

QoS coopérative pour l'adaptabilité des protocoles de Transport dans le contexte des activités de groupe

François Armando

▶ **To cite this version:**

François Armando. QoS coopérative pour l'adaptabilité des protocoles de Transport dans le contexte des activités de groupe. Informatique [cs]. INSA de Toulouse, 2010. Français. NNT : . tel-00462528

HAL Id: tel-00462528
https://theses.hal.science/tel-00462528

Submitted on 10 Mar 2010

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

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



THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Institut National des Sciences Appliquées*
Discipline : *Informatique et Télécommunications*

Présentée et soutenue par *François ARMANDO*
Le *17 Février 2010*

QoS coopérative pour l'adaptabilité des protocoles de Transport dans le contexte des activités de groupe

---

JURY

Mr Nazim Agoulmine	*Rapporteur*
Mme Pascale Vicat-Blanc Primet	*Rapporteur*
Mme Louise Travé-Massuyès	*Examinateur*
Mr Amine Berqia	*Examinateur*
Mr Christian Fraboul	*Examinateur*
Mr Jacques Turbert	*Examinateur*
Mr Christophe Chassot	*Directeur de Thèse*
Mr Khalil Drira	*Directeur de Thèse*

Ecole doctorale : Mathématiques Informatique Télécommunications de Toulouse (MITT)
Unité de recherche : *Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)*
Directeurs de Thèse : *M. Christophe Chassot, M. Khalil Drira*

À ma mère Juliette,
à Marie,
à Louise . . .

Avant-propos

Je remercie tous ceux sans qui ce travail n'aurait jamais été possible...

Table des matières

Table des matières v

Liste des figures vii

Introduction 1

1 Etat de l'art 5
- 1.1 Définitions de la Qualité de Service . . . 7
  - 1.1.1 Principes généraux . . . 7
  - 1.1.2 QoS spécifique aux applications multimédias . . . 8
  - 1.1.3 Paramètres de QoS réseau . . . 9
- 1.2 Approche basée réservation pour la garantie de la QoS 10
  - 1.2.1 Modèles précurseurs pour la garantie de QoS . . . 10
  - 1.2.2 Extensions : gestionnaires de ressources . . . 12
- 1.3 Approche basée adaptation pour l'optimisation de la QoS . . . 13
  - 1.3.1 L'auto-adaptabilité des futurs services de communication 14
  - 1.3.2 Les cadres architecturaux pour la gestion de l'adaptation 17
- 1.4 QoS au niveau de la couche Transport . . . 19
  - 1.4.1 Les protocoles de Transport classiques . . . 20
  - 1.4.2 Les nouveaux protocoles de Transport, à architecture configurable . . . 21
  - 1.4.3 Le Framework ETP : Enhanced Transport Protocol . . . 22
  - 1.4.4 Mécanismes de Transport Classiques . . . 24
  - 1.4.5 Mécanismes de Transport pour le multimedia . . . 29
  - 1.4.6 Gestion coopérative de la QoS . . . 31
- Conclusion . . . 32

2 Principes d'architecture d'un système de gestion auto-adaptative de la QoS 35
- 2.1 Besoins des acteurs du système NETQOS . . . 37
  - 2.1.1 Définition des acteurs . . . 37
  - 2.1.2 Besoins des utilisateurs et des applications . . . 38
- 2.2 Description de l'architecture . . . 39
  - 2.2.1 Présentation générale du système . . . 39
  - 2.2.2 Description des services . . . 41
  - 2.2.3 Architecture fonctionnelle du système . . . 43
- 2.3 Spécification détaillée des composants de l'entité APA 49
  - 2.3.1 Choix remarquables de conception . . . 49
  - 2.3.2 Spécification détaillée du PDM . . . 54
  - 2.3.3 Spécification détaillée du PAM . . . 55

2.3.4 Spécification détaillée du PEM . . . 56
Conclusion . . . 60

3 Gestion coopérative de la QoS au niveau Transport 63
3.1 Répartition de la bande passante . . . 67
3.1.1 Contexte applicatif et exemple : les OIU . . . 67
3.1.2 Description des Opérations d'Intervention d'Urgence (OIU) 68
3.1.3 Problématique et approche . . . 70
3.1.4 Formalisation du problème . . . 73
3.1.5 Calcul de la répartition de bande passante . . . 74
3.1.6 Problème P_1 : minimiser la bande passante globale cédée 75
3.1.7 Ajout d'un deuxième critère et définition du problème P_2 : minimiser l'excédent de bande passante libérée . . . 80
3.1.8 Conclusion sur l'utilisation des critères de minimisation globale (z_1) et de minimisation des excédents (z_2) . . . 82
3.2 Gestion coopérative de la QoS dans le système NETQoS 82
3.2.1 Identification des connexions . . . 82
3.2.2 Classe MoMe . . . 83
3.2.3 Mise en œuvre du processus de coopération . . . 84
3.2.4 Diagramme de séquence pour la coopération . . . 84
3.3 Outils et protocoles pour la récupération des informations . . . 86
3.3.1 SNMP . . . 86
3.3.2 Traceroute . . . 87
3.3.3 Protocoles de routage . . . 88
3.3.4 Fonctions de Monitoring dans ETP . . . 88
Conclusion . . . 89

4 Mise en œuvre et évaluation de performance 93
4.1 Evaluation de performance de l'APA . . . 95
4.1.1 Spécification du contexte de mesure . . . 95
4.1.2 Métriques mesurées . . . 97
4.1.3 Scénarios de mesures, résultats et analyses . . . 97
4.1.4 Conclusions . . . 102
4.2 Gestion coopérative de la QoS et répartition de débit 104
4.2.1 Contrôle de congestion utilisé : TFRC . . . 105
4.2.2 Mise en œuvre de l'approche . . . 105
4.2.3 Etude de faisabilité - Scénarios simples . . . 106
4.2.4 Etude de faisabilité - Scénarios plus complexes . . . 109
Conclusion . . . 116

Conclusion 119

Bibliographie 123

Bibliographie de l'auteur 129

Glossaire 133

Liste des figures

1.1 Architecture d'une entité de Transport 24
1.2 Espace des protocoles à ordre et de fiabilité partiels 30
1.3 Architecture d'un Transport multimédia à connexions d'ordre partiel 31

2.1 Acteurs du système NETQOS 37
2.2 Les entités principales du système NETQoS 40
2.3 Use Cases du système NETQoS 41
2.4 Use Case du service S3 : Satisfaire les politiques des acteurs 42
2.5 Use Case S3/1 : Satisfaire une politique utilisateur exécutant une application NETQoS-Unaware 44
2.6 Diagramme de structure composite de NetQoS 45
2.7 Diagramme de structure composite de l'APA 47
2.8 Diagramme de classe de l'APA 51
2.9 Autre classes du package APA 52
2.10 Diagramme de séquence d'un provisionnement au niveau Réseau/Transport (UC1) 53
2.11 Diagramme de séquence d'une adaptation au niveau Réseau/Transport (UC2) 54
2.12 Diagramme de classe du PDM 55
2.13 Diagramme de classe du PAM 56
2.14 Diagramme de classe des composants internes PEM 57
2.15 Problème des multiples entités gérées à un niveau donné . . 57
2.16 Interactions de l'ETP Agent avec l'APA.PEM et les entités ETP 58
2.17 Diagramme de classe de l'ETP agent 59
2.18 Diagramme de structure composite de l'ETP Agent 59
2.19 Stack ETP Agent pour une adaptation à distance 60

3.1 Scénario d'Opération d'Intervention d'Urgence 69
3.2 Réseau de connexions 76
3.3 Réseau de connexions 78
3.4 Classe CollaboratingConnection 83
3.5 Classes Modifiées 83
3.6 Diagramme de séquence Policy Violation 85
3.7 Diagramme de séquence pour une adaptation coopérative . 86
3.8 Fonctions de Monitoring dans ETP 89

4.1 Les entités principales du système NETQoS 95
4.2 T-APA : Histogramme du % de requêtes de provisionnement servies par intervalle de temps 98

4.3 T-APA : Temps de transit pour une requête de provisionnement . . . 99
4.4 T-APA : Temps de transit moyen d'une requête de provisionnement vs. taux d'arrivée . . . 100
4.5 APA : Evolution de la longueur des files d'attente en fonction du temps . . . 100
4.6 APA : Histogramme du % de requêtes d'adaptation servies par intervalle de temps . . . 102
4.7 PAM : Temps de transit pour une requête d'adaptation . . . 103
4.8 APA : Temps de transit moyen pour une requête d'adaptation vs. taux d'arrivée . . . 103
4.9 T-APA : Evolution de la longueur des files d'attente . . . 104
4.10 Scénario I : Flux suivant la même route . . . 107
4.11 Scénario II : Flux de sources distinctes . . . 108
4.12 Scénario III : Flux de sources distinctes, bottleneck distincts 109
4.13 Topologie réseau de la simulation ns-2 . . . 110
4.14 Débit TFRC modifié . . . 112
4.15 Débit TFRC modifié et lissé . . . 113
4.16 Débit TFRC modifié - version 2 . . . 114
4.17 Débit TFRC modifié et lissé - version 2 . . . 114
4.18 Débit TFRC + Shaper avec connexions donneuses non contraintes . . . 116

Introduction

Contexte, problématique et positionnement des travaux

La "Société de l'information", telle qu'est qualifiée notre société actuelle, repose sur la communication à grande vitesse, à haut débit, et en tout point. Ceci s'est concrétisé par la multiplication de nouvelles technologies de communication, tant au niveau des réseaux sans fil, tels que WiFi ou Bluetooth, que des équipements terminaux, incluant aujourd'hui des équipements mobiles de type PDA ou smartphone.

Ces évolutions préfigurent une mobilité généralisée des utilisateurs de l'Internet. Dans un futur proche, la communication devrait être diffusable encore plus largement par une utilisation massive de processeurs embarqués de type capteurs ou actionneurs : l'Internet sera alors extrêmement diffus, mobile et hétérogène, au moins à sa périphérie, c'est-à-dire là où se situeront les utilisateurs, qu'ils soient humains ou plus généralement objets communicants.

L'horizon qui se dessine ainsi est celui d'un Internet ambiant au travers duquel :
- les utilisateurs seront mobiles et donc susceptibles d'utiliser des ressources en différents endroits, sans avoir à considérer leur environnement courant ;
- la communication sera diffusable partout mais avec des caractéristiques différentes (en termes de débits, de taux de pertes, ...) en fonction des ressources disponibles ;
- le réseau et ses services seront à même de satisfaire, voire d'anticiper les besoins des utilisateurs, en tirant au maximum partie des capacités de l'environnement, incluant les utilisateurs eux-mêmes, dans lequel ceux-ci évolueront.

Cet environnement futur ouvre des perspectives applicatives nouvelles autorisant la coopération dynamique d'utilisateurs mobiles. Nous nous intéresserons particulièrement au cadre défini par des exercices tels que les Opérations d'Intervention d'Urgence (OIU) en situation de crise. Dans ce type de contexte :
- le réseau est en tout ou partie constitué par les utilisateurs eux-mêmes ;
- les ressources en bande passante sont potentiellement limitées et variables en fonction de la topologie du réseau, qui peut évoluer avec la progression de la mission ;
- les participants appartiennent à des groupes structurés, induisant une

hiérarchisation potentielle des collaborations susceptible d'évoluer avec l'activité ;
– enfin, par la nature des communications (audio, vidéo) ainsi que par le caractère d'urgence de la mission, des besoins apparaissent en terme de Qualité de Service (QoS) ou de sécurité sur les transferts de données.

Pour venir en support de ce type d'activités, les futurs services de communication devront être hautement dynamiques, c'est-à-dire activables à la demande des utilisateurs. Ces services devront également être aptes à prendre en compte des besoins et des contraintes multiples et évolutifs dans le temps en fonction de l'exercice ciblé, de la mobilité des utilisateurs, des flux de données échangées ou encore des caractéristiques de l'environnement de communication.

Pour aborder cette dynamicité, l'adaptabilité au contexte apparaît comme le concept majeur qu'auront à définir et à intégrer les solutions proposées. Il s'agit d'une part de définir de nouveaux modèles de représentation du contexte ciblé, et d'autre part, de traiter dynamiquement les évolutions de ce contexte. La coopération semble un cadre nécessaire pour traiter l'adaptabilité au contexte. Cette notion existe déjà au travers des solutions dites pair à pair développées par exemple au niveau applicatif pour la découverte de fichiers ou de route à délai minimal ou permettant de contourner les pare-feu ; elle existe aussi au niveau des réseaux sans fil et ad hoc, dont les nœuds sont constitués par les terminaux, susceptibles de servir de routeurs entre les utilisateurs trop éloignés les uns des autres pour communiquer directement.

Dans ce contexte, les travaux présentés dans ce mémoire s'inscrivent dans la problématique de l'adaptabilité des systèmes de communication pour la QoS. Plus précisément, le problème abordé porte sur l'adaptation dynamique et coopérative de protocoles de Transport configurables, pour répondre aux besoins en QoS d'applications multimédias distribuées dans un contexte d'activités coopératives de groupe de type opérations d'intervention d'urgence.

Contributions

De nombreux travaux ont été menés ces dernières années sur le thème des protocoles de Transport orientés QoS, d'abord à mécanismes de reprise des pertes paramétrables en termes d'ordre et de fiabilité partiels [AME 94], puis configurables plus largement par la possibilité d'y composer des mécanismes protocolaires choisis à bon escient lorsque l'établissement de la connexion [KOH 06], [EXP 03b], [EXP 03a].

Partant de la plus récente des propositions précédentes (ETP - Enhanced Transport Protocol), les contributions présentées dans ce mémoire visent d'une part à répondre au besoin en architecture de contrôle pour l'adaptation dynamique des protocoles de Transport configurables, et d'autre part à répondre au problème du choix des actions d'adaptation de ces protocoles dans un contexte d'activités de groupe coopératives. Au travers de

cette seconde contribution, nous proposons une approche de gestion coopérative des protocoles de Transport permettant de prendre en compte les priorités des communications déduites du caractère hiérarchique des activités de groupe ciblées.

La première contribution a été réalisée dans le cadre d'un projet européen (projet IST NETQoS) et porte sur une proposition d'architecture conceptuelle pour la mise en œuvre de l'adaptation des protocoles de communication, en particulier de niveau Transport. Suivant une approche inspirée d'une part par le cadre proposé pour la gestion de réseaux basée politique [STE 99], et d'autre part par celui de l'Autonomic Computing [IBM 06], les contributions exposées portent de façon plus spécifique sur l'élaboration de l'architecture du composant central du système, chargé de concevoir, d'adapter et de faire appliquer des politiques opérationnelles aux entités en charge de la gestion directe de la QoS, typiquement de niveau Transport.

La deuxième contribution porte sur le problème de la décision et propose une approche originale de gestion coopérative des protocoles de Transport pour la QoS dans les activités de groupe. Inspirée des propositions de gestion de la QoS dans les protocoles de Transport à multi-connexions [CHA 96], [OWE 98], [STE 04], l'approche explorée vise à privilégier les connexions de Transport les plus prioritaires au détriment des connexions les moins prioritaires. Nous nous intéressons plus spécifiquement au problème de la répartition de la bande passante entre ces connexions, en ciblant un objectif non plus d'équité (comme le vise TCP), mais de différenciation en fonction de la priorité des connexions. Après avoir défini le problème et les données nécessaires à sa résolution, nous le formulons sous la forme d'un problème de programmation linéaire résolu grâce à la méthode du simplexe. Deux critères d'optimisation sont définis et discutés.

Plan du mémoire

Ce document est structuré de la façon suivante :

Le chapitre 1 présente un état de l'art des propositions de gestion de la QoS requise par les applications, notamment multimédias. Ces propositions, qui relèvent des différents niveaux de l'architecture TCP/IP, sont classées en deux approches. En premier lieu, il existe des solutions consistant à réserver les ressources au niveau du réseau, afin de garantir une QoS aux applications. En second lieu, dans les environnements où la réservation de ressources n'est pas envisageable, les solutions consistent à adapter les protocoles de communication afin d'optimiser la QoS. Nos contributions s'inscrivent dans cette deuxième approche. Plus spécifiquement, elles visent une optimisation de la QoS par le biais des protocoles et des mécanismes de niveau Transport. Nous présentons ainsi dans ce premier chapitre un large éventail de solutions de niveau Transport orientées QoS, en particulier le concept de protocoles configurables.

Le chapitre 2 présente notre première contribution à l'adaptabilité des pro-

tocoles de communication, notamment de niveau Transport. Il ne s'agit pas d'une contribution consistant à modifier, à améliorer ou à proposer un nouveau protocole de Transport. Nous proposons dans ce chapitre les principes d'architecture d'un système de gestion de la QoS permettant de piloter un protocole de Transport configurable. L'architecture de ce système, dont la conception suit une approche top/down, a été développée dans le cadre du projet Européen NETQoS. Elle vise à définir les composants et leurs interactions nécessaires à une adaptation dynamique de solutions de QoS non pas seulement de niveau Transport mais relevant aussi d'autres niveaux, notamment le niveau Réseau. Dans ce chapitre, nous présentons la proposition d'architecture générale pour le système NETQoS, puis nous détaillons nos contributions spécifiques au projet au travers du composant de gestion de l'adaptation (APA - Automated Policy Adaptor) et de son interaction avec le protocole de Transport ETP.

Le chapitre 3 présente notre deuxième contribution, relevant des actions d'adaptation pour optimiser la QoS au niveau Transport dans un contexte d'activité de groupe de type OIU, induisant des priorités différentes entre les communications. L'idée originale consiste à envisager une gestion dite "coopérative" de la QoS, qui permet, en cas de pénurie de ressources, de privilégier les connexions de Transport les plus prioritaires au détriment d'autres moins prioritaires. Cette gestion coopérative nécessite de pouvoir récupérer des données de supervision issues du réseau et des connexions, puis de prendre une décision de partage des ressources, avant de déployer cette décision et de la mettre en œuvre sur les hôtes d'extrémités par l'intermédiaire d'un protocole de Transport configurable. Le principal effort de notre contribution porte sur la décision. Nous abordons le problème de la répartition de la bande passante entre les connexions en le formulant sous la forme d'un problème d'optimisation sous contraintes. Les extensions à apporter à l'architecture NETQoS pour permettre cette gestion coopérative sont également proposées. Des pistes sont finalement données concernant la façon d'envisager la récupération des données de supervision.

Le chapitre 4 présente une évaluation des contributions apportées dans les chapitres 2 et 3. Des mesures de performance réalisées en simulation permettent de mettre en évidence le bien fondé et les limites des solutions proposées. Nous présentons d'abord une évaluation du composant central (APA) du système NETQoS dans l'optique d'en éprouver la capacité à résister au facteur d'échelle vis-à-vis du nombre de requêtes à traiter. Dans un deuxième temps, nous montrons la faisabilité du principe de gestion coopérative de la QoS présentée au chapitre 2. Des implémentations d'un mécanisme de contrôle de congestion permettant de mettre en œuvre la décision prise par le modèle d'optimisation lié à la gestion coopérative sont testées grâce au simulateur ns-2.

Enfin, nous terminons le manuscrit par une conclusion générale. Celle-ci résume les points abordés dans ce mémoire ainsi que les contributions apportées. Elle indroduit également un ensemble de perspectives pour structurer les recherches futures autour de la problématique abordée.

Etat de l'art

1

Sommaire

1.1 Définitions de la Qualité de Service 7
1.1.1 Principes généraux 7
1.1.2 QoS spécifique aux applications multimédias 8
1.1.3 Paramètres de QoS réseau 9
1.2 Approche basée réservation pour la garantie de la QoS 10
1.2.1 Modèles précurseurs pour la garantie de QoS 10
1.2.2 Extensions : gestionnaires de ressources 12
1.3 Approche basée adaptation pour l'optimisation de la QoS 13
1.3.1 L'auto-adaptabilité des futurs services de communication 14
1.3.2 Les cadres architecturaux pour la gestion de l'adaptation . 17
1.4 QoS au niveau de la couche Transport 19
1.4.1 Les protocoles de Transport classiques 20
1.4.2 Les nouveaux protocoles de Transport, à architecture configurable 21
1.4.3 Le Framework ETP : Enhanced Transport Protocol 22
1.4.4 Mécanismes de Transport Classiques 24
1.4.5 Mécanismes de Transport pour le multimedia 29
1.4.6 Gestion coopérative de la QoS 31
Conclusion 32

Ce chapitre présente un état de l'art des solutions envisagées ces dernières années pour répondre aux besoins en Qualité de Service (QoS) des applications, notamment multimédias, distribuées dans l'Internet.

Les définitions, les principes généraux et les standards attachés au concept de QoS sont tout d'abord introduits en section 1.1. Les besoins en QoS des applications multimédias sont ensuite détaillés, ainsi que les paramètres de QoS réseaux correspondants.

Les sections 1.2 et 1.3 présentent deux approches explorées pour la gestion de la QoS de bout-en-bout. La première approche, basée sur le principe de la réservation de ressources, vise à garantir un niveau de QoS aux applications. La deuxième approche, que nous avons suivie dans nos travaux, est orientée adaptation aux variations des ressources (et plus généralement aux variations du contexte, qu'il soit applicatif ou réseau) ; elle a pour

objectif de maintenir une QoS au plus haut niveau, sans chercher à garantir celui-ci. Nous présentons pour chacune d'elles les solutions existantes ou proposées dans la littérature, en détaillant les problématiques liées à l'approche adaptation.

La dernière partie du chapitre (section 1.4) détaille les solutions de gestion de la QoS au niveau Transport de l'architecture de l'Internet. Les contributions de cette thèse se situent en effet à ce niveau de l'architecture, et visent à développer les protocoles, les architectures de contrôle et les modèles permettant d'optimiser la QoS offerte aux applications par une adaptation des protocoles de Transport. Nous décrivons ainsi les protocoles de Transport classiques et standardisés (UDP, TCP, SCTP, DCCP), puis les nouveaux protocoles de Transport qui permettent de prendre en compte les problématiques de QoS nouvelles, telles que celles issues des applications multimédias et de leur utilisation dans des contextes d'activités coopératives. Nous décrivons en particulier le protocole de Transport ETP (Enhanced Transport Protocol) sur lequel s'appuient les travaux présentés dans ce mémoire. Enfin, cette section décrit les mécanismes de niveau Transport ayant un impact direct sur la QoS, c'est-à-dire le contrôle de congestion et le contrôle d'erreur, et introduit les approches de gestion coopératives qui ont inspiré nos travaux.

1.1 Définitions de la Qualité de Service

1.1.1 Principes généraux

Le terme Qualité de Service (QoS) est abondamment utilisé dans les réseaux informatiques et représente une large collection de technologies et de techniques qui visent à offrir aux applications distribuées et à leurs utilisateurs des priorités non fonctionnelles (dites de QoS, telles qu'un débit garanti ou un délai borné par exemple) sur le transfert de leurs données de bout-en-bout.

On peut classifier ces technologies et techniques en deux familles :
- celle des solutions visant à offrir une QoS statistique (Soft QoS) : les paquets, identifiés par leur classe de trafic, sont priorisés par rapport aux autres paquets dans chaque nœud du réseau. Suivant cette approche, aucune garantie ne peut être faite de bout-en-bout. Des exemples de solutions de ce type sont notamment DiffServ dans l'Internet ou 802.1p dans les réseaux locaux.
- la deuxième famille de solutions vise à offrir une QoS garantie (Hard QoS) : les ressources de communications sont réservées de bout-en-bout pour chaque flux de trafic identifié pour garantir la performance requise. Des exemples de solutions de ce type sont notamment IntServ et RSVP.

Trois notions de QoS ont été définies [HAR 01] :
- la QoS intrinsèque : elle correspondant aux fonctionnalités du réseau mises en place suivant une approche Soft ou Hard, au provisionnement du réseau en bande passante, etc. . . ;
- la QoS perçue : elle reflète l'expérience de l'utilisateur pour un service particulier ; elle est influencée par les attentes de l'utilisateur en comparaison au service mesuré ;
- la QoS évaluée : elle résulte d'une observation (monitoring) selon des métriques définies.

Enfin, la QoS peut se définir de différentes manières, comme étant par exemple :

"La capacité d'offrir différentes priorités à différentes applications, utilisateurs, ou flux de donnée ou de garantir un certain niveau de performance à un flux de données".

Selon ISO 8402 : 1994 [ISO94] : "la totalité des caractéristiques d'une entité qui portent sur sa capacité à satisfaire des besoins formulés et supposés".

Selon ISO 9001 : 2000 [ISO00] : "le degré auquel un ensemble de caractéristiques inhérentes respectent des exigences".

