *moniteur* et des paramètres de référence (fournis par le DBA).
Le *gestionnaire de qualité des données* permet de réajuster la valeur du paramètre MDE qui constitue le paramètre de QdD. Le paramètre MDE est ensuite fourni au *contrôleur de précision* qui écarte les transactions de mise à jour en considération de la valeur de MDE. Un calcul permet d'obtenir le gain d'utilisation obtenu en écartant des transactions de mise à jour.

3.1.4. La boucle de contrôle par rétroaction

La boucle de contrôle par rétroaction sert à stabiliser le système durant les phases de surcharge et de sous-utilisation. Pour cela, elle s'appuie sur le principe d'observation puis d'auto-adaptation. L'observation consiste à prendre en compte l'état de fonctionnement du système et à déterminer s'il correspond aux paramètres de qualité de service initialement spécifiés. À partir de l'observation effectuée, le système adapte ses paramètres, via le *contrôleur de qualité de service*, afin d'augmenter ou de diminuer les transactions acceptées dans le système. Le fonctionnement de la boucle de contrôle par rétroaction doit tendre vers une stabilité du système autour d'une valeur de référence, fixée par le DBA.

3.2. Les systèmes multimédias

La caractéristique principale des systèmes multimédias est l'utilisation de plusieurs médias tels que le son, l'image, la vidéo et le texte. Certains de ces médias (son, vidéo) doivent respecter des contraintes temporelles pour permettre une bonne qualité de visualisation et d'écoute.

3.2.1. Compression MPEG

Le standard MPEG définit un mécanisme pour coder, lors de la compression, la vidéo. Il prend en entrée une séquence vidéo et la compresse selon trois types de frames : Frames I, (Intra-frames), Frames P, (Predicted frames) et Frames B (Bidirectional frames). Chaque image d'entrée est compressée selon un des trois types de frames mentionnées ci-dessus. Les frames I peuvent être considérées comme des frames de référence ; elles sont indépendantes des autres types de frames (précédentes ou suivantes). Les frames P et B ne sont pas indépendantes ; elles indiquent les différences existant par rapport aux frames de référence. Plus précisément, la frame P spécifie les différences par rapport à la frame I précédente, alors qu'une frame B donne une interpolation entre les frames précédente et suivante de type I ou P.

3.2.2. Prise en compte des contraintes temporelles dans les systèmes multimédias

Les systèmes multimédias ont pour seul objectif de permettre à un utilisateur de percevoir une scène réelle ou virtuelle par l'intermédiaire des moyens de communication informatiques. La perception de l'utilisateur d'une scène doit être aussi naturelle que possible. Une perception naturelle par l'utilisateur sous-entend le respect d'un certain nombre de propriétés d'exécution de l'application. Ces propriétés diffèrent de celles des applications traditionnelles. Par exemple, l'utilisation d'un média comme la vidéo nécessite de respecter la fluidité, c'est-à-dire, la périodicité dans la présentation des images. L'utilisation du son impose des contraintes de fluidité encore plus fortes. Par ailleurs, l'utilisation simultanée de ces deux médias nécessite leur synchronisation: le décalage entre le mouvement des lèvres d'un personnage et le son de sa voix au-delà du dixième de seconde devient perceptible par l'utilisateur. Le terme QdS regroupe ces propriétés, qui se résument essentiellement au respect de contraintes temporelles lors de l'exécution de l'application. En effet, les contraintes de débit d'images ou de synchronisation de médias peuvent être explicitées sous la forme de contrainte de temps lors de l'exécution.

3.2.3. Gestion des données multimédias

Les applications multimédias sont très nombreuses. Nous avons limité notre champ d'étude aux applications manipulant des médias dits continus, tels que le son ou la vidéo. Ces applications comportent des contraintes communes dites contraintes de qualité de service. Par exemple, il n'est pas concevable d'autoriser des coupures répétées dans le son ou les images en raison de la surcharge du réseau. Il existe donc des contraintes temporelles telles que la nécessité de continuité de présentation de l'image et du son, mais aussi le besoin de synchronisation de ces deux flux.
Pour assurer une meilleure qualité de service aux clients, il est indispensable que le système puisse s'adapter aux changements. Il doit par exemple disposer d'un contrôleur d'admission qui filtre les transactions à servir. Il doit également disposer d'un ordonnanceur basé sur les priorités pour pouvoir satisfaire en priorité les transactions les plus urgentes. Il faut en effet tenir compte de l'urgence des paquets à transmettre pour que des coupures ne se produisent pas à la réception. Des algorithmes, tels que EDF (Earliest Deadline First) ou RM (Rate Monotonic) (Liu and Leyland, 1973), sont étendus à la gestion des flux multimédias. L'algorithme EDF ainsi que d'autres algorithmes sont également étendus pour une utilisation dans un environnement mobile (Shakkottai and Srikant, 1999).

4. Conception d'applications multimédias suivant une approche QdS

4.1. Une approche "qualité de service" pour les systèmes multimédias

Un premier aspect de notre travail a consisté à effectuer une comparaison entre les SGBDTR et les systèmes multimédias (cf. Tableau 1). Ceci nous a permis de comprendre les spécificités des systèmes multimédias afin de pouvoir y adapter les travaux effectués sur les SGBDTR.

SGBD temps réel	Systèmes multimédias
Données temps réel et non temps réel Transactions temps réel à échéance stricte non critique	Données multimédias Transactions temps réel à échéance stricte non critique
Transactions «utilisateur» décomposées (approche de Milestone) Architecture souvent centralisée	Transactions multimédias à qualité variable (utilisation du format MPEG) Architecture client/serveur

Tableau 1. - Comparaison entre les SGBDTR et les systèmes multimédias.

	SGBD temps réel	Systèmes Multimédias
Mesures de performance	Nombre de transactions qui se terminent avant échéance	Mesure de la qualité de service «humaine»
Qualité des transactions	Nombre de sous transactions optionnelles	Nombre de frames par seconde
Qualité des données	Différence entre la valeur de la donnée réelle et celle de la donnée stockée	Pas de qualité des données à gérer
Contrôleur d'admission	Limite le nombre de transactions «utilisateur» acceptées	Limite le nombre de clients effectuant des demandes

Tableau 2. La gestion de la qualité de service dans les SGBDTR et les systèmes multimédias.

Vis-à-vis de la gestion de la gestion de la QdS, nous avons, dans le Tableau 2, effectué une comparaison entre la gestion de la qualité de service dans les SGBDTR et dans les systèmes multimédias. À partir du résultat de cette comparaison, nous pouvons appliquer les méthodes de gestion de QdS utilisées dans les SGBDTR aux systèmes multimédias.

4.2. Le calcul imprécis appliqué au flux MPEG

Grâce aux dispositifs fournis par le standard MPEG, le calcul imprécis peut être effectivement introduit dans les vidéo MPEG afin de fournir des résultats approximatifs quand les ressources viennent à manquer. Comme nous l'avons déjà signalé précédemment, les frames I, P et B dans un groupe d'images compressées au format MPEG ont une certaine interdépendance. La reconstruction de la première frame P dépend de la frame I. Le décodage du reste des frames P dans le même groupe d'images dépend de la frame P précédente. En outre, le décodage des frames B dépend des frames I et P précédentes et de la frame P suivante. Avec ces dépendances, ignorer n'importe quelle frame B ne cause aucune autre perte de frames. Mais ignorer une frame P ou une frame I cause la perte de toutes les frames qui suivent et de toutes les frames B qui précèdent au sein d'un même groupe d'images.
Prenant en considération ces faits, la notion d'imprécision a pu être appliquée sur les groupes d'images au format MPEG. Ainsi, on peut considérer les frames I comme des sous-parties obligatoires et les frames P et B en tant que sous-parties optionnelles. Par conséquent, si nous

devons dégrader la qualité de la vidéo en supprimant des frames, nous commençons par supprimer les frames B, puis les frames P et en dernier lieu les frames I.