Selon [ITU 93] (et [ETS 94]) : "l'ensemble des caractéristiques d'un service de télécommunication qui lui permettent de satisfaire aux besoins explicites et aux besoins implicites de l'utilisateur du service".

Selon l'IETF [CRA 98] : l'IETF se focalise plutôt sur la notion intrinsèque de la QoS et ne traite pas les aspects de perception. Sa définition est la

suivante : "Un ensemble de service pré-requis à remplir par le réseau lors du transport d'un flux".

Dans ce contexte très général, nous nous intéressons maintenant à une famille particulière d'applications, les applications multimédias, et nous exhibons les besoins en QoS qui en résultent.

1.1.2 QoS spécifique aux applications multimédias

Ces dernières années, les évolutions conjointes de l'informatique et des télécommunications ont conduit à l'émergence de nouveaux types d'applications distribuées, multimédias, telles que les applications de visioconférence ou de vidéo à la demande, ou encore les applications de simulation interactive distribuée.

Comparativement aux applications classiques du type transfert de fichiers, consultation de pages Web ou encore messagerie électronique, ces applications présentent des contraintes nouvelles sur le transfert de leurs données [WU 00] :

- débit garanti : pour obtenir une qualité de présentation acceptable, les applications multimédias ont besoin d'un réseau offrant un débit minimal relativement constant ;
- délai de transit borné : contrairement aux transferts de données classiques (texte, images fixes,...) qui ne sont pas soumis à des contraintes de délai strictes, les applications multimédias requièrent un délai de bout-en-bout borné lié aux médias eux-mêmes (audio ou vidéo) et/ou au degré d'interactivité souhaité entre leurs utilisateurs. Chaque paquet doit arriver à destination en un temps inférieur au seuil de perception humain. Si ce n'est pas le cas, le paquet peut être considéré comme inutilisable par l'application réceptrice ;
- fiabilité partielle : la perte de paquets lors du transfert peut conduire à une dégradation de la qualité de présentation des données multimédias. Il faut donc maîtriser ces pertes. Mais contrairement aux applications classiques nécessitant une fiabilité totale dans le transfert de données, les applications multimédias peuvent accepter de perdre certains paquets afin de répondre par exemple aux exigences temporelles, tout en offrant une qualité acceptable : on parle alors de contrainte de « fiabilité partielle » ;
- ordre partiel : de façon analogue, les contraintes d'ordre, notamment entre les différents flux d'une même application multimédia, ne sont plus les mêmes que dans un transfert de données textuelles. En effet, ces applications impliquent généralement une manipulation coordonnée et un transfert de plusieurs types de média (texte, graphisme, audio, vidéo,...). Il en résulte un besoin en synchronisation des différents média du coté de l'application réceptrice. Un exemple de synchronisation de média audio et vidéo est celui du mouvement des lèvres et de la voix (visioconférence). On parle alors de contrainte « d'ordre partiel ».

1.1.3 Paramètres de QoS réseau

La QoS perçue par l'utilisateur d'une application multimédia est influencée par deux facteurs : la qualité offerte par l'application elle-même et la qualité que l'application reçoit du réseau. Si une application utilise un codec audio ou vidéo très bas débit, alors la qualité sera toujours très faible, même si le réseau ne dégrade pas le flux de données par des délais excessifs ou des pertes. L'ITU définit dans les recommandations [ITU01], [REC 02], les paramètres de QoS réseaux suivants :

- IPTD (IP Packet Transfer Delay) : ce paramètre définit le délai de transfert qu'un paquet peut subir durant son transfert dans le réseau. Cette métrique est particulièrement importante pour les applications ayant des contraintes de temps réel. L'exemple classique est un flux de voix sur IP (VoIP) qui doit être délivré avec un délai faible pour permettre un niveau de qualité acceptable. L'IPTD est généralement exprimé par le délai minimum, le délai maximum ou le délai moyen ;
- IPDV (IP packet delay variation) : ce paramètre est défini comme la différence entre la borne supérieure de l'IPTD à 10^{-3} près et l'IPTD minimum. Il est important de garder la valeur de l'IPDV à un niveau bas ou très bas pour les applications temps réel ;
- IPLR (IP Loss Ratio) : ce paramètre identifie le nombre de paquets IP perdus sur le nombre total de paquets IP transmis. Il est important de garder la valeur de l'IPLR à un niveau bas pour toutes les applications temps réel qui ne peuvent pas retransmettre les paquets perdus. Pour les applications non temps réel les paquets perdus doivent être retransmis. La tolérance des applications multimédias à certaines pertes permet cependant de compenser les imperfections du réseau ;
- IPER (IP Packet Error Rate) : ce paramètre identifie le rapport entre le nombre de paquets erronés sur le nombre de paquets transmis ;
- IPPT (IP Packet Throughput) : ce paramètre identifie le débit requis. Il est égal au nombre total de paquets IP reçus sans erreurs sur un intervalle de temps donné. Du point de vue de l'application, le débit requis peut être un paramètre important. Les besoins en débit sont directement liés au codec employé.

Face aux besoins en QoS des applications multimédias, deux approches majeures ont été explorées ces dernières années :

- la première approche est basée sur le provisionnement d'une QoS garantie par réservation de ressources dans le réseau, en tenant compte des besoins des applications exprimés en termes de paramètres de QoS réseau. Cette approche, qui a notamment contribué au développement d'autres modèles de services au niveau IP que le service de Best Effort actuel, est généralement accompagnée de mécanismes de contrôle d'admission et de mécanismes de signalisation visant à mettre en adéquation les besoins applicatifs et la disponibilité des ressources existantes.
- la deuxième approche, utilise le degré de tolérance des applications qui ne nécessitent pas de garanties de QoS dures, et qui peuvent s'adapter à certaines variations des ressources du réseau. Ces applications réalisent des adaptations en cours d'exécution qui garantissent un service aux utilisateurs meilleur que celui offert sans adaptation. Par exemple, en

réponse à une dégradation de la QoS, une application adaptative peut remplacer un codec par un autre pour réduire la QoS requise en termes de débit.

Nous détaillons ces deux approches dans les sections 1.2 et 1.3 suivantes. Rappelons que nos travaux s'inscrivent dans l'approche orientée adaptation, appliquée dans notre cas, non pas directement au niveau applicatif, mais au niveau Transport, dans l'optique d'abstraire les développeurs d'applications de la complexité de gestion de cette adaptation.

1.2 Approche basée réservation pour la garantie de la QoS

Face aux limites du modèle de service de l'Internet (qualifié de Best Effort, c'est-à-dire n'offrant aucune garantie de performance), deux groupes de travail de l'IETF, IntServ puis DiffServ, ont été successivement constitués au milieu des années 90 pour étudier la mise en œuvre de nouveaux services au niveau IP dans l'Internet. Si les modèles d'architectures résultants ont posé des bases importantes en matière de gestion de la QoS dans l'Internet, ces modèles présentent cependant des limites importantes qui ont empêché leur mise en œuvre effective (cas du modèle IntServ notamment) ou suscité des travaux complémentaires (dans le cas DiffServ notamment).

Dans cette section, nous résumons les principes des modèles IntServ et DiffSev et nous introduisons les propositions d'extension envisagées pour aboutir à une garantie de QoS de bout-en-bout invocables à la demande.

1.2.1 Modèles précurseurs pour la garantie de QoS

1.2.1.1 Intserv

Le modèle IntServ introduit, deux nouveaux services IP en plus du Best Effort, GS (Guaranted Service) [SHE 97] et CL (Controlled Load) [WRO 97a]. GS veut satisfaire les besoins des applications à fortes contraintes temporelles ; CL vise à répondre aux applications moins exigeantes, et notamment les applications adaptatives. IntServ repose sur trois principes fondamentaux :

– une réservation de ressources est effectuée par session au niveau de chacun des hôtes et routeurs du chemin de donnée ; elle se traduit par l'établissement et le maintien d'un état ; une session désigne au moins un socket de destination (unicast ou multicast) ; la granularité la plus fine est le flux ;
– cette réservation est effectuée via le protocole RSVP [BRA 97a] [BRA 97b], dans le cadre d'une phase d'établissement de réservation ; celle-ci est initiée par le(les) émetteur(s) mais ce sont le(les) récepteurs qui effectuent la requête de réservation. La requête est propagée de routeur en routeur, qui effectuent chacun tour à tour un contrôle d'ad-

mission local, et enregistrent en cas de succès un état de réservation pour la session considérée ;
– le maintien des réservations est assuré par la réception de messages de rafraîchissement, émis par le(les) émetteurs. Une réservation peut également être relâchée explicitement.

L'architecture IntServ [WRO 97b], [WRO 97a], [SHE 97], standardisée par l'IETF, est composée des éléments suivants :
– un ordonnanceur de paquet : il ordonnance les paquets selon une politique potentiellement différente du FIFO en utilisant un ensemble de file d'attente et de timers ;
– un classificateur : il classe chaque paquet entrant dans les classes de QoS ;
– un contrôleur d'admission : il implémente les algorithmes de contrôle d'admission pour déterminer si un nouveau flux peut être admis ou pas ;
– un protocole de réservation, en l'occurrence RSVP [BRA 97b] (Resource Reservation Protocol), qui véhicule les informations nécessaire à la création et au maintien d'un état pour chaque flux sur les routeurs impliqués dans le chemin de données.

Le modèle IntServ présente des limites importantes qui en interdisent le déploiement à l'échelle du plein Internet :
– d'une part, le principe de réservation au niveau de tous les routeurs, qui nécessite un état par flux et la signalisation nécessaire à son maintien ; se pose alors un problème de mise à l'échelle de l'approche vis-à-vis du nombre de flux à QoS ;
– d'autre part, l'ignorance de la structure multi-domaines de l'Internet ; cette structure de fait reflète la nécessité d'administrer les domaines de façon indépendante vis-à-vis de leurs politiques internes de gestion du routage, de la QoS ou de la sécurité.

1.2.1.2 DiffServ

Le modèle DiffServ [NIC 98], [LI 98], [BLA 98] a été conçu pour répondre aux limites d'IntServ. Il est basé sur les principes fondamentaux suivants :
– DiffServ écarte la notion de flux en cœur de réseau, et introduit à la place celle de classe : les paquets d'une même classe sont traités identiquement par tous les routeurs du chemin empruntés entre l'entrée et la sortie du domaine considéré ;
– à la différence d'IntServ, DiffServ ne propose pas de contrôle d'admission , mais s'appuie sur la notion de contrat (SLA) négocié entre utilisateur et fournisseur, préalablement au transfert des données. La partie technique d'un SLA, le SLS , spécifie (à minima) la quantité maximale de trafic de chaque classe que l'utilisateur a le droit d'injecter dans le domaine, ainsi que le traitement (conditionnement) que subiront les paquets entrants ;
– pour une classe de service, un SLA garantit un comportement (PHB - Per Hop Behavior) identique sur chaque routeur, à l'échelle du do-

maine ; deux PHB ont été standardisés, EF (Expedited Forwarding) [JAC 99] et AF (Assured Forwarding) [HEI 99]. EF vise un service équivalent à celui d'une ligne dédiée, AF vise un service à débit minimal garanti, avec différents niveaux de priorité à la perte et au délai ;
– en entrée du réseau, le conditionnement des paquets se traduit d'abord par leur classification, sur la base soit d'une marque unique (champs TOS des paquets IPv4 par exemple), soit de plusieurs champs d'entête pouvant porter sur les données utilisateur (typiquement sur l'entête des PDU de niveau Transport : n° de port source et/ou destination). S'en suit un contrôle de la conformité du trafic au regard du SLS, et l'application de mesures de mise au gabarit (policing) pour le trafic non conforme ; ce policing peut se traduire par le rejet, le retard ou le basculement dans une autre classe. Le trafic conforme est marqué suivant sa classe d'appartenance, puis injecté dans le réseau. L'ordonnancement (de type Priority Queuing, Weighted Fair Queuing, etc.) des paquets dépend de leur classe.

La proposition DiffServ est actuellement considérée comme la plus adaptée pour la gestion de la QoS dans l'Internet. Cependant, elle présente plusieurs problèmes (paramétrage des PHB, établissement des accords entre administrateurs des différents domaines, ...). Le passage au multi-domaine accroit les difficultés de gestion de la QoS pour plusieurs raisons :
– le provisionnement est plus complexe, car il inclut aussi le dimensionnement des ressources de chaque domaine par rapport aux contrats SLA/SLS avec les domaines adjacents.
– le contrôle d'admission doit être prolongé aux liens inter-domaines pour assurer la disponibilité des ressources de bout-en-bout.
– les domaines de l'Internet étant hétérogènes, les domaines peuvent fournir des services différents. En conséquence, une signalisation multi-domaine doit être donc mise en place pour gérer la continuité de la qualité de service pour une communication traversant plusieurs domaines.

De nombreux travaux ont traité du provisionnement et de la disponibilité des ressources, soit de façon isolée, soit dans le cadre d'architectures, souvent issues de grands projets. Nous les introduisons ci-après.

1.2.2 Extensions : gestionnaires de ressources

Le concept de gestionnaires de ressources a été défini pour aider à la mise en œuvre des fonctions nécessaire à la réservation, au contrôle et à la supervision des ressources dans l'Internet. Les gestionnaires de ressources ont été proposés pour différents types de ressources. Dans le contexte des architectures DiffServ, la notion de Bandwidth Broker (BB) [NIC 99] est proposée pour venir en support du service EF. Un BB est un gestionnaire des ressources en bande passante d'un domaine de l'Internet.

L'implémentation d'un BB pour la réservation dynamique de bout-en-bout dans un environnement DiffServ multi-domaines est discutée dans [KHA 01]. Un bandwidth broker centralisé dans chaque domaine ré-

seau réalise le contrôle d'admission, le provisionnement de ressources et d'autres décisions de politiques. Une architecture à base de BB est destinée à fournir des services garantis de bout-en-bout en utilisant des garanties de délai par flux ou des garanties de délai par classe. Dans [TER 99], un système de BB two-tier est conçu et implémenté pour fournir un provisionnement de QoS par classe pour une architecture DiffServ. Des systèmes de BB hiérarchiques ou distribués permettant un passage à l'échelle pour la gestion de la QoS et des SLA ont été proposés en découplant le plan de données du plan de contrôle de la QoS [ZHA 01]. En utilisant une architecture à base de BB, les routeurs de cœur ne maintiennent pas d'états de réservation de QoS, par flow ou par agrégats. Les états de réservation de QoS sont stockés et gérés seulement par le BB dans un domaine réseau.

Les architectures à base de BB présentent les avantages suivants :
- en maintenant les états de réservation de QoS seulement sur les BB, les routeurs de cœur ne gèrent pas les fonctions de contrôle de la QoS, comme le contrôle d'admission, ce qui les rend potentiellement plus efficaces ;
- le plan de contrôle de la QoS est découplé du plan de données, permettant au fournisseur de service réseau d'introduire de nouveaux services sans nécessairement requérir la mise à jour matérielle ou logicielle des routeurs de cœur ;
- le provisionnement de QoS et le contrôle d'admission pour une utilisation optimisée du réseau est possible. Le contrôle d'admission est réalisé au niveau du chemin entier, et pas seulement saut-par-saut ;
- les problèmes de fiabilité, robustesse et passage à l'échelle du plan de contrôle de QoS sont traités séparément et sans ajouter de complexité au plan de données ;
- la QoS utilise différents fournisseurs de service Internet et le routage inter-domaine est possible.

Partant de l'ensemble de ces concepts, le projet EuQoS [CAL 08], [RAC 08] a proposé et développé une architecture de gestion de la QoS pour l'Internet multi-domaines hétérogènes ; l'effort d'EuQoS a porté sur le contrôle d'admission et sur la signalisation en envisageant le provisionnement sur la base d'accords de peering.

1.3 Approche basée adaptation pour l'optimisation de la QoS

Complémentaire à la vision précédente, l'approche consistant à maintenir au plus haut niveau la QoS requise sans chercher à en garantir pour autant un niveau donné peut s'avérer incontournable lorsque les ressources en bande passante ne sont pas pérennes dans le temps. C'est par exemple le cas quand les utilisateurs finaux disposent de moyens de transmission sans fil et se déplacent durant les communications. C'est d'autant plus nécessaire en réponse aux besoins dynamiques d'applications complexes, mobiles et coopératives, distribuées dans des environnements réseaux hé-

térogènes, en partie sans fil, dont les ressources sont variables et évolutives.

Le problème alors soulevé est celui du provisionnement dynamique de services de communication auto-adaptatifs, c'est-à-dire :
- *invocables à la demande*, en tenant compte des besoins applicatifs et des utilisateurs, et des contraintes de l'environnement de communication,
- et capables de *s'adapter de façon transparente* pour les utilisateurs : d'une part, à l'évolution des besoins de ces utilisateurs, et d'autre part, à l'évolution des contraintes de l'environnement de communication, liée par exemple à la mobilité des utilisateurs.

L'objectif de cette section est de détailler cette problématique et d'introduire les éléments de solutions existants au démarrage de nos travaux.

1.3.1 L'auto-adaptabilité des futurs services de communication

[FAR 06] explicite quatre phases dans le provisionnement dynamique de services : la spécification, la découverte, le déploiement et la gestion des services.

L'*auto-adaptabilité* est un enjeu majeur du provisionnement de ces nouveaux services, qui concerne plus spécifiquement les phases de déploiement et de gestion. Les travaux récents menés dans ce contexte permettent de mettre en lumière les différentes facettes de la problématique. Aucune de ces propositions ne fournit encore de solution globale, chacune n'abordant qu'une partie du problème.

Les solutions proposées diffèrent en plusieurs points, relatifs aux objectifs ciblés, aux niveaux (au sens couches du modèle OSI) concernés, aux actions d'adaptation elles-même, aux caractéristiques de l'adaptation, et enfin à la façon de gérer l'adaptation et son autonomie.

Objectifs. Les objectifs ciblés par l'adaptation sont multiples, et peuvent par exemple consister :
- à gérer les aspects collaboratifs, comme la gestion de l'entrée/sortie dynamique d'utilisateurs dans un exercice de jeux distribués, ou de collaboration à la rédaction d'un document commun,
- à maintenir la connectivité et de la QoS au niveau du réseau d'accès lors du passage d'un réseau sans fil à un autre [BAL 04] [KAL 06],
- à optimiser la QoS de bout-en-bout dans l'Internet Best Effort : [WU 01] [YU 03] [EXP 03b] [AKA 04],
- à répondre aux problèmes de déploiement liés à la présence de firewalls, ou à gérer la confidentialité des informations échangées, etc. [SKA 04],
- à optimiser les ressources des équipements impliqués, en termes d'énergie, de capacité de calcul/stockage [MAR 01].

Niveaux. Les niveaux considérés relèvent aussi bien des couches hautes (Application, Middleware, Transport) que des couches basses (Réseau, MAC et Physique) du modèle initialement posé par [ZIM 80]. Cependant, le principe d'indépendance des couches est remis en cause, en autorisant la prise en compte, au niveau considéré, d'informations dont la séman-

tique est d'un niveau différent ; c'est le principe de base du cross-layering. Par exemple, [AKA 04] utilise des informations de niveau MAC comme paramètres d'entrée de son adaptation au niveau Middleware/Transport, pour optimiser la QoS de communications multimédias en temps réel dans un environnement de réseaux sans fils hétérogènes.

Adaptation. Les solutions d'adaptation proposées dans la littérature sont multiples et touchent toutes les couches.

Au niveau Applicatif, [WU 01] cible par exemple l'adaptation des applications de streaming vidéo aux variations de performances de l'Internet Best Effort. Il présente différentes techniques d'adaptation par le biais de deux mécanismes : un contrôle de congestion TCP-friendly implantable de trois façons : basé débit, basé écartement ou compression d'images, et un contrôle d'erreur implantable, là encore, de multiples façons : basé retransmissions contraintes temporellement, basé FEC, basé codage, basé interpolation.

[SHA 01] propose d'adapter le contenu des informations multimédias en fonction de l'état courant du client, du système ou du réseau pour prendre en compte les besoins en QoS de bout-en-bout. Il propose de s'appuyer sur RTP (Real time Transport Protocol), pour implémenter un schéma de contrôle basé politique pour les adaptations de niveau application. Plusieurs techniques d'adaptation de contenu pour l'amélioration de performance perçue par l'utilisateur final sont ainsi présentées.

Au niveau Transport, la réaction de TCP aux congestions du réseau (contrôle de congestion de type AIMD) est un exemple bien connu d'adaptation ; cependant, il ne tient pas compte des besoins en QoS temporelles ou en débit de l'application.

DCCP [FLO 06] vise une meilleure adéquation en permettant aux utilisateurs d'activer le mécanisme de contrôle de congestion TCP-friendly (TFRC par exemple [FLO 08]) le moins pénalisant au regard de ses besoins en QoS. Cependant, il n'offre pas plus de QoS qu'UDP et ne permet pas de répondre (par exemple) aux besoins minimums en fiabilité des applications multimédias. SCTP cible une adaptation aux pannes du réseau via le concept de multi-connexion et de multi-homing [STE 07].

[AKA 04] traite du déploiement de différents types d'applications entre utilisateurs mobiles dans l'Internet sans fil hétérogène de prochaine génération. Face à leurs besoins en QoS, il propose une couche de Transport au-dessus de UDP et TCP indépendante du type de réseaux sous-jacents, visant à s'adapter aux caractéristiques du réseau d'accès en cours d'utilisation. L'adaptation proposée repose sur le paramétrage dynamique de mécanismes de contrôle de congestion par le biais d'informations fournies par le niveau MAC. L'adaptation est locale à la machine émettrice.

[WON 01] [EXP 03b] [MOC 05] abordent l'adaptabilité des protocoles de Middleware/Transport sous l'angle architectural, par le biais de *framework* autorisant l'assemblage dynamique de mécanismes protocolaires (*micro protocoles*, *processing modules*). La composition vise à rendre un service de Transport adapté à un besoin donné, en modifiant au besoin la structure interne du protocole.

Au niveau Réseau, [DAS 04] et [CHE 05] abordent le problème du routage ad-hoc orienté QoS. [SKA 04] cible la configuration dynamique et sécurisée de services IP au sein d'une coalition de réseaux fixes/mobiles gérés de façon indépendante à l'image des AS de l'Internet multi-domaines. Sa solution suit une approche PBNM et définit quatre phases : définition d'une relation de confiance entre PDP, négociation de la politique entre PDP, configuration dynamique des éléments de chaque AS (routeur, firewall, ...), et enfin établissement d'un VPN pour le transport des données relatives au service configuré. Cependant, il ne définit pas comment redéfinir la politique en cas de problème, laissant cette tâche de l'administrateur. Dans un contexte analogue, [SAM 05] aborde le problème de l'élaboration de politiques auto-adaptatives, c'est-à-dire capable de se reconfigurer en cas de problème. Sa solution repose sur l'apprentissage du contexte courant.

Au niveau MAC, [BAL 04] cible le processus de décision du handover vertical (i.e. redirection dynamique des flux applicatifs d'une interface de réseau sans fil vers une autre) pour remédier aux problèmes de déconnexion et de QoS posés à un utilisateur mobile dans un contexte de terminaux hétérogènes et de réseaux sans fil hétérogènes. Une architecture de gestion du handover à base de clusters est proposée. Elle inclut un gestionnaire d'adaptation par cluster qui sélectionne le réseau adéquat à partir de la localisation de l'utilisateur, de ses exigences et préférences, et de la QoS requise par l'application. [KAL 06] aborde une solution qui vise à améliorer la latence lors du processus de handover. Cependant, les contraintes de QoS (de type bande passante par exemple) ne sont pas considérées.

Caractéristiques de l'adaptation. Les solutions d'adaptation proposées dans la littérature sont qualifiables de différentes façons.

Comportementale vs. architecturale. A l'image de TCP et des solutions proposées par [WU 01] ou [AKA 04], l'adaptation peut être qualifiée de comportementale (ou d'algorithmique), lorsque le comportement des composants adaptatifs est paramétrable ou modifiable. La structure interne des composants n'est en revanche pas modifiée, ce qui facilite l'implémentation mais limite la capacité d'adaptation. En effet, elle peut nécessiter l'ajout de nouveaux comportements, non initialement prévus. Il est alors nécessaire de recompiler le composant et l'adaptation ne peut se faire en temps réel. L'adaptation est architecturale lorsque la structure interne du composant adaptatif peut être modifiée. A l'image de [WON 01] et [MOC 05], le framework proposé dans [EXP 03b] vise à la conception d'un protocole de Transport dont la structure peut être modifiée en fonction des besoins applicatifs et les contraintes du réseau. Le remplacement d'un module protocolaire par un autre est facile à implanter si le composant remplaçant possède les mêmes interfaces que le composant remplacé ; l'ajout ou le retrait de composant(s) est plus complexe à implanter : [EXP 03b] fournit une solution en dotant tous les composants de la même interface.

Verticale (local) vs. Horizontale (distribuée). L'adaptation peut concerner un composant local à une machine ou distribué entre plusieurs machines (typiquement, un protocole). Dans le premier cas, l'adaptation peut être

qualifiée de verticale. Dans le second cas, on peut la qualifier d'horizontale. [BRI 01] soulève les problèmes de synchronisation des entités adaptatives paires dans le cas d'une adaptation horizontale et propose un protocole de gestion de cette synchronisation.

Run time vs. Design time.

L'adaptation est qualifiée de *run-time* quand elle est réalisée pendant l'exécution du service. Lorsqu'il faut recompiler le logiciel considéré, l'adaptation est qualifiée de *design-time*.

Gestion de l'adaptation et de son autonomie. La gestion de l'adaptation et de son autonomie consiste à définir comment guider l'adaptation et comment la déployer. Ceci nécessite des réponses à plusieurs besoins liés :
- à la découverte des services/composants,
- à l'évaluation de l'état des ressources du réseau et des machines : besoin en métrologie du réseau, des ressources machines, ...
- à l'évaluation des performances de l'adaptation en vue de la modifier si elle s'avère inadaptée,
- au déploiement de nouveaux paramètres et/ou composants nécessaire à l'adaptation, ce qui nécessite une signalisation, une capacité à intégrer en temps réel ces données, ...
- et de façon plus générale, à la conception d'une architecture de mise en œuvre des solutions aux besoins énoncés.

Adaptabilité multi-niveaux. L'intérêt d'une adaptabilité à différents niveaux simultanément a déjà été souligné [BRI 01], [LAN 04] ; elle constitue l'un des enjeux majeurs du cross-layering. Pour les OIU, l'adaptabilité multi-niveaux est nécessaire pour prendre en compte, non seulement les variations des performances du réseau (au niveau des connexions de bout-en-bout par exemple), mais également les changements de coopération induits par l'évolution de la mission.

Cependant, gérer l'adaptation à différents niveaux nécessite de définir comment coordonner les différentes adaptations ; à défaut, elle peut conduire à des performances bien en deçà de celles ciblées. Par exemple, dans l'objectif d'optimiser la QoS de bout-en-bout, une adaptation du débit d'émission à une dégradation des performances du réseau, à la fois au niveau Application (e.g. par réduction de la taille des images) et au niveau Transport (e.g par un contrôle de congestion de type TFRC [FLO 08]), peut s'avérer trop importante et conduire ainsi à une solution sous optimale.

1.3.2 Les cadres architecturaux pour la gestion de l'adaptation

Offrir un niveau QoS, qu'il soit garanti ou adaptatif, nécessite la mise en œuvre d'une architecture de contrôle incluant les différentes étapes à franchir pour aboutir à une prise de décision et à son application au niveau des entités impliquées dans la fourniture de QoS.

Plusieurs cadres très généraux ont été proposés en ce sens, d'abord pour la gestion de réseau basée politique, puis de façon générale pour la gestion de systèmes informatiques autonomes. Nous introduisions ci-après

deux de ces cadres, dans lesquels s'inscrivent nos contributions en termes d'architecture et pour la prise de décision.

Cadre architectural pour la gestion de réseaux basée politique : la proposition PBNM.

La notion de politique a été introduite comme une approche prometteuse pour répondre aux problèmes de gestion de réseaux orientés QoS. Une politique correspond à un ensemble de règles de haut niveau qui guident le comportement des composants du réseau impliqués dans la fourniture de QoS. Une fois définies et stockées dans une base de données (répertoire) par les administrateurs du réseau, ces politiques sont ensuite traduites en politiques opérationnelles, exécutables par les composants du réseau impliqués, tels que les routeurs.

Le groupe de l'IETF Policy Framework a développé ce concept et proposé un cadre architectural pour une gestion de réseau basée politique [STE 99]. Ce cadre (appelé PBNM : Policy Based Networking Management) comprend quatre composants : Management Console, Policy Repository, Policy Decision Point (PDP) et Policy Enforcement Point (PEP). Ces composants utilisent deux sortes de protocoles : les protocoles d'accès à la base de stockage des politiques opérationnelles et les protocoles de transferts de politiques.

La console de gestion (Management Console) fournit une interface entre l'administrateur et le système de gestion de la politique. Le répertoire de politiques stocke les politiques qui sont applicables à un domaine réseau. Les PDP, appelés également serveurs de politiques, traduisent les politiques de haut niveau stockées dans la base de données, en politiques de bas niveau. En combinant ces informations avec d'autres informations du réseau (concernant par exemple l'état du réseau), le PDP peut produire les règles à implémenter par le PEP. Le PEP reçoit les règles issues du PDP et les déploie sur les éléments du réseau correspondants. Le PEP peut réaliser des actions telles que le filtrage ou le marquage des paquets. En plus de déployer les règles établies par le PDP, le PEP peut envoyer des informations au PDP concernant des changements de configurations par exemple.

La communication entre le PDP et le PEP est réalisée en utilisant un protocole de transfert de politiques tel que COPS (Common Open Policy Service) [DUR 00], développé par l'IETF. Les informations à transmettre par COPS sont stockées dans une PIB (Policy Information Base). COPS peut être utilisé soit en mode Outsourcing, soit en mode Provisionning. En mode Outsourcing, le PEP contacte le PEP chaque fois qu'il a besoin d'informations sur la façon de traiter un évènement particulier. Le PEP répond avec les règles appropriées à utiliser par le PEP. COPS utilise TCP pour une communication fiable. En mode Provisioning, le PDP envoie toutes les informations requises par le PEP à l'initialisation. Toutes les règles pertinentes sont stockées dans le PEP et tous les évènements sont traités en utilisant ces informations, sans qu'il y ait besoin d'une nouvelle communication entre le PDP et le PEP.

[SAM 05] propose d'améliorer les systèmes de gestion de réseau basée

politique pour aborder le problème de la gestion autonome de réseau, notamment DiffServ. Il propose pour cela de s'appuyer sur un modèle hiérarchique de politiques et sur le concept d'apprentissage, permettant de s'adapter dynamiquement aux besoins en QoS.

Nos travaux, menés dans le cadre du projet NETQoS, s'inspire de cette approche d'une part en déclinant le concept de politique, et d'autre part en distinguant les rôles de PDP et PEP dans l'adaptation du niveau Transport.

Cadre architectural pour les systèmes autonomes : la proposition IBM.

Plus récemment, la volonté affirmée ces dernières années de développer les propriétés d'adaptation des systèmes informatiques, qu'ils soient ou non orientés communication, a conduit IBM a proposer le concept d'Autonomic Computing ou Calcul Autonomique pour aborder le problème de la conception de systèmes capables de s'adapter et de répondre aux changements de l'environnement [IBM 06].

Le calcul autonomique est une métaphore biologique qui vise, par exemple dans le corps humain, à permettre à toute personne de prendre des décisions et de faire des efforts sans s'encombrer du calcul du rythme cardiaque ou de la quantité d'adrénaline nécessaire à son fonctionnement. Les systèmes conçus selon ce paradigme doivent donc être capables d'anticiper les besoins et de résoudre les problèmes avec un minimum d'intervention humaine.

Pour atteindre cet objectif, un système autonome doit comporter plusieurs capacités :
- une capacité d'auto-configuration (*self-configuring*), afin de pouvoir s'adapter dynamiquement aux changements d'environnement,
- une capacité d'auto-réparation (*self-healing*, pour découvrir, diagnostiquer et agir afin de remédier aux perturbations du système,
- une capacité d'auto-optimisation (*self-optimizing*), pour ajuster l'utilisation des ressources et équilibrer les charges de travail afin de maximiser l'utilisation de ces ressources,
- et enfin une capacité d'auto-protection (*self-protecting*), afin d'anticiper, de détecter, d'identifier et de protéger le système vis-à-vis des menaces.

Nos travaux ne se revendiquent pas pleinement du cadre donné par IBM mais s'inscrivent cependant dans les aspects de *self-configuring* et *self-optimizing* pour la gestion de la QoS.

La section suivante présente maintenant plus en détails les propositions de gestion de la QoS au niveau Transport, cible privilégiée de l'auto-adaptation pour la QoS ciblée par nos travaux.

1.4 QoS au niveau de la couche Transport

La couche Transport se situe au niveau 4 de la pile protocolaire du modèle OSI. Elle a pour but de fournir des solutions de bout-en-bout pour le transfert des données depuis une application émettrice vers une application réceptrice à travers une interconnexion de réseaux hétérogènes.

Les services rendus par la couche Transport sont directement liés à ceux fournis par la couche Réseau, notamment pour ce qui concerne leur QoS.

1.4.1 Les protocoles de Transport classiques

L'IETF a standardisé plusieurs protocoles de Transport. Nous pouvons citer TCP, UDP, SCTP, DCCP. Cependant, les protocoles de Transport principalement utilisé dans l'Internet sont TCP (Transmission Control Protocol) et UDP (User Datagram Protocol) qui fournissent respectivement un service orienté connexion, en mode flux d'octets, garantissant ordre total et fiabilité totale sur le transfert des données applicatives (TCP) et un service sans connexion orienté message, ne garantissant ni ordre, ni fiabilité (UDP).

TCP [POS 81] base son service de fiabilité totale et d'ordre total sur un mécanisme de retransmission visant à récupérer les paquets perdus. Ce mécanisme entraine une augmentation non contrôlée du délai de bout-en-bout. De plus TCP implémente un mécanisme de contrôle de congestion basé sur les algorithmes de Slow Start et de Congestion Avoidance qui introduit également des variations de débit ou de délai importantes en réponse à des congestions du réseau.

UDP [POS 80], a contrario, n'offre aucune garantie d'ordre ni de fiabilité. Son implémentation ne repose sur aucun mécanisme de contrôle de congestion, ce qui implique que son utilisation n'introduit pas de variation incontrôlée de débit et de délai. UDP est donc plus adapté aux communications multimédias que ne l'est TCP. Cependant, le fait qu'il ne réponde pas aux congestions du réseau présente un risque pour l'Internet puisqu'une utilisation importante d'UDP peut entrainer l'effondrement du réseau dues aux congestions. De plus, quand les flux sont en concurrence pour la bande passante, l'algorithme de contrôle de congestion de TCP entraine un partage équitable de la bande passante. Quand il est utilisé avec des flux UDP, qui n'implémente pas de contrôle de congestion, TCP obtient seulement une part minoritaire de la bande passante totale. Les flux UDP sont alors dits "TCP-unfriendly".

DCCP (Datagram Congestion Control Protocol) [KOH 06] est une alternative aux protocoles TCP et UDP utilisés dans l'Internet. Il offre un service non fiable, non ordonné, semblable à UDP mais fournit un mécanisme de contrôle de congestion qui peut-être choisi par l'utilisateur, comme par exemple TFRC (TCP Friendly Rate Control) [FLO 08]. DCCP est plus adapté aux applications interactives utilisant UDP. En plus de permettre une sélection du contrôle de congestion, DCCP offre d'autres possibilités comme un mécanisme d'acquittement permettant à l'application d'implémenter son propre mécanisme de gestion de la fiabilité.

SCTP (Stream Control Transport Protocol) [STE 00] a été initialement développé pour le transfert des messages de signalisation pour la voix sur IP (Voice over IP). Cependant, son utilisation est maintenant envisagée pour d'autres applications. Les principales caractéristiques de SCTP sont le multi-homing ou encore le concept de multi-flux à l'intérieur d'une unique association. Ce concept d'association permet d'envisager des mé-

canismes spécifiques permettant d'étendre le concept de fiabilité partielle [STE 04]. De plus, des options de TCP sont incluses dans la spécification de SCTP comme le *Selective Acknowledgment*, ou encore la prise en compte du mécanisme d'*ECN* (Explicit Congestion Notification). Enfin, les attaques de sécurité telles que les attaques SYN ne sont plus possibles avec SCTP.

1.4.2 Les nouveaux protocoles de Transport, à architecture configurable

Afin de faire face à des environnements de communication de plus en plus hétérogènes et aux différents besoins des applications, une approche plus récente a été explorée ces dernières années, autorisant une configuration à la demande du comportement ou de structure des protocoles de Transport. Le protocole se veut ainsi adaptable en fonction par exemple des besoins applicatifs ou du contexte réseau.

L'adaptation ciblée peut être de deux types : le premier type peut être qualifiée d'adaptation comportementale, ce qui signifie que le protocole de Transport autorise l'application à spécifier les différents paramètres des mécanismes à implémenter. Ce type d'adaptation a été étendu par le concept d'adaptation architecturale, qui signifie qu'en plus des paramètres des mécanismes que l'utilisateur peut choisir, les différents mécanismes eux-mêmes peuvent être choisis par l'utilisateur. Par exemple, le choix de l'algorithme de contrôle de congestion de DCCP parmi une liste prédéfinie est un exemple d'adaptation architecturale.

La plupart des propositions de cadre (ou framework) de composition de protocoles configurables sont basées sur la notion de micro-protocole. Un micro-protocole peut être défini comme la brique de base que l'on peut composée, soit dans un schéma hiérarchique, soit dans un schéma non-hiérarchique. Un micro-protocole est un module logiciel qui fournit une interface de communication permettant son intégration dans une pile de micro-protocoles. Un framework est un environnement fournissant les outils nécessaires et les mécanismes pour la composition d'une architecture. [VAN 09] propose une classification des approches existantes selon deux axes : les approches hiérarchiques et les approches non-hiérarchiques.

L'approche hiérarchique est basée sur le standard OSI dans lequel les couches de protocole sont empilées les unes au-dessus des autres. Chaque message est ensuite traité en séquence pendant qu'il transite à travers la pile. Différentes architectures suivant le modèle hiérarchique ont ainsi été proposées :

- x-Kernel : conçu au début des années 1990, x-kernel [HUT 91] est destiné au système orienté réseau. Il fournit une API à l'utilisateur pour créer et instancier dynamiquement différents protocoles qui peuvent être utilisés par des applications s'exécutant sur le système. Bien que la proposition x-kernel ne soit pas dédiée spécifiquement au Transport, il s'agit d'une des premières approches modulaires qui a été à la base de nombreuses expérimentations par d'autres concepteurs de framework.
- APPIA : basé sur le modèle hiérarchique introduit dans Ensemble [BIR 00], APPIA [MIR 01] fournit un framework pour la composition

d'une pile de micro-protocoles à un niveau donné de couche protocolaire. La notion de "canaux" est définie dans APPIA comme la succession de micro-protocoles qu'un message applicatif doit traverser. De plus, APPIA définit deux types de message différents : les messages internes dont le rôle est de servir la communication intra-couche, et les messages externes qui doivent être traités.

D'autres modèles ont été proposées pour améliorer les performances des protocoles configurables. Ces modèles sont basés sur le concept de souscription aux notifications d'évènements. Les principales propositions sont Coyote, ADAPTIVE et Cactus.

- Coyote : introduit par l'Université d'Arizona, Coyote [BHA 98] est le premier framework de composition de protocoles non hiérarchiques, basé sur des évènements. Il améliore le modèle hiérarchique de x-kernel en autorisant une composition non-hiérarchique des différentes couches de x-kernel. Le système est composé d'un contrôleur principal et de différents micro-protocoles. Les évènements qui guident l'exécution du système sont définis soit par le framework, soit par les micro-protocoles eux-mêmes. Sur occurrence d'un évènement, le framework notifie itérativement tous les micro-protocoles qui ont souscrit à cet évènement.
- ADAPTIVE : l'environnement de communication ADAPTIVE (ACE) introduit une approche non hiérarchique pour concevoir des protocoles configurables [SCH 93]. ACE automatise la configuration et la reconfiguration du logiciel de communication en liant dynamiquement les services aux applications en cours d'exécution et en mettant en œuvre ces services avec un ou plusieurs processus ou threads. Ces services sont implémentés par une collection de composants orientés objets.
- Cactus : il s'agit d'une extension du système Coyote présenté ci-dessus permettant d'inclure des mécanismes de gestion de la QoS. Cactus introduit l'idée selon laquelle le comportement des micro-protocoles peut être changé "en cours d'exécution". Cependant, les implémentations sont limitées puisque tous les micro-protocoles qui pourraient être requis à n'importe quel moment au cours du processus de communication, doivent être instanciés quand l'exécution du protocole démarre.

Inspiré de ces différentes contributions, ETP est une proposition de cadre pour la configuration dynamique de protocoles de Transport orientée QoS. Nos travaux s'appuyant directement sur cette proposition, nous détaillons à présent ce framework.

1.4.3 Le Framework ETP : Enhanced Transport Protocol

En combinant les avantages des framework hiérarchiques et non hiérarchiques, ETP (Enhanced Transport Protocol) [EXP 03a] propose une approche hybride où un plan hiérarchique est défini pour implémenter des fonctions de contrôle de QoS et un plan non-hiérarchique pour réaliser des fonctions de gestion de la QoS. En d'autres termes, les actions effectuées par paquet sont réalisées suivant le modèle hiérarchique alors que les opérations liées à la gestion de la connexion suivent le paradigme orienté évè-

nement. Cette approche permet de clairement séparer les différents plans qui interviennent pour fournir un service de Transport donné.

Le framework ETP a été conçu pour répondre à trois besoins nécessaires à la conception d'un protocole de Transport configurable orienté QoS :
- permettre aux applications de communiquer leurs besoins spécifiques en QoS ;
- effectuer le choix des mécanismes permettant de satisfaire au mieux ces besoins en tenant compte des contraintes du réseau ;
- définir l'architecture de mise en œuvre de ces mécanismes, celle ci devant autoriser l'ajout de nouveaux mécanismes pour s'adapter à d'autres contextes de réseaux et d'applications. Ces problématiques ont été traitées en utilisant des mécanismes issus du monde de l'Intelligence Artificielle par [VAN 09].

L'architecture de mise en œuvre des mécanismes de Transport s'articule autour de deux plans : un plan de contrôle et un plan de gestion :
- plan de contrôle : il contient les opérations exécutées de manière synchrone sur les données applicatives (ADU) émises ou reçues. C'est dans le plan de contrôle qu'est réalisée la composition des mécanismes permettant d'appliquer la politique choisie pour offrir le service attendu. L'implémentation de ce plan est faite suivant le modèle des architectures hiérarchiques de type V_STREAMS [RIT 84]. Les mécanismes sont implantés sous la forme de composants (classes JAVA par exemple) appelés modules de traitement (*processing modules*), communicants par message et s'exécutant de manière séquentielle. Cette structure permet de remplacer facilement un mécanisme par un autre ou d'ajouter un mécanisme, si cela est nécessaire ;
- plan de gestion : c'est dans le plan de gestion que sont décidées, évaluées et modifiées au besoin les politiques appliquées par le plan de contrôle. L'implémentation de ce plan est faite selon le modèle des architectures non-hiérarchiques du type Cactus [GAR 01] ou ADAPTATIVE [SCH 93]. Concrètement, il s'agit d'opérations (mécanismes locaux ou protocolaires) effectuées sur les flux de données de manière asynchrone. Là encore, ces opérations sont implantées à l'aide de *processing modules*. Comparativement aux opérations du plan de contrôle (qui portent sur chaque paquet individuellement), les opérations du plan de gestion sont réalisées sur des échelles de temps plus grandes, car elles nécessitent par exemple des informations statistiques relatives au réseau. Nous verrons cependant dans la suite que le plan de gestion inclut également des traitements par paquet.

La section suivante détaille l'architecture d'implémentation de ces deux plans.

1.4.3.1 Implémentation des plans de contrôle et de gestion

Pour implanter ces deux plans, le framework ETP propose une architecture unifiée (en émission comme en réception) comprenant un composant pour le plan de gestion et deux composants pour le plan de contrôle :

– le composant du plan de gestion (mg de la Figure 1.1) est chargé de récupérer les informations provenant du réseau ou des applications, et de faire appliquer la politique de QoS adéquate. Par exemple, mg peut calculer le débit d'émission nécessaire à l'évitement des congestions et communiquer cette valeur aux composants du plan de contrôle.
– les composants du plan de contrôle chargés d'appliquer la politique définie au niveau du mg, sont le ControlOUT pour le traitement des flux de données sortants et le ControlIN pour les flux de données entrants. C'est par exemple au niveau du composant ControlOUT que sera appliquée la politique de mise en forme du trafic pour respecter le débit d'émission, calculé par le mg.

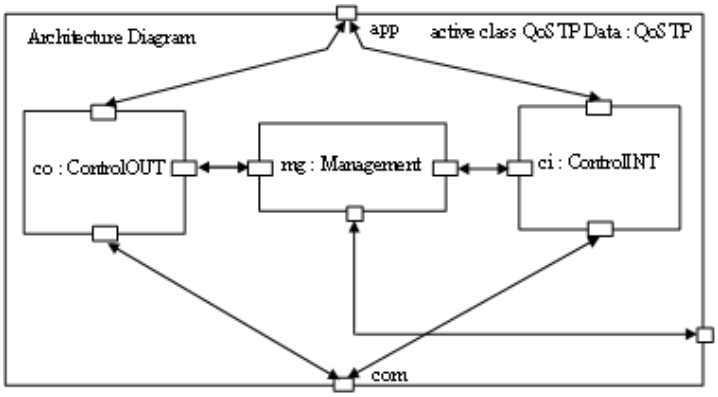


FIGURE 1.1 – *Architecture d'une entité de Transport*

Pour adapter le protocole au contexte applicatif et/ou réseau, le choix effectué repose sur la composition dynamique de mécanismes. Cette composition apparaît dans l'architecture interne des composants des plans de contrôle et de gestion : celle ci n'est pas fixe mais peut être constituée d'autant de processing modules que nécessaires. Pour charger dynamiquement les modules protocolaires appropriés, une description en langage XML de l'architecture de processing modules désirée doit être fournit à ETP (par exemple lors de la création d'un socket par l'application). Le framework fournit tous les outils nécessaires pour convertir la description de la composition en XML en l'instance du protocole de Transport correspondant.

Les sections suivantes décrivent à présent les mécanismes protocolaires ayant un impact sur la QoS offerte aux applications. Les mécanismes classiques (non orientés QoS) sont tout d'abord présentés, puis les mécanismes orientés QoS.

1.4.4 Mécanismes de Transport Classiques

Les principaux mécanismes de Transport ayant un impact direct sur la QoS sont le contrôle de congestion et le contrôle d'erreur.

1.4.4.1 Contrôle de congestion

Le contrôle de congestion est un mécanisme visant à adapter le débit d'émission des entités Transport en fonction de l'état du réseau, en particulier en cas de congestion d'un ou plusieurs routeurs impliqués dans le chemin de données des connexions de Transport considérées. Deux types de contrôle de congestion existent dans la littérature [WU 00] : orienté fenêtre et orienté débit.

1.4.4.1.1 Contrôle de congestion orienté fenêtre Ce mécanisme est celui utilisé par le protocole TCP dans sa version première [POS 81]. Le principe est de tester la bande passante disponible au niveau du réseau en incrémentant une fenêtre dite « de congestion » limitant l'envoi des données sur le réseau. La détection de perte est ensuite considérée comme une congestion du réseau et entraîne la diminution de cette fenêtre. Les mécanismes du contrôle de congestion de TCP sont le Slow Start et le Congestion Avoidance. Ce mécanisme entraîne une variation du débit d'émission. De plus, il est généralement couplé à un mécanisme de reprise des pertes (par retransmission) induisant une augmentation du délai de transmission des données potentiellement inacceptables pour les applications multimédias.

1.4.4.1.2 Contrôle de congestion orienté débit Ce type de contrôle vise à diminuer l'amplitude des variations de débit induites par un contrôle de congestion orienté fenêtre. Il peut être mis en œuvre à l'initiative de l'émetteur ou du récepteur et comporte deux variantes : basé test et basé modèle.