4.3. Contrôle par rétroaction appliqué aux systèmes multimédias distribués

Le contrôle par rétroaction (Lu et al., 2002) est une technique utilisée pour contrôler l'exécution des transactions ou tâces dans les systèmes temps réel et pour permettre l'adaptabilité dans les environnements où la charge ne peut pas être déterminée avec précision (Amirijoo et al., 2006). L'idée est de surveiller l'exécution du système temps réel et d'ajuster la configuration du système de telle manière que sa performance converge vers la spécification de QdS désirée. Une partie importante de l'architecture du système de contrôle avec rétroaction est constituée par les variables suivantes:

- ***yr* ,** niveau de QdS désirée (de référence) par les clients.
- ***y* ,** niveau moyen réel de QdS perçu par les clients.
- ***u*,** niveau moyen de QdS transmis par les serveurs vidéo aux clients.

Le système contrôlé contient les clients, le réseau, les serveurs vidéo, le moniteur et l'ordonnanceur. En entrée du contrôleur, on considère l'erreur d'exécution, c'est-à-dire la différence entre *yr* et *y*. Le contrôleur modifie la variable contrôlée (*y*) en ajustant la variable manipulée (u), qui est donnée en entrée du système contrôlé. Le contrôleur ajuste la variable manipulée de la façon suivante : $u = u + d$ où $d = KI * erreur$, avec $erreur = yr - y$ et $KI \in R$. *d* est utilisée pour ajuster le niveau de qualité de service fournie par les serveurs en fonction de la charge observée.

5. L'approche FCS-MS

Notre approche consiste à reprendre les travaux effectués dans le domaine de la gestion de la QdS dans les SGBDTR et de les adapter aux systèmes multimédias. Pour cela, nous proposons une méthode de contrôle par rétroaction pour les systèmes multimédias distribués, que nous avons appelé FCS-MS (*Feedback Control Scheduling for Multimedia Systems*).

5.1. Architecture de contrôle par rétroaction appliquée aux systèmes multimédias distribués

Nous nous appuyons sur les travaux de Natalia Dulgheru (Dulgheru, 2004) et de l'architecture QMPEGv2 (Ng et al., 2000) pour proposer une architecture de système multimédia distribué (cf. figure 1). L'architecture de systèmes multimédias, que nous proposons, comporte trois parties principales que nous décrivons ci-dessus.

- Le *serveur maître* : il accepte les demandes des clients, désigne les serveurs vidéo devant servir la demande, surveille l'état du système et règle les flux vidéo de sorte que la QdS soit maintenue. Le règlement du flux de données est réalisé en changeant la quantité de temps pendant laquelle le serveur vidéo peut transmettre des données.
- Les *serveurs vidéo* : ils envoient les signaux vidéo aux clients et fonctionnent sous le contrôle du serveur maître.
- Les clients : ils effectuent des requêtes auprès du serveur maître et reçoivent les frames vidéo en provenance du serveur vidéo. Chaque client possède sa propre zone mémoire tampon pour stocker les données reçues. Quand il se produit un changement d'état, il envoie un rapport de rétroaction au serveur maître (retour d'expérience sur la qualité de service reçue).