- Contrôle à l'initiative de l'émetteur
  - Basé test : cette technique consiste à adapter le débit d'émission grâce à une estimation de la bande passante disponible. Pour cela, l'application utilise un paramètre de QoS (comme le taux de perte par exemple) et augmente (respectivement diminue) le débit d'émission si ce taux est au-dessous (respectivement au-dessus) d'un certain seuil. Notons que les entités communicantes doivent mettre en place un mécanisme de feedback leur permettant d'échanger l'information « taux de perte ».
  - Basé modèle : cette technique se base sur un calcul plus explicite de la bande passante disponible grâce à un modèle caractérisant le débit d'une connexion TCP. Ce modèle est donné par la formule 1.1 suivante :

$$D = \frac{1.22 \times MTU}{RTT \times \sqrt{p}} \tag{1.1}$$

avec :
D = débit d'émission
p = taux de pertes
MTU = unité maximale de transmission du réseau
RTT = temps aller-retour des paquets
On remarque que le calcul du RTT nécessite de connaître la date de réception du paquet. Un mécanisme de feedback et d'estampillage

temporel des paquets est alors nécessaire. Un contrôle basé sur un tel modèle permet aux entités émettrices d'être TCP-friendly.

– Contrôle à l'initiative du récepteur
  - Basé test : cette technique est basée sur le principe d'abonnement à des couches correspondant à des débits d'émission (et donc à des qualités de présentation) qui augmentent graduellement. Cette méthode est utilisée dans les environnements multipoints où les récepteurs s'abonnent à des groupes différents correspondant à des débits d'émission différents. C'est une méthode empirique : le récepteur s'abonne à une couche ; s'il y a congestion, il s'abonne à une couche impliquant la réception d'un débit plus faible.
  - Basé modèle : le récepteur estime explicitement la bande passante disponible (grâce à la formule 1.1 par exemple) et s'abonne en conséquence à la couche correspondante.

Notons que l'on peut également envisager de coupler les mécanismes à l'initiative de l'émetteur et du récepteur. On parle alors de contrôle de débit « hybride ».

1.4.4.2 Contrôle d'erreur

En dépit de l'utilisation des méthodes exposées ci-dessus, les pertes restent cependant inévitables et une récupération des pertes s'impose. Dans l'optique de prendre en compte d'autres besoins en QoS que la fiabilité (par exemple le délai ou le débit), il convient d'étudier les conséquences de l'application des mécanismes retenus.

Les techniques dites de FEC (Forward Error Correction) consistent à appliquer des bits de redondance dans les paquets à transmettre afin de faciliter leur correction (voire leur reconstitution). Il faut cependant prendre en compte l'augmentation de débit qui en résulte.

Les techniques de type ARQ (Automatic repeat request) sont envisageables si l'on prend en compte la contrainte de délai. Autrement dit, une retransmission est possible si l'arrivée d'un paquet retransmis ne dépasse pas un seuil temporel fixé. Notons que la détection des pertes en réception nécessite une numérotation des paquets.

Nous détaillons à présent ces deux types de techniques.

1.4.4.2.1 Forward Error Correction (FEC)

Les mécanismes de type FEC [LUB 02] sont basés sur l'ajout de bits de redondance ou d'informations supplémentaires (flux redondants par exemple) permettant la reconstitution du flux si certains paquets ont été perdus lors de la transmission.

Au niveau Application, un premier mécanisme, basé sur le codage à la source (SFEC), consiste à ajouter dans les paquets de données plusieurs versions plus ou moins compressées du flux émis. Le mécanisme de recouvrement des pertes, permet à l'entité de pouvoir toujours présenter à l'utilisateur final le flux émis, même avec une qualité de présentation réduite.

Au niveau Transport, les mécanismes de type FEC ne peuvent pas utiliser des méthodes de compression comme celles évoquées ci-dessus. On utilise donc des méthodes basées sur le codage canal qui consiste à ajouter pour un nombre n de paquets de données, k bits de redondance. Contrairement au mécanisme basé sur la compression au niveau applicatif, le recouvrement des pertes grâce au codage canal permet de reconstruire le flux avec la même qualité que le flux émis.

Les principaux inconvénients des mécanismes de type FEC résultent de l'augmentation du débit d'émission due aux informations de redondance, ainsi que de l'augmentation de délai due aux algorithmes de codage et de compression.

1.4.4.2.2 Automatic Repeat Request (ARQ) Les mécanismes ARQ [FAI 02] sont basés sur la retransmission des paquets perdus. Nous présentons dans la suite les principaux protocoles existants basés sur ce principe.

– Stop-and-Wait : ce mécanisme consiste pour l'émetteur à envoyer un paquet de données et à attendre son acquittement avant l'envoi de la prochaine donnée. Coté récepteur, l'acquittement d'une donnée est envoyé si elle a été reçue en séquence. Cela nécessite donc une numérotation des paquets en émission, permettant également de détecter les paquets dupliqués. On peut également mettre en place un mécanisme de détection des paquets reçus en séquence mais corrompus lors du transfert. L'entité réceptrice peut alors envoyer un acquittement négatif (NACK) pour demander la retransmission du paquet corrompu. Sur réception d'un acquittement négatif, l'entité émettrice retransmet le paquet. Sur réception d'un acquittement positif, elle envoie la donnée suivante. Pour résoudre le problème lié à la perte d'un acquittement, l'entité émettrice doit mettre en place un temporisateur de retransmission (timer) associé à chacun des paquets envoyés. Sur expiration de ce timer, le paquet correspondant est retransmis.

– Go-Back-N : contrairement au Stop-and-Wait, ce mécanisme permet l'envoi de plusieurs données successivement, avant réception du premier acquittement. Ces données, non encore acquittées, sont stockées jusqu'à la réception d'un acquittement. En réception, une donnée reçue en séquence entraîne l'envoi de l'acquittement correspondant à la donnée reçue. Si une perte est constatée (par l'arrivée d'une donnée hors séquence), tous les paquets ayant un numéro de séquence supérieur au numéro attendu sont rejetés. Sur occurrence d'une perte, deux mécanismes peuvent être mis en place par le récepteur. Soit il envoie un acquittement négatif (NACK) correspondant au premier paquet hors séquence, soit il n'envoie aucun acquittement. Au niveau de l'émetteur, la réception d'un acquittement positif permet de libérer une place dans la fenêtre d'émission (fenêtre glissante). La réception d'un NACK entraîne la retransmission de tous les paquets stockés à partir du paquet perdu. Si le récepteur n'envoie pas de NACK ou si l'acquittement est perdu, la retransmission se fait grâce à la mise en œuvre de timer au niveau de l'émetteur. De la même manière, l'expiration d'un timer de retransmission entraîne la réémission de tous les paquets stockés à

partir du paquet considéré comme perdu (c'est-à-dire dont le numéro de séquence correspond au timer de retransmission).

– Selective Repeat : le Selective Repeat est le contrôle d'erreur le plus complexe mais le plus efficace des mécanismes de contrôle d'erreur. Contrairement au Go-back-N où la réception d'un NACK ou l'expiration d'un timer entraîne la réémission de toute la fenêtre de retransmission, un émetteur utilisant un Selective Repeat ne retransmet que les paquets explicitement demandés par le récepteur. Ceci permet de réduire le nombre de retransmissions inutiles mais accroît la complexité des entités émettrice et réceptrice. Notons que l'entité réceptrice doit acquitter individuellement chaque paquet et doit mémoriser les paquets arrivés hors séquence.

– Une mise en œuvre améliorée du SR a été proposée dans le protocole TCP via l'implémentation du concept de Selective Acknowledgment (SACK). Par l'envoi d'un SACK, l'entité réceptrice informe l'émetteur de la réception de tous les paquets reçus (y compris donc ceux reçus hors séquence). Ainsi, l'émetteur ne retransmet que les paquets perdus.

Remarque : les mécanismes de type ARQ entraînent une augmentation non contrôlée du délai. De plus, les mécanismes présentés ci-dessus offrent un service d'ordre total et de fiabilité totale sur le transfert des données. En conséquence, ces mécanismes peuvent être incompatibles avec les besoins temporels et ne prennent pas en compte les besoins d'ordre et de fiabilité partiels des applications multimédias.

1.4.4.3 Conclusion

Développer des mécanismes de contrôle d'erreur et de congestion au niveau Transport plutôt qu'au niveau applicatif offre des avantages en termes de généricité et de réduction de la complexité pour les applications. Cependant, certains mécanismes restent du ressort de l'application. C'est le cas de certaines techniques de FEC comme SFEC, déjà évoqués précédemment.

Les mécanismes de contrôle de congestion, notamment ceux orientés débit, peuvent être mis en œuvre au niveau Transport et permettent d'adapter le débit d'émission aux congestions du réseau. Cependant, la stratégie utilisée par les entités émettrices pour réduire le débit peut prendre plusieurs formes. Elle peut par exemple consister à retarder l'envoi de certains paquets de manière à rester conforme à un débit maximal autorisé. Cependant, cette politique peut s'avérer inadéquate pour les applications multimédias (ou plus généralement temps réel) qui n'accepteraient pas une augmentation trop importante du délai de transit.

On peut alors, par exemple, envisager non plus de retarder l'envoi des paquets, mais plutôt d'envoyer uniquement les paquets les plus prioritaires. On peut également modifier la compression ou le codage des données à émettre, au coût d'une diminution acceptable de la qualité du flux. Ces mécanismes, naturellement du ressort de l'application, sont actuellement

mis en œuvre dans certaines des applications multimédias utilisées dans l'Internet [WU 00].

1.4.5 Mécanismes de Transport pour le multimedia

De nouveaux concepts ont été proposés au niveau Transport pour offrir une meilleure QoS que celle offerte par les protocoles UDP et TCP, en tirant parti des caractéristiques des flux véhiculés.

1.4.5.1 Contrôle de congestion orienté QoS

1.4.5.1.1 TFRC TCP-Friendly Rate Control [FLO 08] est un mécanisme de contrôle de débit à l'initiative de l'émetteur basé sur un modèle. Il a été conçu pour minimiser les variations brusques de débit afin de répondre aux besoins des applications multimédias. Il permet également de respecter les flux TCP vis-à-vis de l'utilisation de la bande passante.

L'émetteur ajuste le débit d'émission des paquets en fonction des feedback envoyés par le récepteur. En fonction des informations contenues dans le feedback, l'émetteur calcule le débit d'émission selon la formule 1.2 suivante :

$$TFRC = \frac{s}{R\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \tag{1.2}$$

avec $TFRC$ le débit en octets/s, s la taille d'un paquet en octets, R le RTT en secondes, t_{RTO} la valeur du timer de retransmission en secondes, $p \in [0..1]$ le taux d'évènements de pertes calculé par l'émetteur, b le nombre maximal de paquets acquittés par un acquittement TCP. Si aucun feedback n'est reçu, l'émetteur diminue le débit d'émission par deux.

TFRC répond donc à trois besoins dans le cadre d'un mécanisme de transport orientés QoS : le respect des flux TCP (TCP-Friendliness), la minimisation des variations brusques de débit et l'indépendance vis-à-vis du contrôle d'erreur. Cependant, la politique mise en œuvre pour respecter le débit d'émission autorisé consiste à retarder l'envoi des paquets (sans distinction de ces paquets), ce qui peut poser certains problèmes vis-à-vis des contraintes de délai des applications multimédias.

1.4.5.1.2 Contrôle de congestion basé sur TFRC et orienté QoS. Afin de ne pas pénaliser les applications ayant des contraintes de délai strictes, une solution proposée dans [EXP 03a] consiste à ne pas systématiquement retarder l'envoi des paquets, mais à « choisir » les paquets à envoyer prioritairement. Cela suppose que la couche Transport puisse récupérer une description du flux applicatif à transférer lui permettant d'en déduire la priorité des paquets le composant. Le choix d'envoyer ou non un paquet peut se faire également après avoir estimé, grâce au RTT et au temps inter-paquets, la date de réception du paquet. Si la date de réception dépasse un délai acceptable, le paquet peut être considéré par l'application réceptrice comme obsolète.

En s'inspirant de la technique d'abonnement par couches dans le contexte du multicast au niveau IP, ce mécanisme de rejet de paquets en émission a été proposé pour les flux MPEG composés de trois types d'images : I, B, et P.

Il définit 3 niveaux (par exemple) de qualité :
– Couche 2 : images I, P et B ;
– Couche 1 : images I et P seulement ;
– Couche 0 : images I seulement.

En fonction de l'estimation de la date de réception des paquets, l'émetteur adapte son débit d'émission en « s'abonnant » à la couche adéquate, c'est-à-dire en envoyant les paquets en fonction de leur priorité.

1.4.5.2 Contrôles d'erreur orientés QoS

Les mécanismes de contrôle d'erreur ont fait l'objet de nombreux travaux, visant à en améliorer les performances ou à en étendre les fonctionnalités.

1.4.5.2.1 Contrôle d'erreur à garantie d'ordre et fiabilité partiels Alors que les protocoles usuels n'offrent soit aucune garantie d'ordre ou de fiabilité sur le transfert des données utilisateur (UDP), soit une garantie d'ordre total et de fiabilité (TCP), les nouvelles applications (en particulier multimédias) présentent des besoins exprimables en termes de fiabilité partielle (des pertes sont parfois tolérables) et d'ordre partiel (les données peuvent être remises à l'application réceptrice dans un ordre non plus total ou nul, mais partiel, correspondant au besoin en synchronisation des flux manipulés - audio et vidéo par exemple).

Face à cette lacune, un nouveau concept de protocole a été introduit, les protocoles à ordre et fiabilité partiels [AME 94], permettant d'offrir tous les services partiellement ordonnés et partiellement fiables, en fonction des besoins des applications (Figure 1.2).

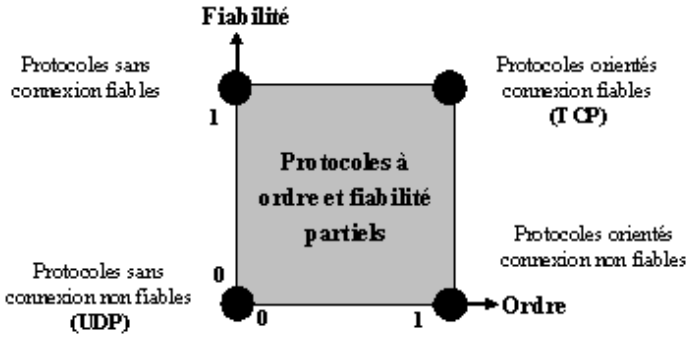


FIGURE 1.2 – *Espace des protocoles à ordre et de fiabilité partiels*

Deux principaux avantages découlent de cette proposition : [OWE 98] montre que l'utilisation de ces protocoles permet :
– d'optimiser l'utilisation des ressources du réseau et
– de minimiser le délai de transit des paquets tout en respectant les

contraintes de fiabilité et de synchronisation logique (ordre partiel) inhérentes au transfert de documents multimédias.

1.4.5.2.2 Fiabilité partielle différenciée et temporisée Le concept de fiabilité partielle différenciée [EXP 03a] est basé sur la notion de couche introduite précédemment. La retransmission d'un paquet est effectuée sur la base des contraintes de fiabilité partielle de l'application réceptrice. Cependant, elle est également couplée à un mécanisme prenant en compte les contraintes de délai, qui peut être mis en œuvre soit au niveau de l'émetteur soit au niveau du récepteur.

Approche émetteur : lorsque l'émetteur constate la perte d'un paquet, il effectue la retransmission seulement s'il constate que cette perte n'est pas acceptable au regard des besoins en fiabilité partielle de l'application réceptrice. Il doit également prendre en compte le fait que le paquet retransmis doit arriver à destination avant d'être considéré comme obsolète ;

Approche récepteur : le récepteur peut lui-même demander explicitement la retransmission d'un paquet en fonction de ses besoins en fiabilité partielle et de ses contraintes de temps. Cependant, il peut être plus difficile pour le récepteur de déterminer quels sont les paquets qu'il souhaite recevoir en priorité ou ce qu'il veut obligatoirement recevoir. Cela suppose qu'il connaisse la structure du flux, ce qui n'est pas toujours possible, par exemple dans le cas des flux MPEG ou H263.

1.4.6 Gestion coopérative de la QoS

En parallèle des travaux précédents, d'autres travaux ont abordé le problème de la QoS Transport sous un angle coopératif.

Introduit dans [CHA 96] et étendu dans [OWE 98], puis plus récemment dans [HEI 03], un concept important est celui de multi-connexion (MC).

Une MC permet à son utilisateur de gérer plusieurs connexions (chacune dédiée à un flux spécifique) entre deux utilisateurs de façon non indépendante. Par exemple, deux connexions d'une même MC (une pour l'audio, une pour la vidéo) peuvent être dégradées de façon différente lorsque elles ont à réagir à une congestion du réseau.

La figure Figure 1.3 illustre les principes d'architecture d'une multi-connexion de niveau Transport à ordre et fiabilité partiels à la fois intra et inter-flux tel que proposée dans [CHA 96].

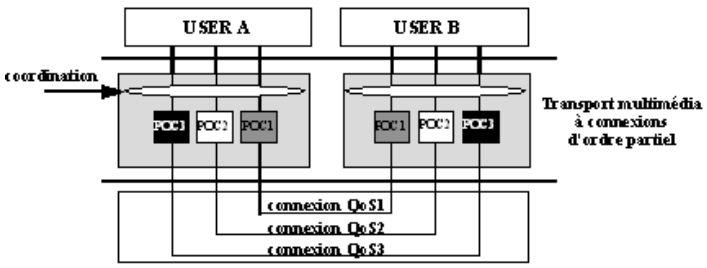


FIGURE 1.3 – *Architecture d'un Transport multimédia à connexions d'ordre partiel*

L'architecture proposée est composée de trois connexions de Transport à ordre et fiabilité partiels (POC : Partial Order Connection), fournissant chacune une QoS répondant aux caractéristiques du flux transporté (texte, audio et vidéo par exemple).

A la différence des approches classiques, les trois connexions sont coordonnées entre elles pour préserver les relations de synchronisation logiques existant entre les flux véhiculés (texte synchronisé en sous titre de l'audio par exemple, lui même synchronisé avec la vidéo).

La coordination est envisagée par le biais de deux paramètres : l'ordre partiel et la fiabilité partielle.

La gestion de l'ordre partiel se situe non seulement au travers de chaque connexion, mais également entre ces connexions. En cela, elle permet de prendre en compte les contraintes de synchronisation logique à la fois intra et inter-flux de l'application. Une telle architecture permet ainsi de mettre en œuvre le service d'ordre partiel multimédia le plus adapté aux contraintes de synchronisation logique (c'est-à-dire hors aspect temporel) de l'application.

L'intégration de la notion de fiabilité partielle repose sur un mécanisme de contrôle des pertes pouvant être mis en œuvre soit « connexion par connexion », soit « par groupe de connexions ». L'objectif de chacun de ces mécanismes est de permettre la remise au plus tôt d'une donnée non « délivrable » au regard de l'ordre partiel (intra et inter-flux), au prix de la déclaration de perte de la(des) donnée(s) qui la précède(nt) dans l'ordre partiel. La déclaration de perte doit respecter la composante « fiabilité » de la QoS requise sur chaque connexion.

Notons également les travaux suivants :

[BAL 01] propose une solution pour traiter le problème du contrôle de congestion pour des agrégats de flux, grâce au *congestion manager* (CM) qui permet de multiplexer plusieurs flux d'une application qui partagent le même chemin.

[OTT 07] aborde le problème de la coordination de flux pour les applications multimédias. Il propose pour cela un protocole permettant aux applications d'obtenir des informations sur l'état du réseau et notamment concernant la bande passante. Les applications sont ensuite libres d'appliquer leur propre schéma de coordination en fonction de leur besoin.

Conclusion du chapitre

L'objet de ce chapitre était de présenter le panorama des solutions de gestion de la QoS dans l'Internet et d'y positionner nos travaux. Nous avons d'abord motivé ces solutions en explicitant les besoins applicatifs, issus en particulier des évolutions des applications, aujourd'hui multimédias (audio, vidéo, texte) et distribuées dans le cadre d'activités potentiellement complexes, telles que les activités coopérative de groupe.

Les différentes solutions présentées ont été classées en deux approches :

l'une basée réservation pour la garantie de la QoS, et l'autre basée adaptation pour l'optimisation de la QoS. La première approche suppose une relative stabilité des ressources du réseau, et trouve son contexte d'application dans l'Internet classique, multi-opérateurs (autrement appelé multi-domaines), basé sur une infrastructure de réseaux essentiellement filaires. Cette hypothèse peut cependant être remise en cause dans des contextes de réseaux amenés à se développer dans le futur, typiquement mobiles, sans fil, ad-hoc, connectés ou non à l'Internet ; dans de tels contextes, des ressources réservées à un instant donné peuvent ne plus être disponibles à un autre instant : l'approche basée réservation doit alors être complétée, voire remplacée par une approche visant à optimiser la QoS en adaptant le système de communication aux variations du contexte. Ce besoin est renforcé si l'on considère des contextes d'applications non indépendantes où l'on souhaite, pendant l'exécution, accorder des priorités à certaines applications par rapport à d'autres en fonction des performances courantes du réseau et des préférences des utilisateurs.

Nos travaux s'inscrivent dans cette deuxième approche et visent plus spécifiquement à contribuer au développement de protocoles de Transport auto-configurables, aptes à répondre aux besoins en QoS d'applications non indépendantes distribuées dans des réseaux ne permettant pas ou n'offrant pas la possibilité de garantir une QoS. L'adaptation ciblée touche les mécanismes locaux ou distribués du protocole qu'il s'agit *in fine* de pouvoir paramétrer, voire remplacer, dynamiquement.

Les problèmes à résoudre pour mettre en œuvre une telle auto-adaptation ont été énoncés dans ce chapitre. Nous en avons traité certains dans le cadre des travaux présentés dans ce mémoire, qui sont énoncés ci-dessous :

– le premier problème consiste à définir l'architecture de mécanismes locaux ou distribués nécessaires au contrôle de l'adaptation. Le chapitre 2 présente notre proposition d'architecture, développée dans le cadre du projet NETQoS, en s'inspirant notamment des solutions de gestion de réseau basée politique, présentées en section 1.3.2. La résistance au facteur d'échelle de cette architecture est considérée dans le chapitre 4, dédiées à la simulation et aux mesures de performances ;
– le second problème concerne les actions d'adaptation ciblées. Les mécanismes de gestion de la QoS au niveau Transport, comme le contrôle de congestion et le contrôle d'erreur, ont été décrits dans ce chapitre. Partant de certains de ces mécanismes, notre deuxième contribution propose une gestion coopérative de la bande passante pour des connexions de niveau Transport non indépendantes, impliquées dans un contexte d'activités coopératives de groupe telles que les opérations d'intervention d'urgence (OIU). Un modèle d'optimisation de la répartition de la bande passante est proposé au chapitre 3 ; l'évaluation des bénéfices et des limites de ce modèle est présentée au chapitre 4.

2 Principes d'architecture d'un système de gestion auto-adaptative de la QoS

Sommaire

2.1 Besoins des acteurs du système NETQOS 37
2.1.1 Définition des acteurs 37
2.1.2 Besoins des utilisateurs et des applications 38
2.2 Description de l'architecture 39
2.2.1 Présentation générale du système 39
2.2.2 Description des services 41
2.2.3 Architecture fonctionnelle du système 43
2.3 Spécification détaillée des composants de l'entité APA 49
2.3.1 Choix remarquables de conception 49
2.3.2 Spécification détaillée du PDM 54
2.3.3 Spécification détaillée du PAM 55
2.3.4 Spécification détaillée du PEM 56
Conclusion 60

Le chapitre précédent a montré que plusieurs solutions d'optimisation ou de garantie de la QoS ont été proposées aux différents niveaux de l'architecture OSI, en particulier aux niveaux Réseau et Transport. Compte tenu des évolutions technologiques actuelles, choisir et adapter de façon automatique les solutions les plus adéquates au regard des contraintes (ou des capacités) de l'environnement réseau et des besoins applicatifs est un enjeu majeur. Faire face à ce défi nécessite cependant de répondre à plusieurs problèmes, qui ont été résumés en conclusion du chapitre 1.

Ce nouveau chapitre présente notre première contribution, liée au besoin de contrôle pour la gestion adaptative de la QoS dans un environnement de réseaux hétérogènes. Inspirée de la gestion de réseau basée politique, la gestion proposée repose sur l'application de politiques adaptatives concourant à la mise en œuvre d'actions d'adaptation, au niveau du réseau et des hôtes (en particulier au niveau Transport), pour l'optimisation de la QoS en fonction de l'évolution des besoins et des contraintes du réseau.

L'architecture proposée résulte d'un travail mené dans le cadre d'un projet européen, le projet NETQoS, dont l'objectif était de concevoir un modèle conceptuel pour l'adaptation orientée QoS au niveau Réseau et Transport, indépendant d'un environnement réseau spécifique. Nos contributions ont porté d'une part sur l'élaboration de l'architecture du système NETQoS, et d'autre part de façon plus spécifique, sur l'élaboration de la structure du composant central du système (dénommé par la suite APA pour *Automated Policy Adaptor*), chargé de concevoir, d'adapter et de faire appliquer des politiques opérationnelles aux entités en charge de la gestion directe de la QoS, typiquement de niveau Transport (en l'occurrence ETP).