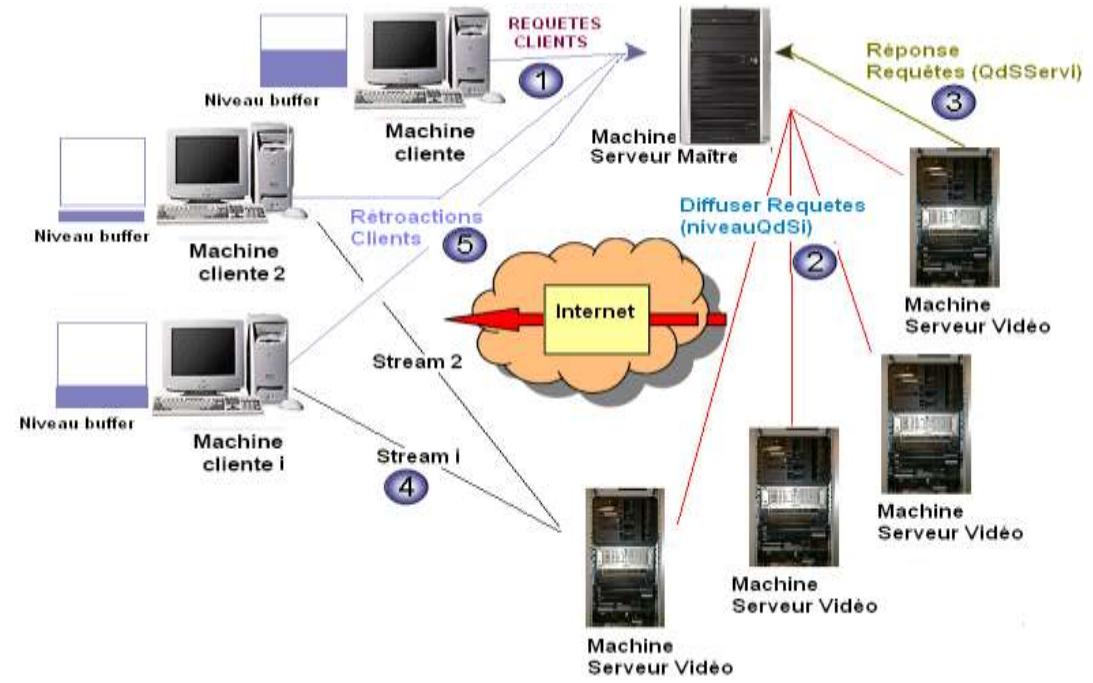

Figure 1. Scénario de fonctionnement de l'architecture FCS-MS.

5.2. Un scénario d'utilisation

Le fonctionnement typique de l'architecture que nous proposons, basé sur une boucle de rétroaction, est le suivant (cf. Figure 1) :

- Le client initie la demande d'une vidéo et envoie cette requête au serveur maître.
- Le serveur maître diffuse la requête "client" aux serveurs vidéo disponibles dans le système.
- À la réception des réponses, le serveur maître sélectionne le serveur vidéo offrant la meilleure QdS par rapport à celle exigée.
- Une connexion est établie entre le client et le serveur vidéo, qui permet la diffusion du flux vidéo.
- Le client envoie un rapport de rétroaction au serveur maître.
- Le serveur maître agit comme un ordonnanceur qui régule le flux et la qualité de service des données en fonction des conditions réseau et des retours d'expérience des clients. Si la qualité de service reçue par le client ne correspond pas à celle demandée alors le serveur maître demande au serveur vidéo de l'adapter.

La boucle de rétroaction consiste ici à adapter la qualité de service en fonction des conditions de charge du système (les serveurs et le réseau). On observe la qualité de service perçue par le client et si nécessaire on augmente ou on diminue la qualité de service côté serveur.

5.3. Architecture de contrôle par rétroaction au niveau du serveur maître

Un second niveau de contrôle par rétroaction est présent au niveau interne du serveur maître. La figure 2 illustre l'architecture interne du serveur maître : elle est composée d'un moniteur qui collecte, sur l'ensemble des clients connectés, leur niveau de QdS respectifs (*yi*), un ordonnanceur qui est sollicité par le moniteur dans le but d'orienter la redistribution du flux en fonction de la priorité du client et des mesures de performances reçues par le moniteur. Les valeurs ainsi collectées par le moniteur et traitées par l'ordonnanceur permettent d'alimenter le contrôleur de QdS et font partie de la boucle de contrôle par rétroaction qui contribue à stabiliser le système. Le contrôleur de QdS permet de réajuster les paramètres de QdS en fonction des valeurs déterminées par le moniteur, ainsi qu'en fonction de l'ordonnanceur et des paramètres de référence. Les valeurs

ainsi obtenues sont transmises aux serveurs vidéo relatifs aux clients ordonnancés selon une certaine priorité qui va être prise en considération pour la prochaine diffusion.