La suite de ce chapitre présente nos contributions en termes d'architecture, modélisée en UML, en faisant directement référence au projet NETQoS. Pour concevoir le système, nous nous sommes inspirés de méthodes issues du génie logiciel. Après avoir identifié les acteurs du système et leurs besoins, nous construisons une représentation orientée fonctionnalité par le biais des actions permises par le système. Il en découle une première définition des composants. L'élaboration de diagrammes de séquence permet de spécifier les interactions et les messages échangés entre ces composants. Une démarche itérative conduit ensuite à raffiner la structure des composants, notamment celui de l'APA auquel nous avons plus spécifiquement contribué.

Ainsi, la section 2.1 identifie les bénéficiaires (dénommés « acteurs » en UML) du système NETQoS et leurs besoins. Plus spécifiquement, elle précise les besoins de deux types d'acteurs directement concernés par une gestion adaptative de la QoS au niveau Transport : les applications et leurs utilisateurs.

La section 2.2 s'intéresse au modèle d'architecture générale de NETQoS : les principes généraux du système et les services offerts (donnés sous la forme de Use Cases UML) sont tout d'abord présentés ; les composants fonctionnels majeurs de l'architecture NETQoS sont décrits dans un second temps, ainsi que leurs principales interactions.

Enfin, la dernière partie (section 2.3) détaille le modèle d'architecture du composant spécifique de notre contribution (l'APA), incluant son application à la gestion adaptative d'entités de Transport configurables, en particulier celles du framework ETP présenté au chapitre 1. Notons ici que le modèle proposé pour l'APA est orienté implémentation, sachant que la démonstration réelle de deux prototypes JAVA du système NETQoS a été effectuée lors des deux dernières revues du projet.

2.1 Besoins des acteurs du système NETQOS

Suivant la démarche évoquée précédemment, cette section décrit les acteurs du système NETQoS, et précise leurs besoins.

2.1.1 Définition des acteurs

Plusieurs acteurs ont été identifiés comme bénéficiaires potentiels du système NETQOS (figure 2.1). Nos travaux nous ont amenés à considérer plus spécifiquement les acteurs "Application" et "Utilisateur", directement intéressés par une adaptation de niveau Transport.

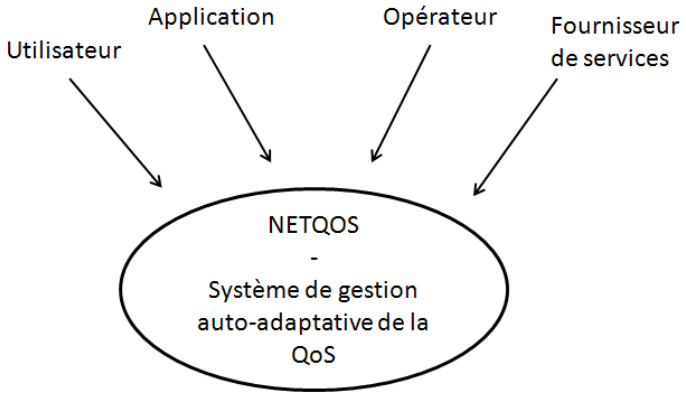


Figure 2.1 – *Acteurs du système NETQOS*

L'utilisateur est une personne physique utilisant un terminal connecté au système NETQOS. Il exécute depuis son terminal le lancement d'applications distribuées pour l'accès à distance à des données ou pour dialoguer avec d'autres personnes. Un utilisateur est le client d'un ou plusieurs fournisseurs de service.

L'application désigne un programme ou ensemble de programmes distribués lancés sur le terminal d'un utilisateur (vidéo à la demande, visioconférence, VoIP, ...) ou sur un serveur d'application d'un fournisseur de service (serveur de vidéo à la demande, serveur de travail collaboratif, passerelle VoIP, ...). Une application peut être consciente du système NETQoS et, par exemple, lui communiquer explicitement ses besoins en QoS lors de son exécution. Dans le cas contraire, le système doit découvrir l'existence de l'application, et satisfaire ses besoins en QoS de façon transparente pour celle-ci. L'utilisateur ou le fournisseur de service peuvent exprimer les besoins d'une application auprès du système NETQoS. Nous avons qualifié le premier type d'applications "NETQoS-Aware" et celles de second type "NETQoS-Unaware".

L'opérateur est une entité responsable de l'administration du réseau. Le réseau est chargé de transporter le trafic généré par les applications utilisées par le fournisseur de service ou les utilisateurs.

Le fournisseur de service est une entité offrant des services aux utilisateurs en utilisant le réseau de l'opérateur. Par exemple, une entreprise offrant un service de Vidéo à la demande ou un fournisseur de passerelle VoIP

autorisant les utilisateurs à appeler des lignes du Réseau Téléphonique Commuté (RTC) depuis leurs terminaux.

La section suivante présente les besoins de deux de ces acteurs, les acteurs "utilisateur" et "application", directement concernés par une adaptation orientée QoS de niveau Transport. Ces besoins sont rappelés succinctement, car ils ont déjà été en partie détaillés dans l'état de l'art (section 1.1).

Les besoins en QoS de l'opérateur et du fournisseur de service relèvent davantage de solutions de niveau Réseau et ne seront pas décrites dans ce manuscrit.

2.1.2 Besoins des utilisateurs et des applications

Utilisateur

L'utilisateur doit pouvoir exprimer ses besoins en QoS auprès du système NETQoS en des termes simples de son point de vue, par le biais d'une interface graphique qui lui masque la complexité technique du système sous-jacent. Il peut ajouter, modifier, supprimer des préférences concernant ses besoins. Par exemple, l'utilisateur peut spécifier quelles applications sont les plus importantes de son point de vue. Il peut aussi spécifier ce qu'il souhaite voir se passer en termes de dégradation de qualité lorsqu'il passe d'un réseau d'accès filaire à un réseau sans fil.

Un exemple de préférence peut être : "En temps normal, je veux pouvoir téléphoner avec une qualité comparable à un téléphone portable. En revanche, lorsque je parle avec une personne dans le cadre professionnel, la qualité doit être aussi bonne que lorsque je téléphone depuis un téléphone fixe".

Pour une même application, l'utilisateur peut également exprimer des priorités ou une importance relative entre les différents flux. Pour une application de travail collaboratif incluant VoIP et tableau blanc, l'utilisateur peut par exemple exprimer qu'il préfère entendre correctement son interlocuteur, même s'il existe un certain délai pour la mise à jour du tableau, qui a moins d'importance.

Outre le besoin d'exprimer ses préférences, l'utilisateur doit être informé des services offerts par le système, de la QoS effectivement fournie ainsi que des informations de facturation.

Application

Les besoins des applications peuvent être exprimés en termes de paramètres de QoS de base ou en des termes plus évolués. Rappelons que la recommandation de l'ITU [ITU01] et [REC 02] définit des paramètres de QoS de base par :

– IPTD (IP Packet Transfert Delay) : le délai de transfert d'un paquet ;

– IPDV (IP Packet Delay Variation) : la variation du délai entre deux paquets consécutifs ;

– IPLR (IP Packet Loss Ratio) : le rapport entre le nombre de paquets perdus sur l'ensemble des paquets transmis ;

– IPER (IP Packet Error Rate) : le rapport entre le nombre de paquets erronés sur le nombre de paquets transmis.

– IPPT (IP Packet Throughput) : le débit requis peut être un paramètre important du point de vue de l'application. Il est égal au nombre total de paquets IP reçus sans erreur sur un intervalle de temps donné.

2.2 Description de l'architecture

Les acteurs du système et leur besoin ayant été identifiés, nous décrivons à présent les principes de conception du système NETQoS.

2.2.1 Présentation générale du système

L'architecture conceptuelle du système NETQoS s'inspire du modèle PBNM présenté dans l'état de l'art en ceci qu'elle distingue plusieurs niveaux de politique ; trois niveaux ont été considérés dans NETQoS :

– les politiques de plus haut niveau correspondent aux besoins et préférences très généraux des acteurs, illustrés par exemple pour l'acteur utilisateur dans la section précédente ;
– les politiques intermédiaires correspondent aux objectifs de QoS réseau à satisfaire pour les applications, enrichis éventuellement par un niveau de priorité ;
– enfin, les politiques opérationnelles correspondent aux choix de services, mécanismes et/ou protocoles mis en œuvre au niveau des entités en charge de la communication, par exemple aux niveaux Réseau et Transport.

Notons d'ores et déjà que l'une des fonctionnalités du système NETQoS sera liée à sa capacité à raffiner le plus haut niveau de politiques en politiques de niveau intermédiaire, puis le niveau intermédiaire en politiques de niveau opérationnel.

Le second modèle architectural dont s'inspire le système NETQoS et le modèle de l'Autonomic Computing également présenté dans l'état de l'art. Etendant ainsi le modèle PBNM, le système NETQoS inclut les fonctions nécessaires à l'adaptation des politiques opérationnelles en fonction de l'évolution du contexte, liée d'une part à la possibilité que les acteurs modifient leur politique de haut niveau, et d'autre part à un changement d'état des ressources du réseau. Les fonctions de supervision des politiques de haut niveau et des performances des politiques opérationnelles,

et les fonctions d'adaptation des politiques opérationnelles sont donc nécessaires.

Dans ce positionnement, quatre entités fonctionnelles ont été identifiées pour le système NETQoS (Figure 2.2). Elles visent à permettre aux utilisateurs du système d'exprimer leur besoins et à celui-ci de les interpréter, d'en élaborer une politique opérationnelle, de la déployer et d'en évaluer l'adéquation aux besoins en vue d'une révision si le contexte change.

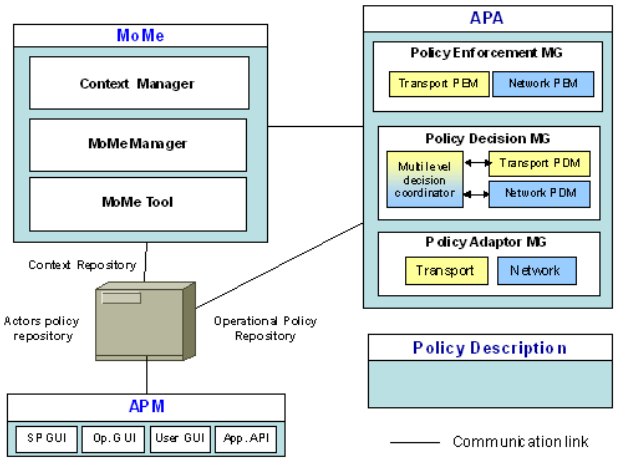


FIGURE 2.2 – *Les entités principales du système NETQoS*

Ces quatre entités, dont les interactions seront décrites ultérieurement, sont les suivantes :

– l'entité de description des politiques (Policy Description - POLD) représente un ensemble d'ontologies utilisées pour spécifier les politiques de niveau acteurs

– l'entité de gestion des préférences des utilisateurs (Actor Preference Manager - APM) est destinée à fournir à chaque utilisateur du système NETQoS une API sous la forme d'une interface graphique (GUI) lui permettant de définir des politiques dynamiques basées sur des ontologies. Ces politiques (besoins, préférences, profils, ...) peuvent être exprimées avant ou pendant la communication.

– l'entité de gestion de l'adaptation (Automated Policy Adaptor - APA) est l'entité centrale du système de communication. Sa fonction est de décider des politiques opérationnelles à déployer (notamment au niveau Transport) pour permettre au système de communication de fournir une QoS optimisée en tenant compte des politiques dynamiques des acteurs, exprimées via l'APM et de l'évolution des performances du réseau.

– le module de supervision et d'analyse/mesure (Monitoring and Measurement - MoMe) est chargé d'observer et de mesurer l'évolution du contexte applicatif et réseau (changement des besoins utilisateurs, chan-

gement des ressources d'extrémité ou, ...). Il doit également évaluer l'efficacité des politiques opérationnelles.

2.2.2 Description des services

Pour aboutir à l'architecture de mise en œuvre de ces entités fonctionnelles, différents cas d'utilisation (Use Case UML) du système ont été définis. Ils permettent d'illustrer les services fournis aux acteurs présentés précédemment. Ces cas d'utilisation sont décrits avec les diagrammes UML adéquats. A titre d'exemple, un Use Case caractéristique est détaillé sous la forme d'un Use Case textuel. Les autres cas d'utilisation textuels sont décrits dans le délivrable [CHA 07d].

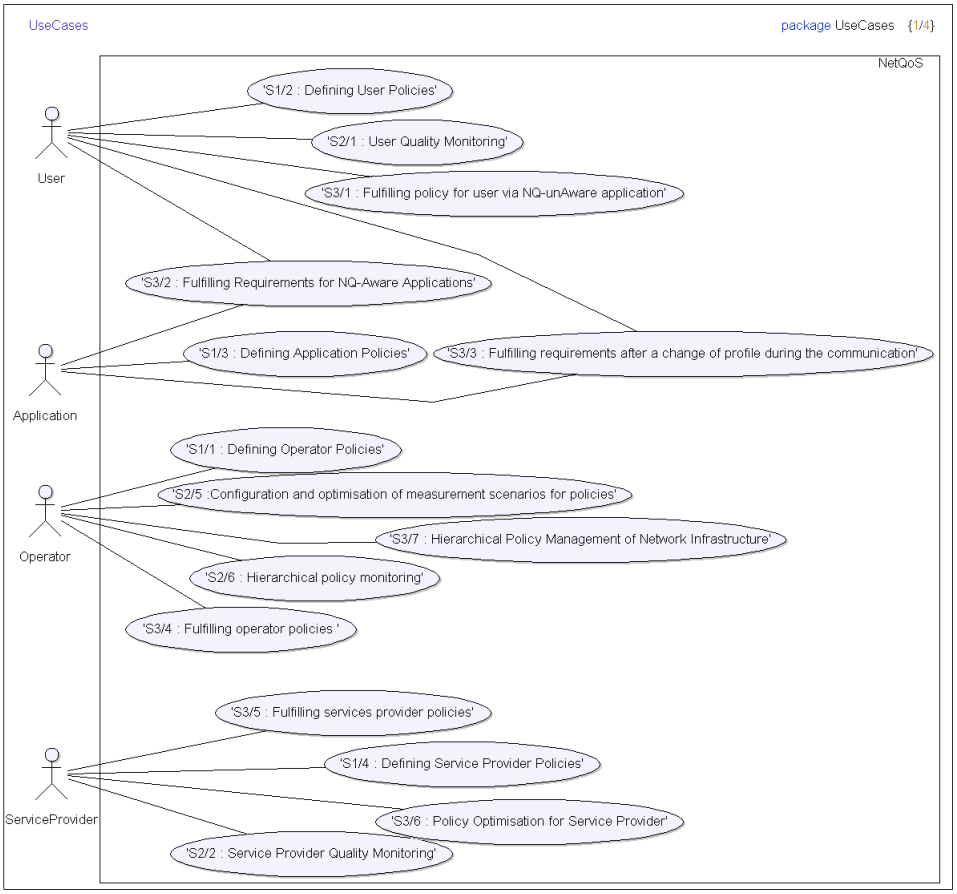


FIGURE 2.3 – *Use Cases du système NETQoS*

2.2.2.1 Classification des Use Cases

Les cas d'utilisation sont classés selon trois catégories correspondant aux trois types de services offerts par le système NETQoS (Figure 2.3) :

- services S1 : Définir les politiques des acteurs
- services S2 : Superviser la QoS sans perturber le fonctionnement des applications de façon non intrusive
- services S3 : Satisfaire les politiques des acteurs en adaptant les pro-

tocoles Transport et en configurant les dispositifs réseaux comme les routeurs notamment.

Nos contributions ont clairement concerné les services de type S3.

La figure 2.3 illustre l'ensemble des Use Cases. On y voit par exemple qu'un utilisateur peut interagir avec le système, notamment en définissant ses préférences d'utilisation (exprimer des priorités sur des applications, sur des flux, etc...), en obtenant des informations de supervision ou en obtenant la QoS qu'il souhaite pour une application.

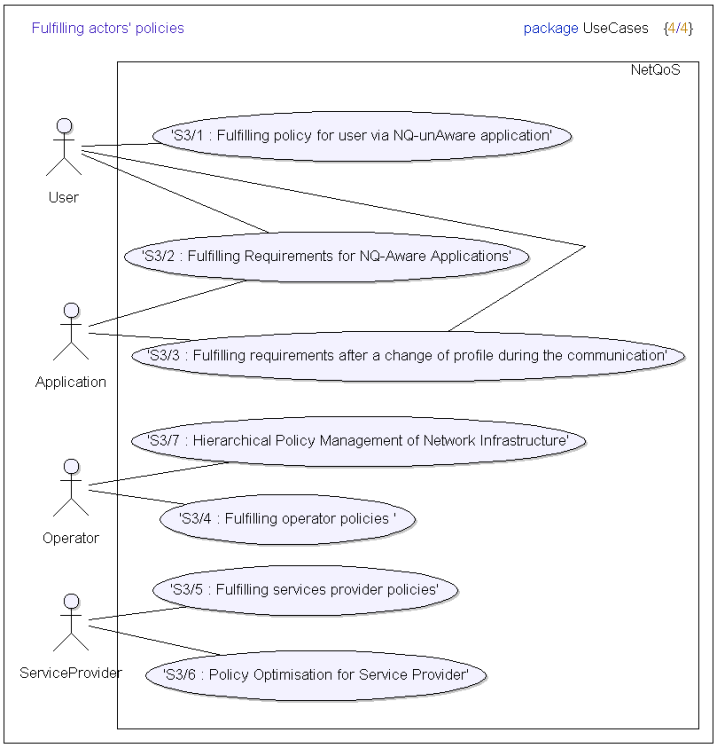


FIGURE 2.4 – *Use Case du service S3 : Satisfaire les politiques des acteurs*

La figure 2.4 illustre tous les Use Cases relatifs aux services d'élaboration et de mise en œuvre des politiques opérationnelles (S3). Nos contributions au projet se situant directement à ce niveau, nous détaillons ci-après l'un de ces services visant à satisfaire la politique d'un utilisateur exécutant une application NETQoS-Unaware.

2.2.2.2 Exemple détaillé : Use Case S3/1, associé à la satisfaction d'une politique utilisateur pour une application "NETQoS-Unaware"

Le Use Case décrit à titre d'exemple dans cette section illustre la prise en compte d'une politique utilisateur pour une application NETQoS-unaware, non avertie de l'existence du système NETQoS.

Ce Use Case permet en particulier d'introduire le rôle de l'entité de gestion de l'adaptation (APA) à laquelle nous avons plus spécifiquement contribué. Pour les besoins de la description qui suit, notons dès à présent que l'APA contient trois blocs fonctionnels qui seront détaillés dans la suite du chapitre (section 2.2.3.2.2) : le gestionnaire du provisionnement initial

des politiques, appelé Policy Decision Manager (PDM), le gestionnaire de l'adaptation des politiques ou Policy Adaptation Manager (PAM), et le gestionnaire de l'application de ces politiques auprès des entités gérées, appelé Policy Enforcement Manager (PEM).

Le diagramme de séquence de ce Use Case est représenté sur la figure 2.5.

L'acteur de ce Use Case est un utilisateur qui exécute une application NETQoS-Unaware. Cet utilisateur est sensé avoir déjà fourni au système NETQoS une expression de la politique qu'il souhaite voir appliquer pour cette application dans le cadre de l'invocation d'un service relevant du service S1. Cette politique de haut niveau est accessible dans un répertoire de stockage des politiques. Le système NETQoS doit alors dynamiquement (c'est-à-dire au lancement de l'application) réaliser les actions nécessaires au niveau des entités de communication gérées (relevant des couches Réseau et Transport) pour satisfaire les besoins/préférences de cet utilisateur.

Les numéros dans le texte du paragraphe ci-dessous correspondent à ceux présents sur le diagramme de séquence de la figure 2.5.

Quand un utilisateur lance une application NETQoS-Unaware (1), le gestionnaire de contexte (CM pour Context Manager, qui est une partie logique du MoMe) détecte l'événement (2) et informe le gestionnaire de provisionnement des politiques (PDM) de l'APA (3). Le PDM récupère alors la politique définie par l'utilisateur pour l'application identifiée (4). En se basant sur ces besoins, le PDM élabore un premier ensemble de politiques opérationnelles permettant de prendre en compte la politique utilisateur (5). Une fois le provisionnement effectué, le transfert de données peut commencer (6). Pendant le transfert, le gestionnaire de supervision des politiques (MoMe) monitore en continue la QoS fournie et informe l'APA en cas de non respect des paramètres requis (7). En fonction de la QoS mesurée, le gestionnaire d'adaptation des politiques de l'APA (PAM) peut déclencher des mécanismes d'adaptation et des changements de politiques opérationnelles élaborées lors de la phase de provisionnement initial (8).

Nous présentons maintenant l'architecture fonctionnelle du système NETQoS, bâtie sur les bases de l'ensemble des Use Cases définis au préalable.

2.2.3 Architecture fonctionnelle du système

2.2.3.1 Diagramme de structure générale

Tel qu'introduit précédemment, le système NETQoS est composé de quatre entités fonctionnelles principales interagissant de manière logique selon le diagramme de structure illustré figure 2.6.

Cette section présente en détail les fonctionnalités de l'entité APA sur laquelle ont porté nos contributions spécifiques à l'architecture du système NETQoS.

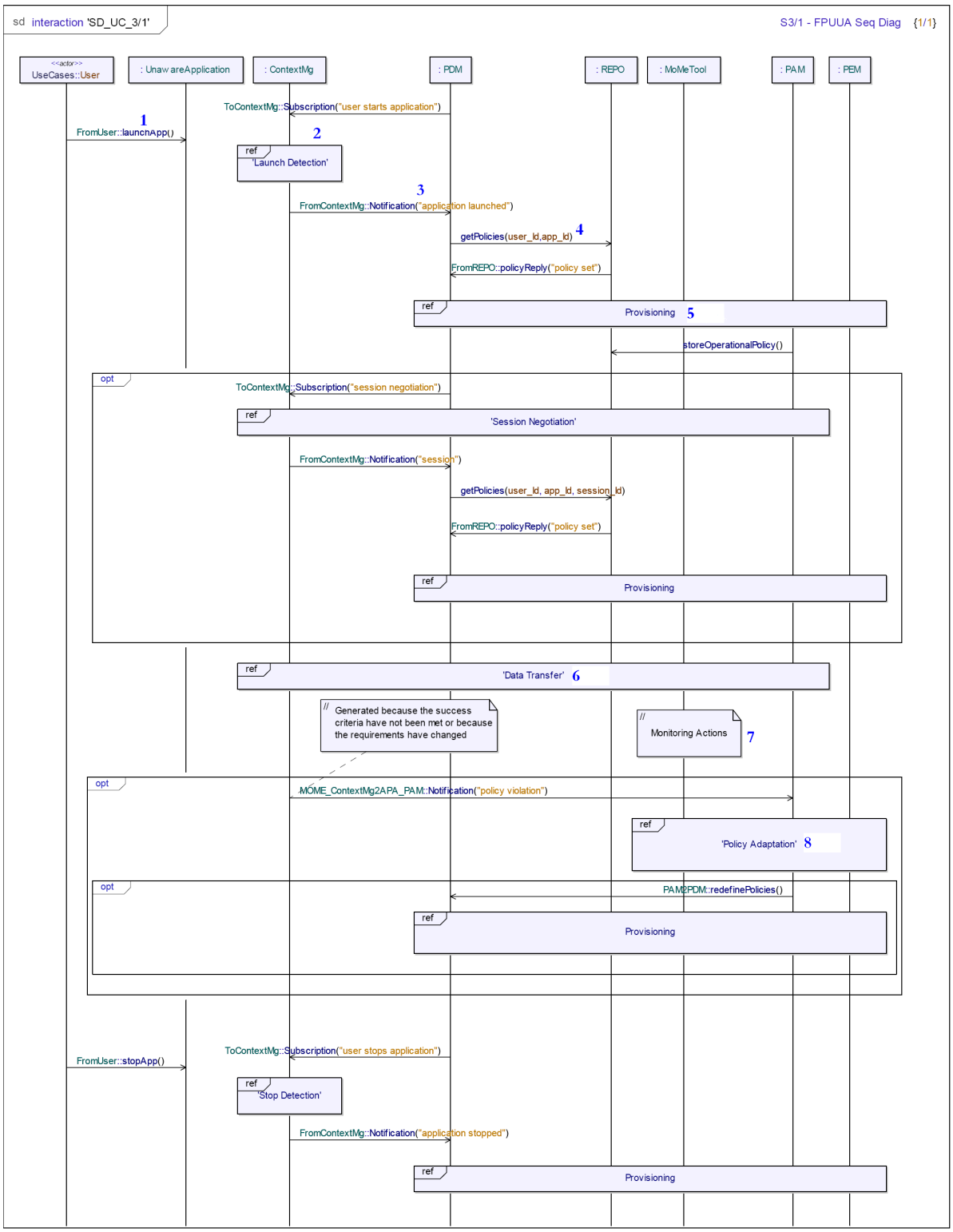


FIGURE 2.5 – *Use Case S3/1 : Satisfaire une politique utilisateur exécutant une application NETQoS-Unaware*

2.2.3.2 L'entité APA

Le rôle de l'entité APA est de réaliser le processus de provisionnement et d'adaptation de politique, c'est-à-dire l'élaboration et l'adaptation des

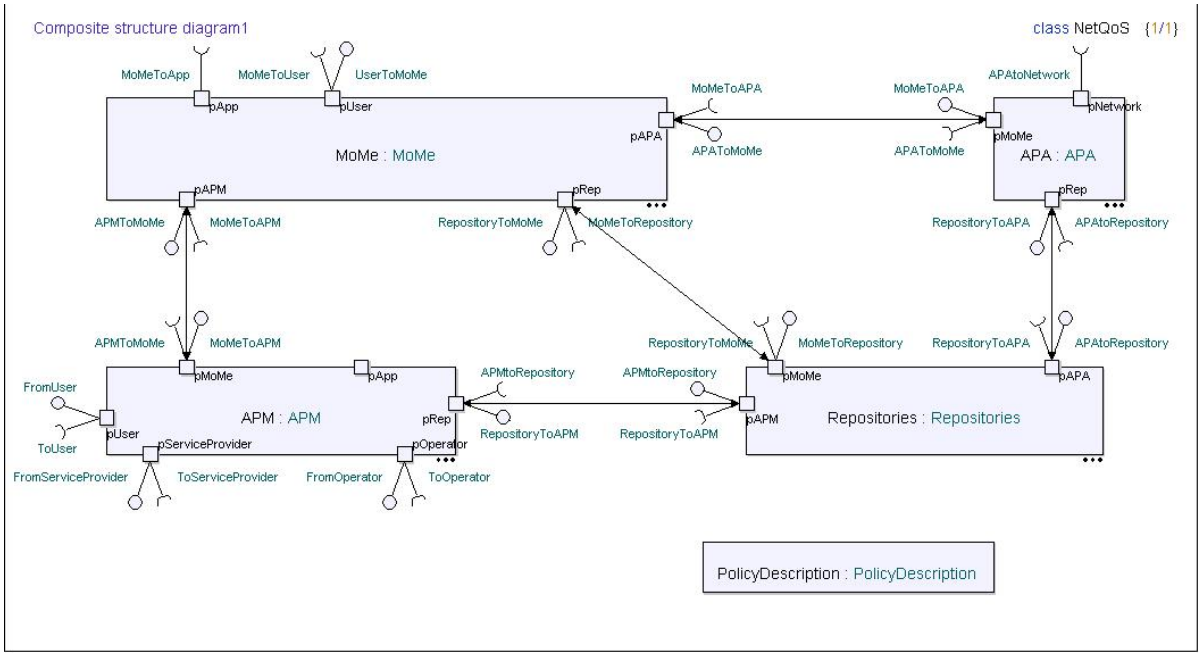


FIGURE 2.6 – *Diagramme de structure composite de NetQoS*

politiques opérationnelles à appliquer à une ou plusieurs couches du système de communication dans le but de satisfaire les politiques de niveau acteur en tenant compte de l'évolution du contexte.

2.2.3.2.1 Les fonctions de l'entité APA

Provisionnement de politique
Telle que définie dans le projet NETQoS, la notion de provisionnement (incluant le re-provisionnement) relève d'une problématique d'adaptation architecturale, c'est-à-dire liée au choix de la composition mécanismes à mettre en œuvre pour satisfaire une politique de niveau acteur. Cette phase est nécessaire au démarrage des communications, mais peut également intervenir en cours de communication si nécessaire.

En tenant compte des objectifs de politique orientée-réseau (objectifs de QoS), des informations de contexte (réseau d'accès) et des informations issues de mesures (taux de pertes observé sur un chemin donné de bout-en-bout) effectuées par l'entité de monitoring (MoMe), l'APA doit définir la politique opérationnelle qui permet de réaliser les objectifs de QoS.

Cette politique opérationnelle est composée de règles de configuration aux différents niveaux de la pile de communication, notamment de niveau Réseau et Transport. Par exemple :

– au niveau Réseau, ces règles peuvent définir le type de classes de service qui pourraient être appliquées quand des services différenciés sont disponibles ; dans ce cas, le SLA contracté avec l'acteur doit aussi être pris en compte dans le processus de décision.
– au niveau Transport, ces règles définissent le choix de composition de différents mécanismes qui seront implémentés dans le protocole de

Transport sous-tendant la communication considérée ; par exemple, la composition d'un contrôle de congestion basé débit tel que TFRC, et d'un contrôle des pertes à fiabilité partielle temporisée peut être décidée pour répondre aux besoins en QoS d'une application multimédia de type streaming vidéo.

Le provisionnement de politique est réalisé :
- quand la politique doit être activée pour la première fois, c'est-à-dire lorsqu'une nouvelle application est lancée ;
- ou lorsqu'une politique opérationnelle active doit être redéfinie, en raison de changement de conditions. Par exemple si le contexte change pendant la communication (changement de réseau d'accès, changement d'objectifs de QoS, ...), alors l'APA peut avoir à revoir les règles de configuration existantes, à un ou plusieurs niveaux d'entités gérées.

Adaptation de politique
Dans NETQoS, la notion d'adaptation de politique traite de l'aspect comportemental du processus d'adaptation. Celui-ci agit sur une politique opérationnelle active et conduit à la modification de la valeur des paramètres des services et des mécanismes sélectionnés. Cependant l'architecture de ces services et mécanismes n'est pas changée, c'est-à-dire que le choix de composition reste le même. Par exemple :
- Au niveau Réseau, le changement de paramètres d'un service basé AF est un exemple d'action d'adaptation.
- Au niveau Transport, la modification du débit d'émission d'une entité de Transport donnée est un autre exemple d'action d'adaptation.

Une adaptation de politique intervient lorsqu'une politique opérationnelle doit être adaptée en raison de changement de conditions. Par exemple, si le critère de succès qui définit la politique n'est pas respecté, alors l'APA peut essayer d'adapter un ou plusieurs niveaux d'entités gérées. Si l'adaptation comportementale s'avère insuffisante, le choix de composition des services/mécanismes sera à reconsidérer dans le cadre d'un nouveau provisionnement de politique en remplacement de celle existante.

2.2.3.2.2 Les blocs fonctionnels de l'APA

Pour implémenter les fonctions de provisionnement/adaptation, l'APA contient trois blocs fonctionnels (Figure 2.7) :

- le Policy Decision Manager (PDM) est chargé de décider des politiques opérationnelles qui permettent de prendre en compte l'ensemble des politiques de niveau acteur en fonction des informations de contexte ;
- le Policy Enforcement Manager (PEM) est chargé de faire appliquer les politiques opérationnelles élaborées par le PDM vers les points de déploiement de niveau Réseau/Transport (typiquement des entités de type ETP pour la gestion d'un protocole de Transport configurable) ;
- le Policy Adaptation Manager (PAM) est chargé de l'adaptation (comportementale) de politique, individuellement ou par sous-ensembles, quand le critère de succès associé à la politique n'est pas rempli.

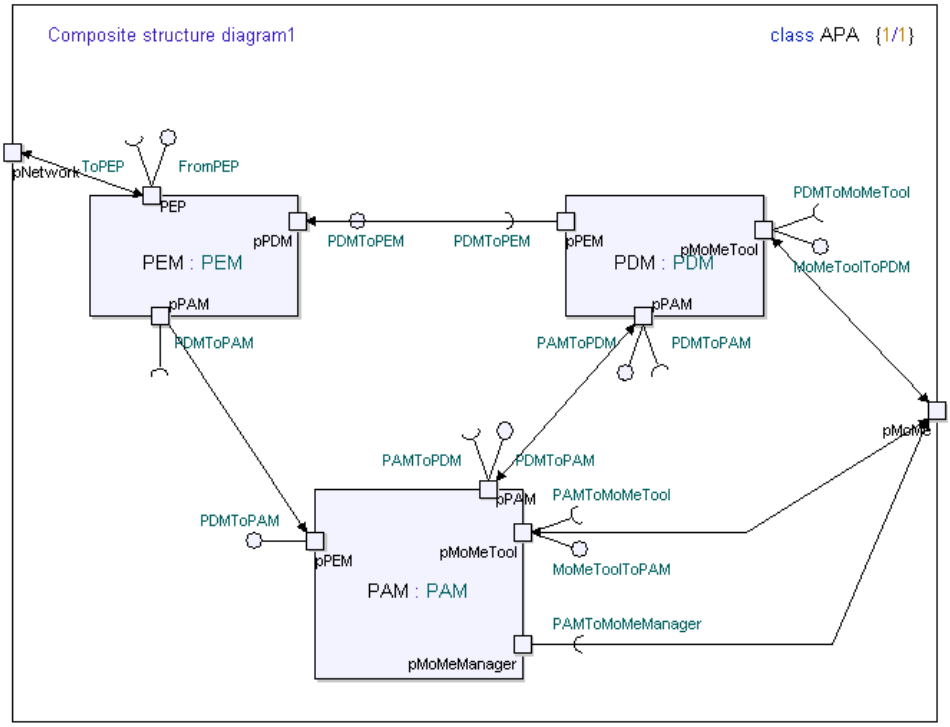


FIGURE 2.7 – *Diagramme de structure composite de l'APA*

Nous détaillons à présent ces trois composants de l'APA.

Policy Decision Manager
Partant des objectifs de QoS définis pour l'application et des préférences des acteurs, le PDM a pour rôle de décider d'un ensemble de politiques opérationnelles à mettre en œuvre aux niveaux d'adaptation considérés (par exemple, niveaux Réseau et/ou Transport). Ce raffinement de politiques peut être réalisé en utilisant des règles de mise en correspondance qui tiennent compte de la dynamicité des politiques acteurs et des informations de contexte. Ce raffinement peut conduire à des conflits de politiques (par exemple quand les politiques de niveau acteur ne peuvent pas toutes être satisfaites) que le système doit résoudre. Dans NETQoS, la gestion des conflits de politique est particulièrement étudiée au niveau Réseau, en tenant compte du SLA négocié au niveau IP. Elle sort du cadre de nos contributions.

Les niveaux de politiques opérationnelles peuvent être traités de manière séparée (gestion de la politique d'adaptation à un seul niveau) ou ensemble (gestion de la politique d'adaptation à plusieurs niveaux) :

– dans le cas d'une adaptation mono-niveau, le PDM est chargé de décider des politiques soit au niveau Réseau, soit au niveau Transport ;
– dans le cas d'une adaptation multi-niveaux, le PDM doit décider des politiques impliquant plusieurs niveaux, consécutivement ou simultanément. Dans ce cas, le PDM doit coordonner les actions à appliquer au niveau Réseau et Transport.

Le PDM est activé dans différentes situations. Lorsque le contexte a été

modifié (un acteur a changé sa politique), le PDM décide de changer une ou plusieurs de ses politiques opérationnelles. Les modifications de contexte sont signalées au PDM par le Context Manager de l'entité MoMe. De même, lorsqu'une politique est évaluée comme étant inefficace, le PDM peut décider de la modifier. Il peut aussi reconsidérer un sous-ensemble de politiques ; les informations relatives à l'inefficacité d'une politique sont fournies au PDM par le PAM. Enfin, lorsqu'une politique ne peut pas être déployée, le PDM est chargé de décider d'une nouvelle politique ; il peut aussi changer des politiques existantes ; les informations relatives à l'impossibilité de déploiement sont fournies au PDM par le PEM.

Chaque fois que le PDM décide d'une (nouvelle) politique opérationnelle, il fournit les règles correspondantes au PEM. Si la politique est déployable, le PDM informe le PAM du déploiement de la nouvelle politique.

Policy Enforcement Manager
Le PEM est chargé de faire appliquer les politiques opérationnelles décidées par le PDM vers les entités gérées. Par exemple :

Pour une politique de niveau Réseau, le PEM distribue les règles à mettre en œuvre par les équipements réseau. Ces équipements dépendent du type de réseau sous-jacent et peuvent être par exemple des routeurs de bordure, des Bandwidth Broker, ou encore outils de gestion de réseau). Le PEM doit transmettre au PDM un message de succès ou d'échec sur la politique à déployer (par exemple, un réseau basée sur un principe de réservation de ressources peut ne pas pouvoir appliquer la politique requise en raison d'une pénurie de ressources) ;

Pour une politique de niveau Transport, le PEM distribue les règles de configuration du protocole de Transport à appliquer sur les hôtes d'extrémités. Par exemple, le déploiement d'une politique peut consister à donner des règles de composition des mécanismes Transport ; dans ce cas, tous les mécanismes sont sensés être disponibles sur les hôtes concernés ou récupérables par ceux-ci.

Le PEM se veut indépendant des technologies Réseau et Transport qui sont utilisées pour mettre en œuvre les politiques ; autrement dit, le PEM fournit les règles à appliquer dans un langage indépendant de celui utilisé par le système de communication géré.

En conséquence, le PEM fournit des interfaces génériques permettant la communication avec les entités gérées, et des adaptateurs (appelés Agent dans NETQoS) doivent être implémentés, pour traduire les règles génériques du PEM en des règles spécifiques à une technologie particulière. La section 2.3.4 de ce chapitre illustre comment nous avons instancié le concept d'agent pour l'adaptation au niveau Transport.

Le PEM est activé dans différentes situations. Lorsqu'une (nouvelle) politique doit être appliquée, le PEM doit fournir les règles données par le PDM aux composants du système de communication. Lorsque les règles de politique ne peuvent pas être appliquées (par exemple, si une réservation de ressources ne peut pas être faite), le PEM doit informer les PDM

que sa politique ne peut pas être activée. L'impossibilité de déploiement est notifiée au PEM par le système de communication.

Policy Adaptation Manager
Le PAM est chargé de l'adaptation comportementale d'une politique lorsque son critère de succès n'est pas satisfait. Grâce à des alarmes envoyées par le MoMe, signalant la violation d'un critère de succès, le PAM décide des paramètres à modifier dans la politique opérationnelle existante. Quand l'adaptation souhaitée n'est pas possible, le PDM est informé de l'échec, en vue d'une re-provisionnement de politique. Si l'adaptation est possible, le PAM la fait appliquer par l'intermédiaire du PEM.

Le PAM est activé dans différentes situations. Lorsque le PAM est informé par le PDM qu'une nouvelle politique est mise en œuvre, le PAM stocke la politique dans le répertoire de politiques opérationnelles avec des critères de succès (par exemple, le taux de pertes de bout-en-bout doit être inférieur à 5%) et souscrire aux tâches de monitoring associées, réalisées par le MoMe. Lorsque le PAM reçoit une alarme du MoMe, il essaye d'adapter la politique défectueuse.

2.3 Spécification détaillée des composants de l'entité APA

Cette section présente la conception fonctionnelle détaillée des composants de l'APA.

La section 2.3.1 présente les choix remarquables de cette conception et deux cas d'utilisation sont présentés pour illustrer les interactions entre les principaux composants de l'APA.

Les sections 2.3.2, 2.3.3 et 2.3.4 détaillent ensuite la structure d'implémentation des différents composants de l'APA. Les interactions entre ces composants, leurs sous-composants et les autres composants du système NETQoS, sont illustrés à travers les deux cas d'utilisation présentés dans la précédente section.

2.3.1 Choix remarquables de conception

2.3.1.1 APA multi-niveaux ou mono-niveau

La structure de l'APA a été conçue de manière à pouvoir fonctionner à un seul niveau du système de communication (par exemple au niveau Transport ou au niveau Réseau) ou à plusieurs niveaux (par exemple au niveau Réseau et au niveau Transport). L'objectif est de permettre l'implémentation de l'APA en considérant des points de vue indépendants (point de vue de l'opérateur et point de vue de l'utilisateur), tout en gardant la possibilité de considérer un point de vue plus global, en combinant plusieurs niveaux d'adaptation ensembles.

Quand plusieurs niveaux sont impliqués dans le processus d'adaptation/provisionnement, deux approches ont été considérées pour NET-QoS :

– dans la première approche, le processus est réalisé niveau par niveau :
  • pour le provisionnement (fonction du PDM), la décision prise à un niveau donné i dépend de celle prise au niveau i-1 ; par exemple, au-dessus d'un service IP orienté DiffServ, le protocole de Transport à composer dépendra des performances fournies par la classe de service IP sélectionnée.
  • pour l'adaptation, on essaye d'adapter d'abord au niveau le plus haut, puis aux niveaux sous-jacents par ordre décroissant, si l'adaptation n'est pas possible. A chaque niveau, on essaye d'adapter d'abord les paramètres des entités gérées (grâce au PAM) ; si ce n'est pas possible, une adaptation architecturale est ensuite considérée (via le PDM) ;

– dans la seconde approche, qui sort du cadre des travaux présentés dans ce mémoire, le processus de provisionnement et d'adaptation est réalisé en considérant plusieurs niveaux ensembles. L'objectif est de trouver la meilleure configuration qui, à l'échelle de plusieurs niveaux du système de communication, permet de satisfaire les besoins de la politique.

2.3.1.2 Indépendance de l'APA vis-à-vis des moteurs de décision

La structure interne de l'APA a été conçue de façon à permettre l'intégration de différents types de moteur de décision pour le provisionnement ou l'adaptation. Par exemple, ces moteurs peuvent être implémentés avec des règles basées sur des canevas (au sens template) ou basées sur des modèles (modèle d'optimisation par exemple) ; ils peuvent être appliqués par niveau de façon indépendante, ou en combinant plusieurs niveaux ensembles, d'une manière cordonnée. De façon analogue, l'APA se veut indépendant du type d'entité gérée, et doit par exemple pouvoir agir sur des entités de niveau Transport ou des entités de niveau Réseau ou les deux.

Pour garder cette indépendance, PDM, PAM et PEM utilisent des composants que nous avons appelés Decisioner, Adapter et Enforcer, qui peuvent être implémentés librement (Figure 2.8). La seule obligation pour ces composants est d'implémenter les interfaces JAVA suivantes :

– IPolicyDecisioner est l'interface qui doit être implémentée par les Decisioner ; elle est composée d'une seule méthode, takeDecision(OpPolicy) :OpPolicy, qui retourne la politique opérationnelle à appliquer au niveau i considéré. En entrée, elle prend la politique opérationnelle qui a été définie au niveau i-1 ;
– IPolicyAdapter est l'interface qui doit être implémentée par les Adapter ; elle est composée d'une seule méthode adaptPolicy(OpPolicy) :boolean, qui retourne un booléen selon que la politique opérationnelle appliquée au niveau considéré peut être adaptée ou non.
– IPolicyEnforcer est l'interface qui doit être implémentée par les Enfor-

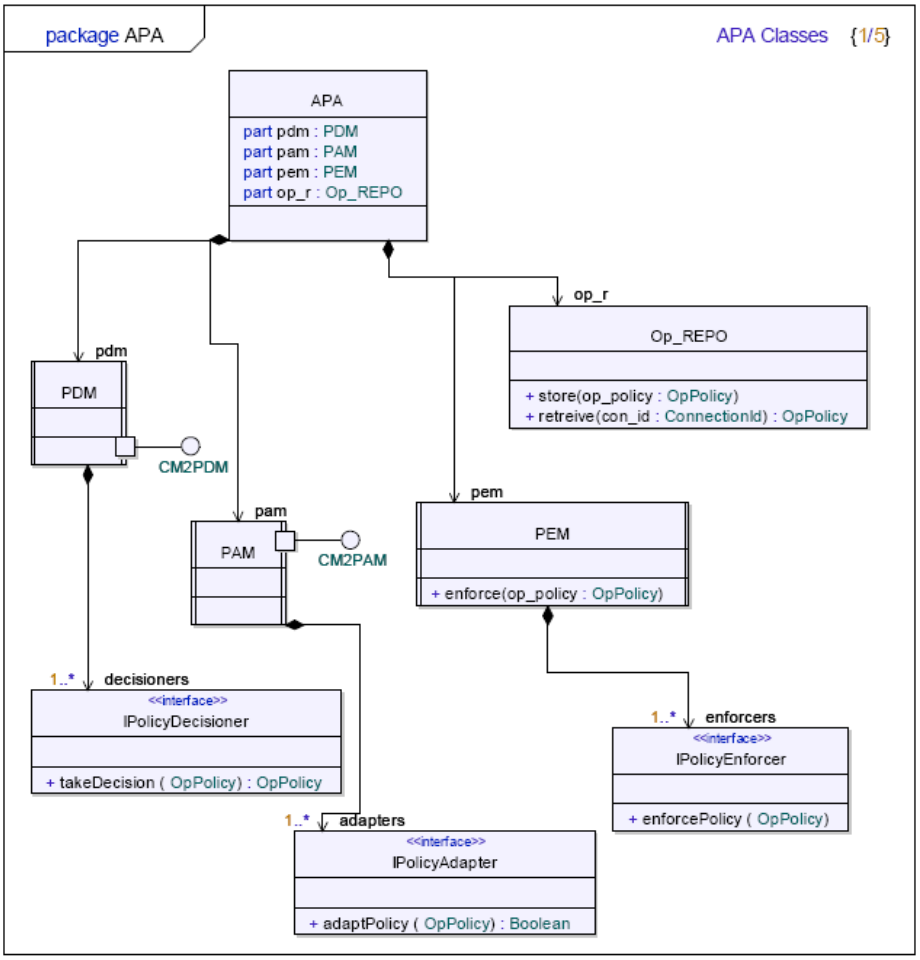


FIGURE 2.8 – *Diagramme de classe de l'APA*

cer ; elle est composée d'une seule méthode enforcePolicy(OpPolicy).

Ainsi, quel que soit le type de moteurs pour le provisionnement, l'adaptation ou la distribution, les codes du PDM, du PAM et du PEM ne changent pas puisqu'ils utilisent toujours les mêmes méthodes pour invoquer les fonctionnalités des Decisioner, Adapter et Enforcer.

2.3.1.3 Autres choix remarquables

La figure 2.8 montre une autre classe interne APA : Op_REPO qui est un répertoire local où sont stockées les politiques opérationnelles. Il est utilisé pour stocker les décisions du PDM/PAM et pour les récupérer quand une adaptation est requise.

La figure 2.9 montre les autres classes utilisées dans la conception détaillée de l'APA :

- la classe LaunchEvent identifie un événement de lancement d'une application ;
- la classe PolViolationEvent identifie une violation de politique pour une application ;

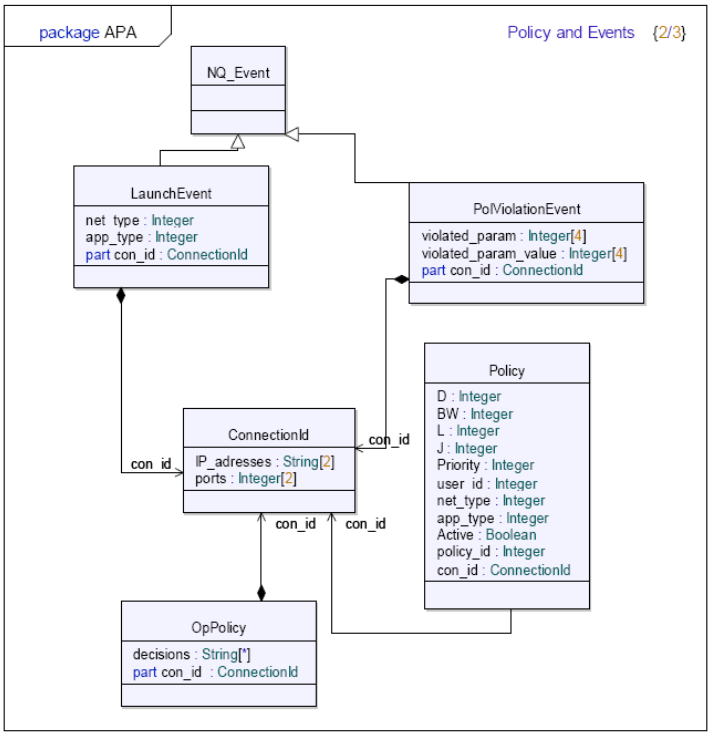


FIGURE 2.9 – *Autre classes du package APA*

– la classe (intermediate) Policy inclut plusieurs paramètres, notamment les paramètres de QoS (débit, délai, taux de pertes, gigue) requis pour une connexion ;
– la classe OpPolicy inclut les décisions d'adaptation/provisionnement du PAM/PDM associées à une connexion identifiée.