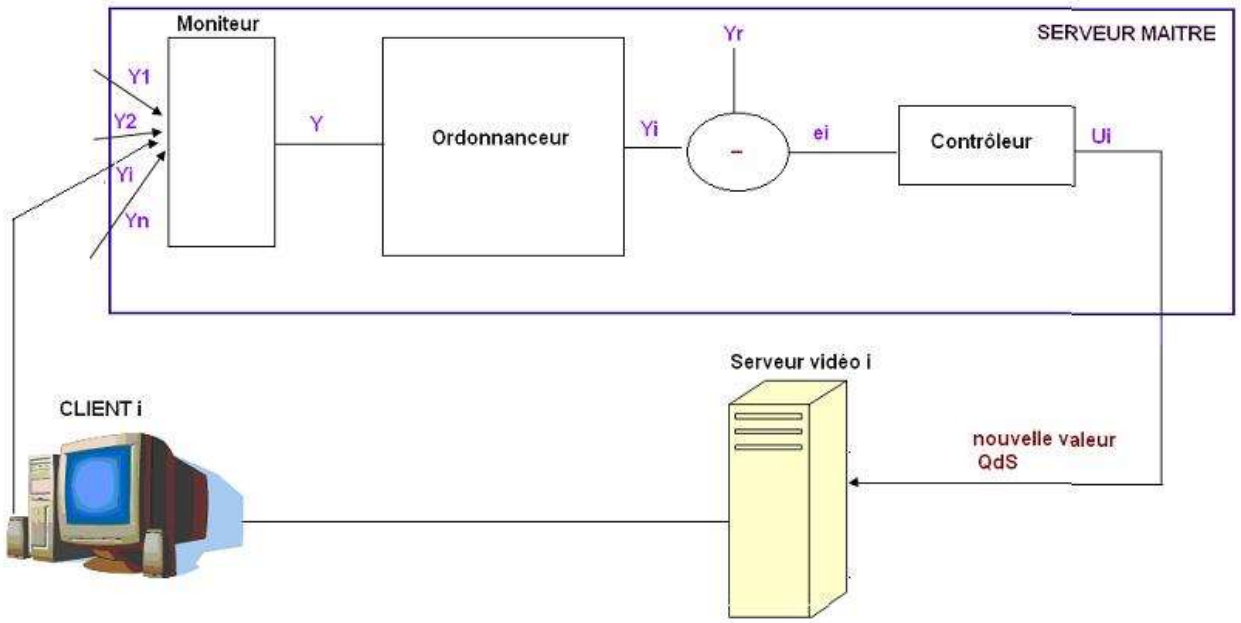

Figure 2. Architecture de contrôle par rétroaction du serveur maître.

6. Simulations et résultats

Pour tester la faisabilité de notre architecture, nous avons eu recours à la conception d'un simulateur afin de vérifier le comportement du système et son adaptation par rapport aux variations de charge.

6.1. Présentation du simulateur

Ce simulateur reprend les différents composants de l'architecture présentée dans la section 5. Un serveur maître est mis à disposition des serveurs vidéo participants à la diffusion des vidéos. Ainsi, ce service permet d'organiser l'ajout de serveurs en leur attribuant un numéro et en référençant les objets accessibles sur ceux-ci. Après le démarrage du serveur maître, les serveurs vidéo qui souhaitent participer à la distribution de vidéos se font connaître auprès de celui-ci et obtiennent ainsi un numéro.

Pour effectuer une requête, le programme client s'adresse au serveur maître, qui distribue la requête aux serveurs vidéo disponibles. Lorsque le serveur maître a désigné un serveur vidéo capable de répondre à la requête client - c'est à dire qu'il dispose du contenu demandé et qu'il est capable de fournir la QdS exigée - il envoie la référence du serveur vidéo au client. Ensuite, la diffusion en provenance du serveur vidéo et à destination du client peut commencer. Après un certain temps de diffusion, un retour sur la qualité de service reçue par le client est envoyé au serveur maître. Ce simulateur a été réalisé en JAVA et une modélisation objet a permis de concevoir les trois grandes parties de l'architecture.

6.2. Objectifs des simulations

Les simulations ont pour objectif de montrer comment notre simulateur peut adapter la QdS des clients efficacement selon la charge courante du système. Le système doit notamment s'adapter lorsque le nombre de clients qui effectuent des requêtes varie dans le temps, faisant ainsi varier la

charge du réseau. La congestion du réseau peut provenir de deux sources différentes, interne et externe comme décrit dans ce qui suit.

- Interne : un grand nombre de clients effectuent des requêtes dans le système. On peut limiter ce nombre de clients par le biais du contrôleur d'admission localisé au niveau du serveur maître.
- Externe : le réseau est utilisé par d'autres applications qui peuvent aussi provoquer la congestion. Notre architecture doit s'adapter en réduisant la qualité de service fournie aux différents clients tout en la maximisant par rapport à leurs exigences initiales.

7. Conclusion et perspectives

Dans ce travail, nous avons proposé une architecture de contrôle par rétroaction pour les systèmes multimédias distribués. Nos travaux reposent sur l'adaptation des travaux effectués pour la gestion de la qualité de service dans le domaine des SGBD temps réel. Notre contribution principale est la proposition de l'architecture FCS-MS (Feedback Control Scheduling for Multimedia Systems) et repose sur l'utilisation d'une boucle de rétroaction et du calcul imprécis appliqué aux flux MPEG. La conception d'un simulateur a déjà permis de valider la faisabilité de notre approche et devrait nous permettre à terme de fournir des résultats démontrant l'apport réel de cette nouvelle approche.
Une des extensions possible de ce travail consiste à modifier l'architecture que nous avons présentée de façon à augmenter sa fiabilité et sa robustesse. Pour cela, une des pistes envisagées est celle des architectures "peer-to-peer" qui pourraient être adaptées à nos besoins. De même, le modèle "FCS-MS" pourrait être étendu pour prendre en considération d'autres types d'applications multimédias telles que les applications interactives.

Amirijoo, M., Hansson, J., and Son, S. H. (2006). Specification and management of QoS in real-time databases supporting imprecise computations. *IEEE Transactions on Computers*. Volume 55, numéro 3, 304-319.
Buttazzo, G. (1997). *Hard Real-Time Computing Systems*. Kluwer Academic Publishers.
Dulgheru, N. (2004). *Management of QoS in Distributed MPEG Video.* Master's thesis, University of Linköping.
Kang, K., Son, S., Stankovic, J., and Abdelzaher, T. (2002). A QoS-Sensitive Approach For Timeliness and Freshness Guarantees in Real-Time Databases. *Proceedings of the Euromicro Conference on Real-Time Systems*. Vienna, Austria: June 19-21, 203-212.
Liu, C. and Leyland, J. (1973). Scheduling algorithms for multiprogramming in hard real-time environment. *Journal of the ACM*. Volume 20, numéro 1, 46–61.
Liu, J., Shih, W.-K., and K.-J. Lin, R. Bettati, J.-Y. C. (1994). Imprecise Computations. *Proceedings of the IEEE*. Volume 82, numéro 1, 83–94.
Lu, C., Stankovich, J., Tao, G., and Son, S. (2002). Feedback Control Real-Time Scheduling: Framework, Modeling and Algorithms. *Real-Time Systems*, Volume 23, numéro 1-2, 85–126.
Ng, J., Leung, K., and Wong, W. (2000). *Quality of Service for MPEG Video in Human Perspective*. Technical report, Hong Kong Baptist University.
Ramamritham, K. (1993). Real-time databases. *Journal of Distributed and Parallel Databases*. Volume 1, numéro 2, 199–226.
Shakkottai, S. and Srikant, R. (1999). Scheduling Real-Time Traffic with Deadlines over a Wireless channel. *Proceedings of ACM Wireless Mobile Multimedia*. Seattle, USA: August 20, 35-42.