2.3.1.4 Scénarios d'application de l'entité APA

Cette section détaille deux cas d'utilisation de l'APA liés 1) au lancement d'une application et 2) à la violation de la politique requise pour une application.

Use Case 1 : Lancement d'une application

Ce Use Case illustre la réponse des composants de l'APA après une notification de lancement d'une nouvelle application. Il inclut deux acteurs : le gestionnaire de contexte (CM), qui notifie cet évènement à l'APA via le PDM, et les agents (à un ou plusieurs niveaux) qui sont notifiés par l'APA (via le PEM) des politiques opérationnelles à déployer par les entités gérées. Deux niveaux consécutifs de provisionnement sont considérés (Réseau puis Transport). La Figure 2.10 fournit une vue de haut niveau du diagramme de séquence impliquant les composants de l'APA et les autres composants du système.

Lorsque le CM notifie le PDM d'un lancement d'une application, ce dernier récupère auprès du répertoire (REPO) les politiques associées à l'évè-

nement. Une fois les politiques récupérées, le PDM invoque les décideurs (decisioner) de niveau Réseau et/ou Transport, récupère auprès d'eux les politiques opérationnelles à mettre en œuvre et les transmet au PEM pour qu'elles soient appliquées. Les politiques sont également stockées dans le répertoire de stockage des politiques opérationnelles.

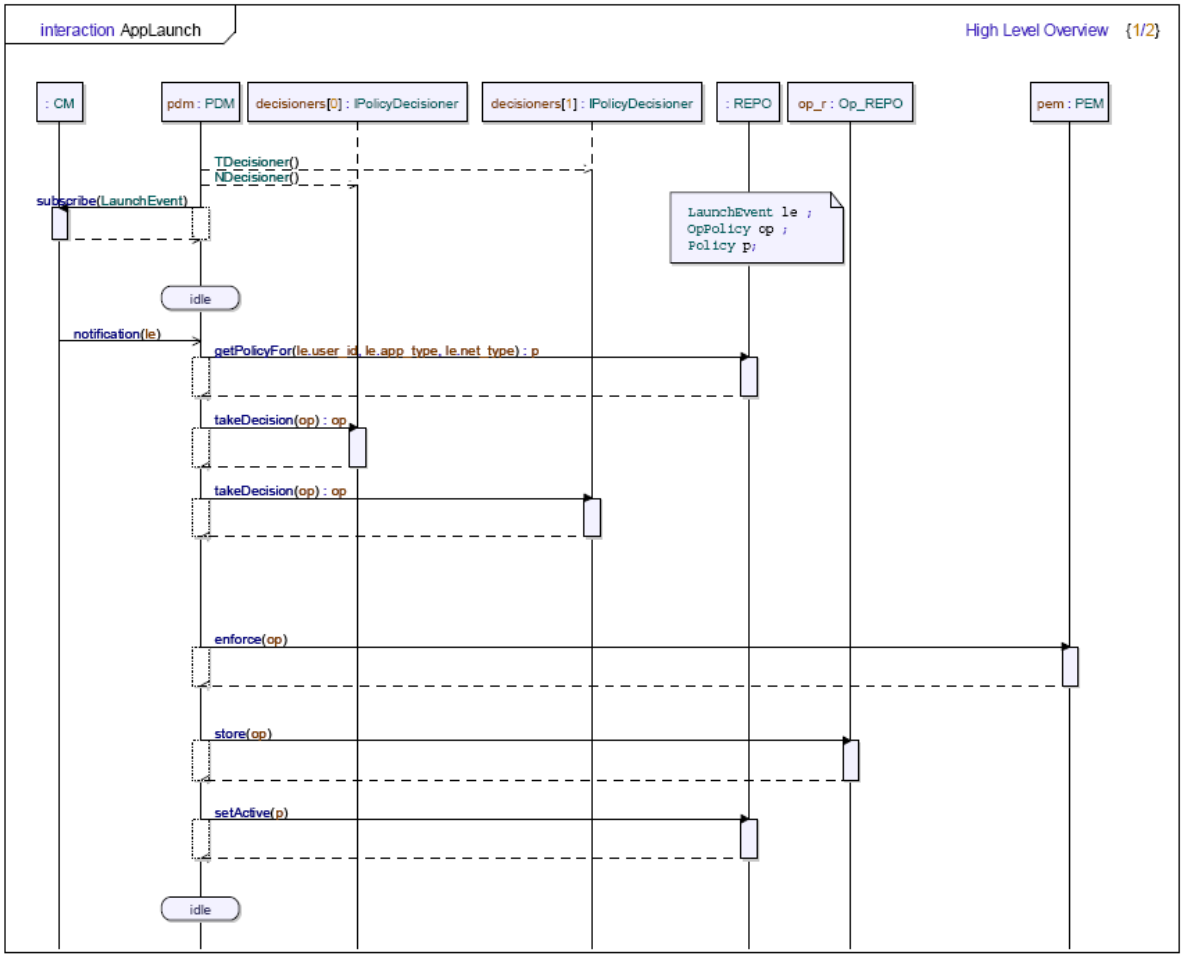


FIGURE 2.10 – *Diagramme de séquence d'un provisionnement au niveau Réseau/Transport (UC1)*

Use Case 2 : Violation de politique

Nous étudions ici la réponse des composants de l'APA après une notification signalant que les objectifs de politique (par exemple les objectifs de QoS pour une application donnée), qui sont supposés être satisfaits pour une politique opérationnelle donnée, ne le sont pas. Il inclut deux acteurs : le gestionnaire de contexte (CM), qui notifie cet évènement à l'APA via le PAM, et les agents (à un ou plusieurs niveaux) qui sont notifiés par l'APA des règles d'adaptation à appliquer par les entités gérées. Deux niveaux consécutifs d'adaptation sont ici considérés (Transport puis Réseau).

La Figure 2.11 fournit une vue haut niveau du diagramme de séquence impliquant les composants de l'APA et les autres composants du système. Lorsque le CM notifie le PAM de la violation de politique, celui-ci récupère auprès du répertoire REPO la politique associée à l'évènement. Il récupère

également les paramètres de la politique opérationnelle. Une fois les politiques récupérées, le PAM invoque l'adaptateur de politique (l'Adapter) de niveau Transport afin de modifier la politique actuelle et prendre en compte le changement de contexte. Si une adaptation est possible, l'Adapter retourne une nouvelle politique opérationnelle au PAM qui la transmet au PEM pour qu'elle soit appliquée. La nouvelle politique est stockée dans le répertoire de stockage des politiques opérationnelles. Si l'Adapter ne peut pas prendre de décision d'adaptation, le PAM informe le PDM de cet état de fait afin que celui-ci prenne une décision de re-provisonnement.

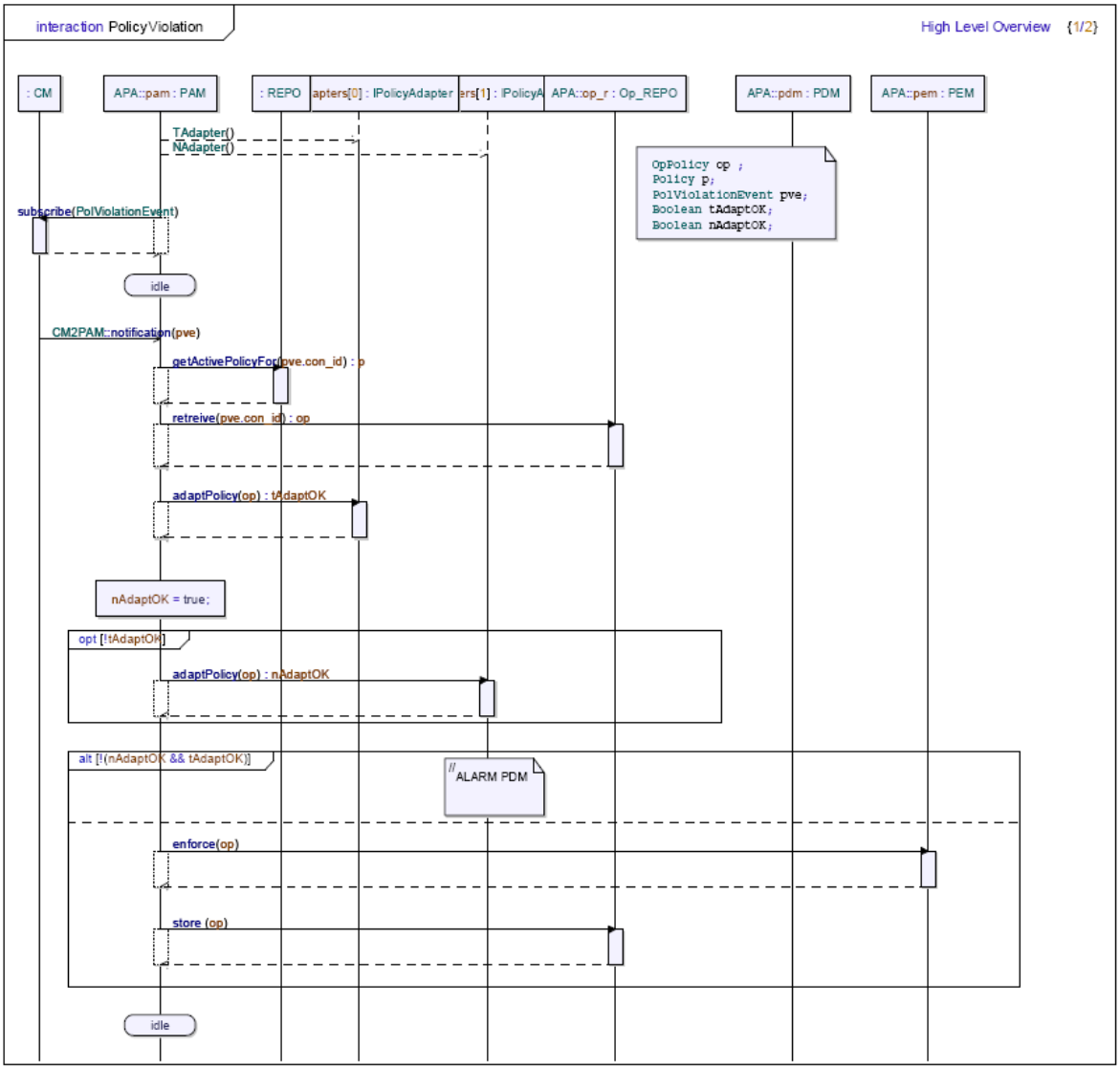


FIGURE 2.11 – *Diagramme de séquence d'une adaptation au niveau Réseau/Transport (UC2)*

2.3.2 Spécification détaillée du PDM

Le PDM est composé de trois entités créées au lancement du système (Figure 2.12).

La première entité, le PDMNotificationManager, traite la notification des

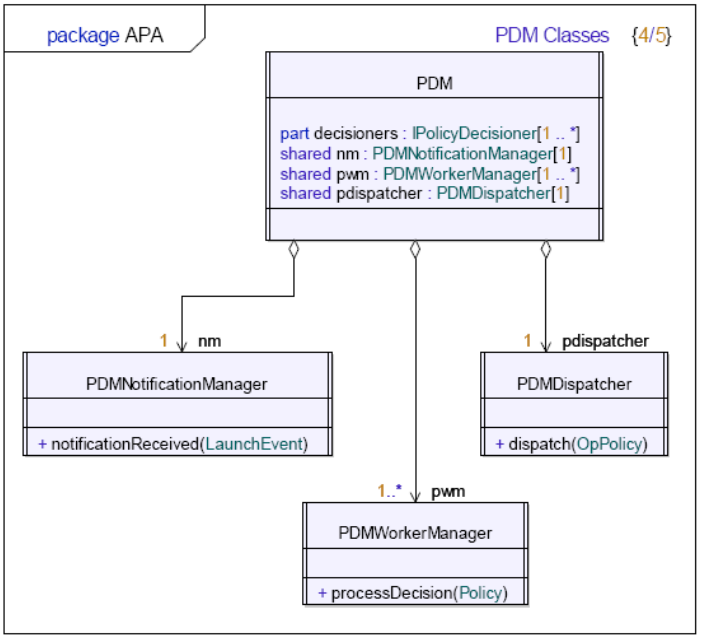


FIGURE 2.12 – *Diagramme de classe du PDM*

évènements auxquels il a souscrit (par exemple, "lancement d'une nouvelle application") venant du CM. Quand une notification survient, il récupère la (les) politiques correspondantes auprès du REPO et la (les) stocke dans une base de données locale. Puis, il invoque la méthode processDecision(Policy) du PDMWorkerManager.

Dérivée de la première entité, le PDMWorkerManager prend la décision de provisionnement nécessaire à un ou plusieurs niveaux (avec l'aide des PolicyDecisioners adéquats). Lorsqu'on lui demande de prendre une décision, le PDMWorkerManager invoque la méthode takeDecision(OpPolicy) : OpPolicy du Decisioner adéquat, soit à un seul niveau ou à plusieurs niveaux successivement du plus bas au plus haut. Pour réaliser cette tâche, le PDMWorkerManager utilise et coordonne un ou plusieurs PDMWorkers. Une fois la politique opérationnelle récupérée, le PDMWorker invoque la méthode dispatch(OpPolicy) du PDMDispatcher.

Enfin, le PDMDispacher traite les interactions avec le PEM pour faire appliquer la politique et la stocker dans le répertoire local (Op_REPO). Lorsqu'il est invoqué par le PDMWorker, le PDMDispacher invoque la méthode enforcePolicy(OpPolicy) du PEM, puis il stocke la politique opérationnelle dans le répertoire local(Op_REPO).

2.3.3 Spécification détaillée du PAM

De façon analogue au PDM, le PAM est composé de trois entités créées au lancement du système (Figure 2.13) :

Le PAMNotificationManager traite la notification des évènements auxquels il a souscrit (par exemple, "violation d'une politique") venant du gestionnaire de contexte (CM). Quand une notification survient, il récupère la (les) politiques intermédiaires et opérationnelles correspondantes

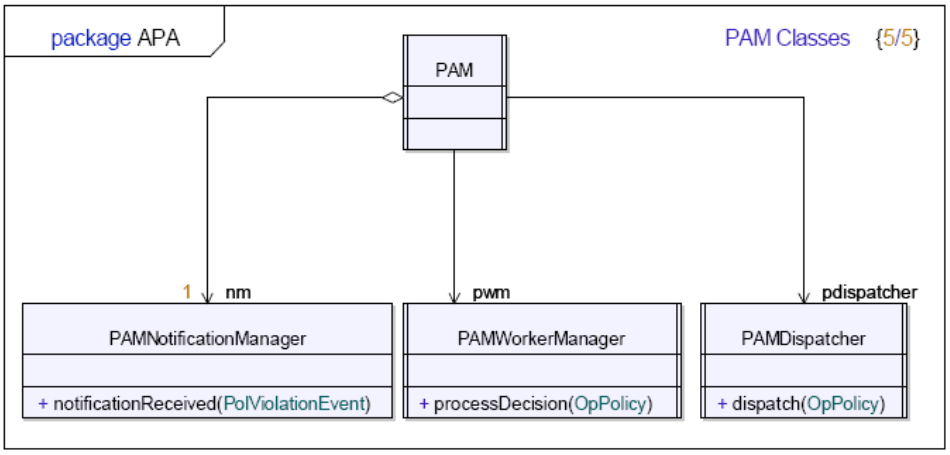


FIGURE 2.13 – *Diagramme de classe du PAM*

et la (les) stocke dans une base de données locale. Puis, il invoque la méthode processDecision(Policy) du PAMWorkerManager.

Le PAMWorkerManager prend la décision de l'adaptation nécessaire à un ou plusieurs niveaux (avec l'aide des PolicyAdapters adéquats). Lorsqu'on lui demande d'opérer une adaptation, le PAMWorkerManager invoque la méthode tadaptPolicy(OpPolicy) : boolean de l'Adapter adéquat, soit à un seul niveau ou à plusieurs niveaux successivement du plus bas au plus haut. Comme pour le PDMWorkerManager, pour réaliser cette tâche, le PAMWorkerManager utilise et coordonne un ou plusieurs PAMWorkers. Si le résultat de l'adaptation est positif, le PAMWorkerManager invoque la méthode dispatch(OpPolicy) du PAMDispatcher. Si l'adaptation n'est pas possible, le PDM est informé par une alarme qu'un re-provisonnement doit être fait.

Enfin, le PDMDispacher traite les interactions avec le(s) PEM(s) pour faire appliquer la politique adaptée. Lorsqu'il est invoqué par le PAMWorker, le PAMDispacher invoque la méthode enforce(OpPolicy) du PEM, puis il stocke la politique opérationnelle dans le répertoire local(Op_REPO).

2.3.4 Spécification détaillée du PEM

Le diagramme de classe du PEM est donné figure 2.14.

Le PEMEnforcementManager est un composant interne du PEM qui, en fonction des différents niveaux de politique inclus dans la politique à déployer, délègue la mise en œuvre de la politique pour chacun de ces niveaux correspondants à une instance de IPolicyEnforcer.

Par exemple, une politique opérationnelle contenant des informations à mettre en œuvre à la fois au niveau de la couche Transport et au niveau de la couche Réseau, utilisera une instance de IPolicyEnforcer specialisée dans la mise en œuvre au niveau Transport (TEnforcer) et au niveau Réseau (NEnforcer).

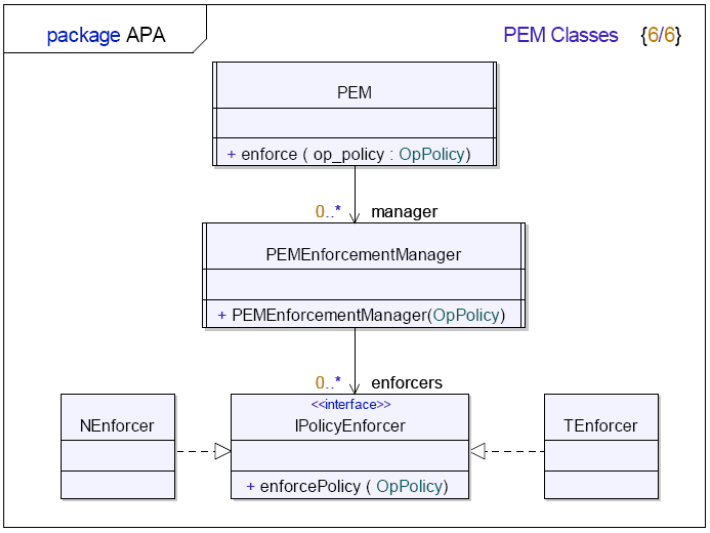


FIGURE 2.14 – *Diagramme de classe des composants internes PEM*

La notion d'Agent

Un des choix remarquables du système NETQoS est de fournir un moyen de gérer la QoS à plusieurs niveaux du système de communication (Transport, Réseau, Liaison de données...). Pourtant, l'existence de différents systèmes fournissant de la QoS à chacune des couches considérées est un élément qui contribue à sa complexité.

Afin d'utiliser au mieux ces différentes technologies tout en maintenant un certain niveau d'abstraction dans le composant APA, les politiques de QoS sont décrites dans un langage propre au système NETQoS (NetQoS policy). Les politiques opérationnelles qui sont issues des décisions prises par l'APA et qui sont acheminées par le PEM vers les entités gérées, sont ainsi décrites dans un langage indépendant de la technologie. Figure 2.15.

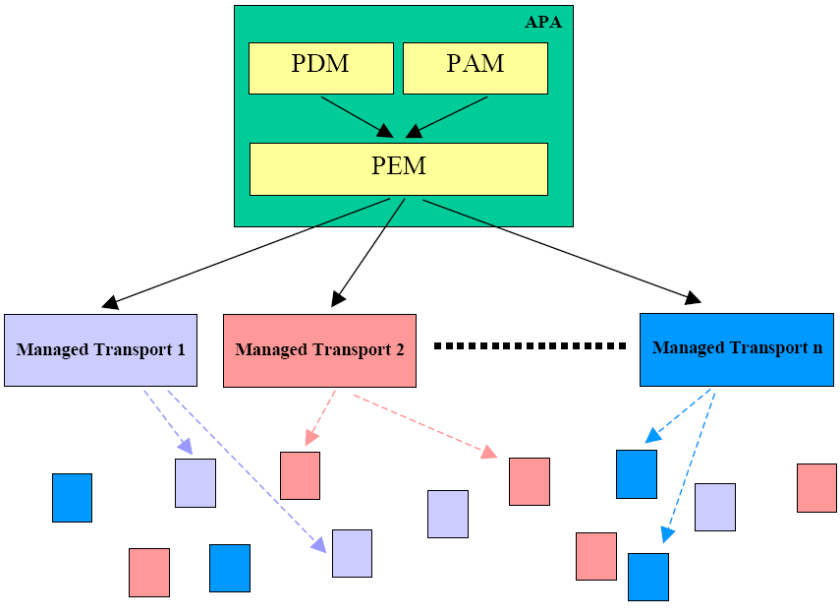


FIGURE 2.15 – *Problème des multiples entités gérées à un niveau donné*

Afin de pouvoir prendre en compte une grande variété de systèmes existants, le principe de l'adaptateur a été appliqué. Ce concept, dénommé Agent par la suite, est ici illustré pour le niveau Transport.

Au niveau Transport, l'agent ETP (ou ETP Agent dans les diagrammes suivants) représente l'instance d'un tel composant adaptateur au niveau Transport.

Les principales fonctions de l'ETP Agent sont d'une part de garder en mémoire la liste des entités ETP existantes pour être capable de réaliser les actions de provisionnement et d'adaptation lorsqu'elles sont requises par l'APA ; d'autre part, l'ETP Agent est responsable de la transformation de la politique opérationnelle de bas niveau en une description compréhensible pour la technologie sous-jacente (en l'occurrence ici ETP).

2.3.4.1 Cas d'utilisation de l'ETP Agent

Les principales fonctions de l'agent l'ETP nécessitent que des entités ETP s'exécutent sur les hôtes d'extrémités et que l'APA soit lancé. Le PEM est l'acteur principal des scénarios d'application. De plus, pour réaliser ces actions sur les entités ETP, l'ETP Agent doit les garder en mémoire. Cette interaction implique seulement l'entité ETP. Les principaux cas d'utilisation de l'ETP Agent : EnforceDecision, ChangeParameter et Register sont représentés sur la Figure 2.16.

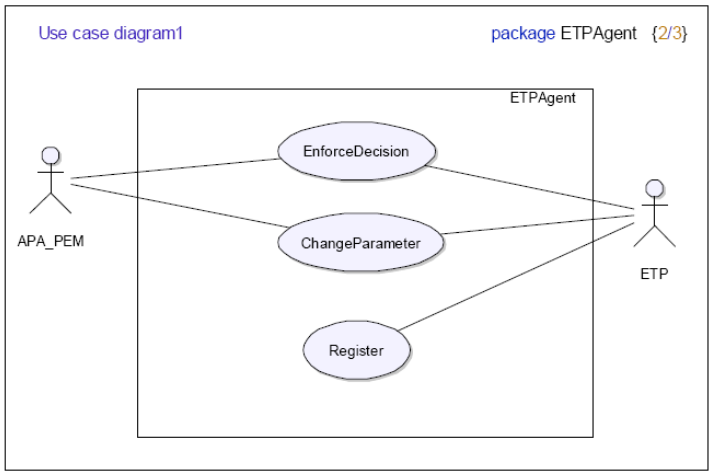


FIGURE 2.16 – *Interactions de l'ETP Agent avec l'APA.PEM et les entités ETP*

2.3.4.2 Structure interne de l'ETP Agent

L'ETP Agent présente des interfaces d'une part avec le PEM, qui doit contacter l'ETP Agent pour réaliser les actions d'adaptation et de déploiement, et d'autre part avec les entités ETP qui réalisent ces actions. Figure 2.17.

Pour chaque requête entrante en provenance du PEM, l'ETP Agent crée une nouvelle instance d'un objet ConnectionHandler qui sera chargé de réaliser les actions requises en fonction du type de message reçu. Ceci per-

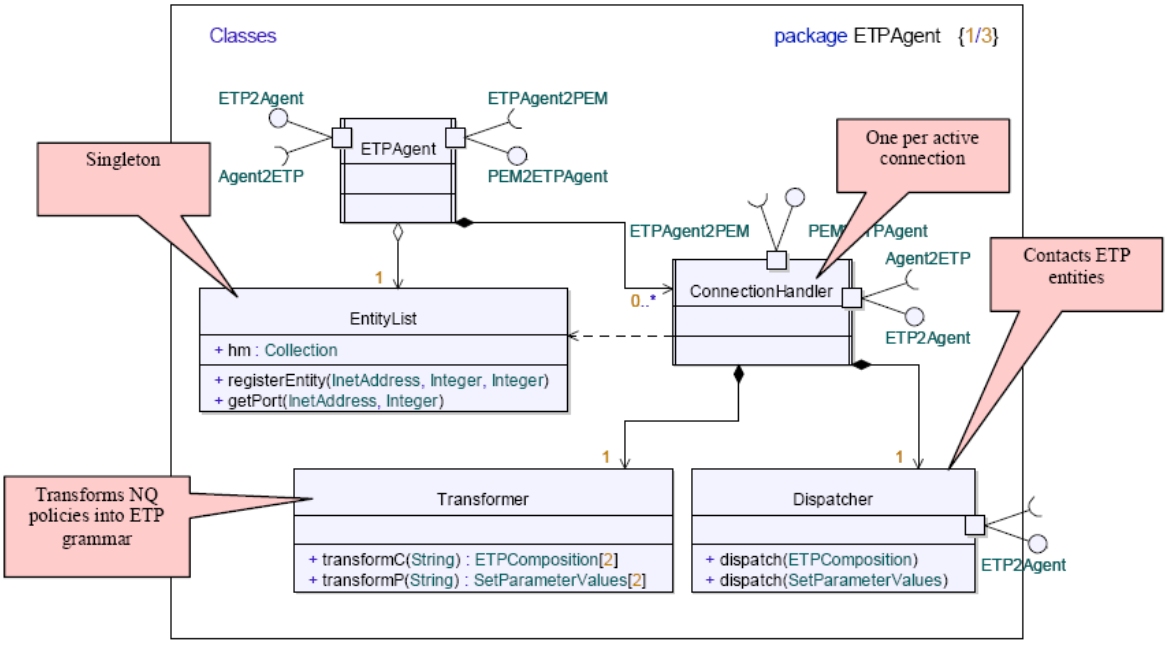


FIGURE 2.17 – *Diagramme de classe de l'ETP agent*

met une gestion multithreadée des actions de déploiement afin que l'échec d'une entité ETP ne retarde pas le traitement d'autres requêtes entrantes. Pour une requête du PEM reçue, les interactions entre les différents objets sont décrits par le diagramme de structure composite de l'ETP Agent (Figure 2.18).

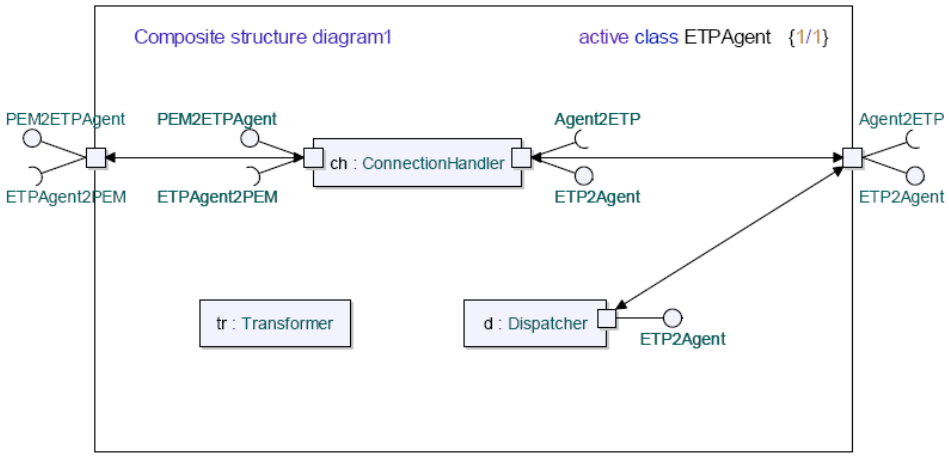


FIGURE 2.18 – *Diagramme de structure composite de l'ETP Agent*

Pour effectuer les actions EnforceDecision et ChangeParameter, le ConnectionHandler de l'ETP Agent utilise deux objets : un Transformer et un Dispatcher. Le Transformer a pour rôle de convertir la politique NETQoS décidée par l'APA, exprimée dans un langage indépendant de la technologie, dans un langage de description d'une composition ETP. De plus, une fois que la conversion a été réalisée, l'objet Dispatcher a pour rôle de contacter et de configurer les entités ETP qui ont besoin d'être adaptées.

2.3.4.3 Le micro-protocole "ETP Agent Interface" dans le plan de Management

Afin de permettre à une instance ETP s'exécutant sur un hôte d'être dynamiquement reconfigurée par le système NETQoS, l'ETP Agent communique avec un module du plan de gestion de l'instance ETP.

Ce module, appelé ETP Agent Interface, est implémenté sous la forme d'un module de traitement (processing module). Pourtant, son comportement diffère des *processing modules* classiques d'ETP. En effet, il n'est pas destiné à manipuler des données appartenant à une communication. Le rôle du processing module ETP Agent Interface est de fournir un moyen à l'ETP Agent d'interagir avec une instance ETP en cours d'exécution.

L'ETP Agent Interface reçoit ainsi des messages venant de l'ETP Agent qui sont utilisés pour reconfigurer la pile de communication active (adaptation architecturale) et pour dynamiquement changer la valeur d'une variable partagée d'une instance ETP (adaptation comportementale).

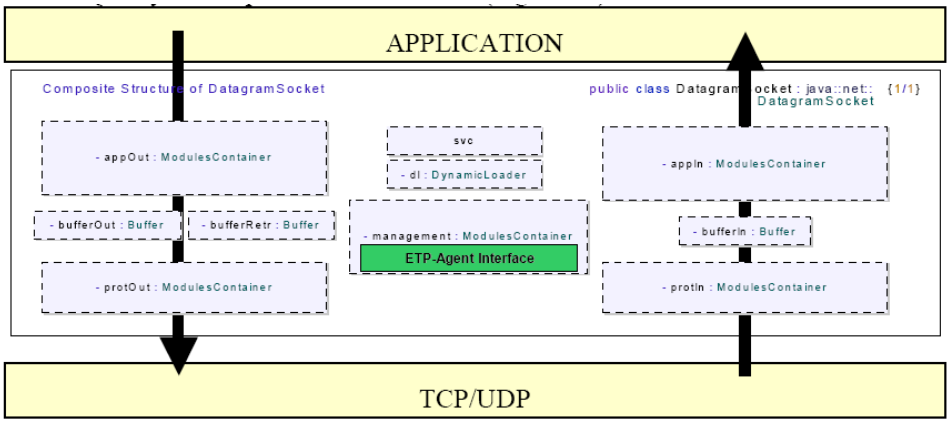


FIGURE 2.19 – *Stack ETP Agent pour une adaptation à distance*

L'ETP Agent Interface est localisé dans le plan de Management des instances ETP s'exécutant sur les hôtes d'extrémités. La pile ETP permettant de dynamiquement configurer une instance ETP interagissant avec le système NETQoS est présenté Figure 2.19.

CONCLUSION DU CHAPITRE

Ce chapitre a présenté nos contributions au besoin en architecture de contrôle pour la gestion adaptative de la QoS basée politique, incluant en particulier son interfaçage avec le protocole de Transport configurable ETP utilisé dans nos travaux. Ces contributions, réalisées dans le cadre du projet européen NETQoS, ciblent une adaptation plus large que celle du niveau Transport, le projet NETQoS ayant pour objectif d'automatiser la gestion de plusieurs niveaux d'adaptation potentiels (typiquement Transport et Réseau), individuellement ou simultanément.

La première partie du chapitre a présenté les acteurs du système NETQoS, et a précisé les besoins des acteurs "utilisateur" et "application" qui sont visés par les solutions d'adaptation de niveau Transport. La deuxième partie a décrit le modèle d'architecture conceptuelle définie pour NETQoS et

auquel nous avons directement contribué. Les différentes fonctionnalités et les composants de cette architecture permettant de décider et de faire appliquer des actions d'adaptation au niveau Transport ont ainsi été présentés au travers de plusieurs cas d'utilisation. Enfin, la dernière partie a présenté plus précisément nos choix de conception détaillés pour les composants nous concernant directement, l'APA et l'ETP Agent.

Rappelons que les objectifs du projet NETQoS étaient de contribuer au développement d'un modèle d'architecture conceptuelle, indépendante d'un environnement réseau spécifique. Cet objectif de généricité vis-à-vis de l'environnement réseau, qui nous a conduit à l'adoption d'une démarche de conception top-down, nous a ainsi amenés à proposer des modèles tant au niveau de l'architecture générale du système que des composants nous concernant directement, qui seraient certainement revus dans une approche guidée par les caractéristiques d'un environnement réseau spécifique. Bien que nous n'ayons pas suivi cette approche, les principes de conception du système NETQoS nous semblent cependant susceptibles d'être directement applicables dans la définition d'un modèle adapté à un contexte de réseau fermé dont les nœuds seraient en nombre limité. Cette intuition, intéressante pour le contexte des opérations d'intervention d'urgence (OIU), n'a cependant pas été explorée dans nos travaux.

En effet, la perspective que nous avons choisie concerne le développement de modèles et d'algorithmes pour le choix des décisions d'adaptation, et l'élaboration des actions d'adaptation impactées par ces modèles. Cette perspective, très peu explorée dans le cadre du projet NETQoS, est en effet nécessaire à la mise en œuvre d'un système tel que NETQoS, qui en l'état des travaux présentés dans ce chapitre ne définit que le cadre architectural pour guider les décisions et leur application.

Le chapitre 3 ci-après présente nos contributions, réalisées hors du projet NETQoS, dans cette perspective.

Gestion coopérative de la QoS au niveau Transport

3

Sommaire

3.1 Répartition de la bande passante 67
3.1.1 Contexte applicatif et exemple : les OIU 67
3.1.2 Description des Opérations d'Intervention d'Urgence (OIU) 68
3.1.3 Problématique et approche 70
3.1.4 Formalisation du problème 73
3.1.5 Calcul de la répartition de bande passante 74
3.1.6 Problème P_1 : minimiser la bande passante globale cédée . 75
3.1.7 Ajout d'un deuxième critère et définition du problème P_2 : minimiser l'excédent de bande passante libérée 80
3.1.8 Conclusion sur l'utilisation des critères de minimisation globale (z_1) et de minimisation des excédents (z_2) 82
3.2 Gestion coopérative de la QoS dans le système NETQoS 82
3.2.1 Identification des connexions 82
3.2.2 Classe MoMe 83
3.2.3 Mise en œuvre du processus de coopération 84
3.2.4 Diagramme de séquence pour la coopération 84
3.3 Outils et protocoles pour la récupération des informations 86
3.3.1 SNMP 86
3.3.2 Traceroute 87
3.3.3 Protocoles de routage 88
3.3.4 Fonctions de Monitoring dans ETP 88
Conclusion 89

Dans le chapitre précédent, nous avons présenté nos contributions en termes d'architecture pour la gestion autonome de la QoS à plusieurs niveaux, en particulier au niveau Transport.

Dans ce chapitre, nous proposons une approche de gestion coopérative de la QoS au niveau Transport. Les mécanismes classiques de gestion de la QoS (contrôle d'erreur, contrôle de congestion) ou les mécanismes de gestion de la QoS orientés multimédias (contrôle d'erreur avec fiabilité partielle, contrôle de congestion tenant compte des contraintes de temps,...)

opèrent connexion par connexion et visent une répartition équitable de la QoS entre les différentes connexions. Cependant, il est envisageable dans certains environnements applicatifs de remettre en cause ce principe d'équité et de considérer une gestion coopérative de la QoS par groupe de connexions. Ceci est le cas notamment pour des applications où les relations entre les utilisateurs sont hiérarchisées. Dans de telles applications, on peut envisager, lorsque les ressources manquent, de maintenir la QoS des connexions des utilisateurs les plus prioritaires au détriment des connexions d'utilisateurs moins prioritaires. La gestion coordonnée de connexion a dejà fait l'objet de travaux que nous avons présentés en section 1.4.6 du chapitre 1. Notre approche de gestion coopérative de la QoS au niveau Transport s'inspire notamment de la gestion de la QoS dans les protocoles de Transport à multi-connexions [CHA 96], [OWE 98], [STE 04].

Pour mettre en œuvre une telle gestion de la QoS, plusieurs aspects sont à considérer :

- *la répartition des ressources*. Connaissant la QoS requise par les connexions impliquées dans la collaboration et la quantité de ressources disponibles dans le réseau, le besoin est de calculer une répartition différente de ces ressources au profit des connexions les plus prioritaires.
- *la récupération des données de supervision*. Pour pouvoir calculer la répartition évoquée ci-dessus, il est nécessaire de collecter les informations de QoS relatives aux différentes connexions. La collecte de ces informations nécessite des échanges de messages et donc la mise en œuvre d'un protocole approprié. Ce protocole dépend de la manière dont est envisagée la mise en œuvre du calcul de la répartition des ressources. Ce calcul peut être centralisé, distribué, sur les hôtes d'extrémités, au niveau des routeurs, etc,...
- *la mise en œuvre au niveau des hôtes d'extrémité*. Le déploiement de la décision de répartition des ressources implique également la mise en œuvre d'un protocole de (re-)configuration pour les entités de niveau Transport. Lui aussi dépendra de la façon dont sont mis en œuvre le calcul de la répartition des ressources et la collecte des données de supervision.

La façon d'implanter les différents composants permettant de répondre aux trois besoins décrits ci-dessus, constitue l'architecture du système de gestion coopérative de la QoS.

Dans ce chapitre, nous nous intéressons dans une première partie (section 3.1) au problème de la répartition de la bande passante entre les connexions en tenant compte de leurs différentes priorités. Après avoir explicité le problème et défini les données nécessaires à sa résolution, nous le formulons sous la forme d'un problème d'optimisation. Deux critères de minimisation sont introduits et discutés.

Dans une deuxième partie, nous présentons les extensions de l'architecture du système NETQoS, présenté au chapitre précédent, nécessaires pour effectuer une gestion coopérative de la QoS. Les extensions proposées concernent toujours l'architecture fonctionnelle.

Enfin, une troisième partie (section 3.3) propose des éléments de réflexion, en termes de protocoles et d'outils pour la collecte des données de supervision.

3.1 Répartition de la bande passante

Nous considérons des activités coopératives distribuées qui s'appuient sur des groupes structurés d'utilisateurs induisant des priorités entre les communications qui se répercutent sur les connexions de Transport sous-tendant le transfert des données correspondantes.

L'évolution des activités coopératives distribuées peut intervenir à des moments prévisibles ou non. Cette évolution peut résulter d'un changement dans la composition du groupe ou dans la répartition des rôles sur les membres du groupe. Cette évolution induit également un changement dans les priorités de communication. Une nouvelle répartition des ressources de communication est alors à calculer. D'un autre coté, et notamment pour les activités soutenues par des moyens de communications sans fil, la redistribution des ressources peut s'avérer nécessaire pour adapter les protocoles de communication aux variations des contraintes de l'infrastructure de communication. Par exemple, le contexte des réseaux sans fil, mobiles et ad-hoc implique une variation des ressources, notamment de la bande passante. On envisage alors de répartir la bande passante de manière différente entre les connexions de Transport en fonction des priorités des utilisateurs et des communications associées.

Cette section explicite à titre d'exemple le contexte des applications coopératives distribuées sur des réseaux mobiles, notamment celui des OIU, puis présente le problème de la répartition de la bande passante dans un contexte de réseau filaire pour une adaptation aux changements dans le contexte applicatif. Le modèle d'optimisation permettant la résolution de ce problème est ensuite présenté et discuté.

3.1.1 Contexte applicatif et exemple : les OIU

Les réseaux sans fil et ad-hoc, les réseaux de capteurs et les équipements mobiles de type PDA ou smartphone, sont les technologies qui sous-tendront le déploiement de nouveaux services dans les futurs réseaux ambiants, ubiquitaires et pervasifs.

Les technologies des réseaux ambiants présentent des caractéristiques différentes de celles sur lesquelles s'appuient les services actuels de l'Internet, c'est-à-dire des réseaux filaires à haut débit, des équipements fixes à fortes ressources en termes de capacité de calcul, de stockage, et d'énergie. Comparativement, dans les réseaux ambiants, la bande passante est variable car fonction de la localisation et de la mobilité des utilisateurs, de leur point d'accès courant et des perturbations courantes des liens sans fil. En outre, les ressources des équipements sont potentiellement limitées.

Par ailleurs, ce type d'environnement ouvre des perspectives applicatives nouvelles autorisant la coopération dynamique d'utilisateurs mobiles dans le cadre d'exercices tels que les OIU en situation de crise. Les OIU impliquent des groupes structurés d'individus collaborant à une mission commune ; par essence, ces opérations ne peuvent pas s'appuyer sur une infrastructure de réseaux uniquement filaires, en raison de la mobilité de

certains des intervenants et de l'absence totale ou partielle de l'infrastructure de communication initiale. Les communications s'appuient donc sur des réseaux sans fil, potentiellement ad-hoc, et sur des machines aux capacités potentiellement limitées, dont les ressources sont par nature variables et le plus souvent limitées.

Pour aider au déploiement de ce type d'activité, les futurs services de communication, qu'ils relèvent des couches application, transport ou réseau auront à prendre en compte des besoins multiples et évolutifs dans le temps liés à l'activité ciblée, à la mobilité des utilisateurs, aux flux de données échangés (audio, vidéo,...) et aux contraintes de l'environnement de communication. De plus, pour des activités telles que les OIU, l'évolution de la mission induira inévitablement des changements dans la hiérarchie et dans la structure des coopérations entre intervenants, par exemple suite aux informations acquises par les exécutants sur le terrain ou par les décisions des coordinateurs de la mission.

Les paragraphes suivants décrivent un scénario d'activité coopérative de type OIU. Au travers de ce scénario simple, nous illustrons en quoi ce type d'activité induit une variation des priorités des communications en fonction de l'évolution de la mission.

3.1.2 Description des Opérations d'Intervention d'Urgence (OIU)

Pour des activités de type OIU, la coopération est basée sur l'échange d'informations entre des participants mobiles, collaborant à la réalisation d'une mission commune. Ces informations peuvent être des données de différents types, notamment des données d'observation et des données d'analyse, produites périodiquement ou immédiatement après un évènement particulier.

Ces données sont transmises sous différentes formes (média), notamment textuelle, audio et/ou vidéo. Chaque média possède différents codecs tels que H263 ou MPEG4, qui induisent des besoins en QoS différents pour le transfert des données de bout-en-bout. Les besoins en QoS peuvent être exprimés en termes de bande passante, de pourcentage de pertes, de contraintes de synchronisation intra/inter-media. Ils peuvent aussi être exprimés sous forme de contraintes temporelles dues à l'interactivité requise entre les participants.

Les participants possèdent des terminaux mobiles, et communiquent via des réseaux filaires ou sans fil. En fonction du rôle qu'il occupe dans la mission, chaque participant fournit un ensemble de services. Les services que les participants doivent fournir sont dynamiquement assignés aux participants selon l'évolution de la mission, en fonction de leurs capacités et leur localisation.

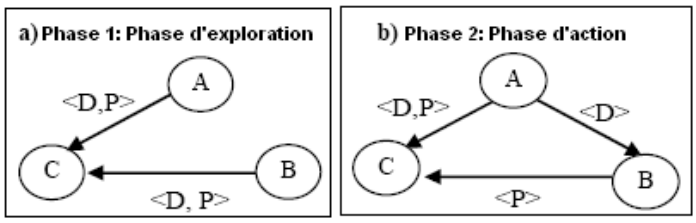
Une équipe d'intervention d'urgence peut ainsi être constituée de participants ayant différents rôles, par exemple : un contrôleur de la mission, plusieurs coordinateurs, et plusieurs équipes d'investigateurs, chacun dirigé par un coordinateur. Chaque participant peut remplir différentes fonctions :

– le contrôleur a par exemple pour fonction de diriger et d'autoriser les actions qui doivent être réalisées par les coordinateurs et les investigateurs ;
– en relation avec le contrôleur et les autres coordinateurs, un coordinateur doit diriger ses investigateurs, en leur précisant les tâches à réaliser. Il doit également collecter, interpréter, synthétiser les informations reçues des investigateurs et les diffuser vers les investigateurs ;
– les investigateurs ont pour fonction d'explorer le champ opérationnel, d'observer, d'analyser, et de faire un rapport décrivant la situation. Les investigateurs peuvent également aider, porter secours, réparer, etc.

On considère le scénario d'une équipe d'intervention d'urgence composée d'un contrôleur fixe, appelé C, et de deux investigateurs, A et B, se déplaçant sur un champ d'exploration. Pour simplifier l'explication du modèle, le contrôleur remplit les rôles du contrôleur et du coordinateur.

Les fonctions remplies par les investigateurs sont d'Observer (O) le champ d'exploration et de Rapporter (R) sur ce qui est observé. Dans les deux cas, des données de retour (feedback) sont envoyées au contrôleur. Les feedback D (associés à la fonction O) sont des données Descriptives ; elles sont transmises sous forme audio/vidéo (a,v). Les feedback P (associés à la fonction R) sont des données Produites ; elles expriment l'analyse de la situation par un investigateur. Elles sont transmises sous forme audio (a).

La fonction du contrôleur est de Superviser l'ensemble de la mission, en décidant des actions à exécuter en fonction de l'observation et de l'analyse des feedback D transmis par les investigateurs. Pour prendre cette décision, le contrôleur peut également utiliser les feedback P transmis par les investigateurs. Enfin, chaque participant est supposé posséder un seul terminal mobile, équipé d'une technologie sans fil pour la communication.



	Com.	Data	Media	Priorité
Phase 1	A-C	D	a,v	haute
		P	a	medium
	B-C	D	a,v	haute
		P	a	medium

	Com.	Data	Media	Priorité
Phase 2	A-C	D	a,v	haute
		P	a	medium
	A-B	D	a,v	basse
	B-C	P	a	basse

FIGURE 3.1 – *Scénario d'Opération d'Intervention d'Urgence*

Le scenario est divisé en deux étapes successives (Figure 3.1).

- **La première étape est l'étape d'investigation.** Les investigateurs A et B envoient de manière continue des feedback D au contrôleur C ; ils envoient également des feedback P périodiquement. Il n'y a pas de différence de priorité entre les communications A-C et B-C, mais la transmission des feedback D est plus importante que celle des feedback P. La première étape se termine quand une situation critique est découverte par un investigateur, par exemple A.
- **La deuxième étape est l'étape d'action.** Après avoir découvert une situation critique, A continue de remplir les mêmes fonctions que durant l'étape d'exploration (Observer et Rapporter), mais envoie aussi les feedback D à l'investigateur B. Il continue d'envoyer des feedback P à C ; B rapporte maintenant seulement des feedback P au contrôleur C ; elles correspondent à l'analyse des feedback D transmis par A.

En raison du caractère critique de la situation décrite par A, la communication A-C est plus importante que les communications A-B et A-C. De plus, les échanges de feedback D entre A et C sont les plus importants ; les données P entre A et C sont d'importance moyenne ; les feedback D entre A et B, et les feedback P entre B et C sont les moins prioritaires.

Ce scénario fait donc apparaitre des priorités entres les communications. Ces priorités sont amenées à évoluer au cours de la mission. Au niveau des connexions de Transport sous-jacentes, on souhaite répartir la bande passante dont bénéficie chacune de ces connexions en fonction de leurs priorités relatives. Cette problématique et la façon dont nous proposons de l'aborder sont décrites dans la section suivante.

3.1.3 Problématique et approche

La problématique liée au contexte précédent est la suivante. Les applications multimédias présentent des besoins en QoS et l'environnement considéré implique potentiellement une pénurie de ressources, notamment une bande passante limitée et variable. De plus il existe des priorités entre les communications, liées au caractère collaboratif des applications ainsi qu'à la structure hiérarchique du groupe d'utilisateurs considéré. Notre approche consiste à tirer partie de ces priorités dans la gestion de la QoS au niveau Transport, en maintenant la QoS associée à une connexion au détriment de la QoS d'une ou plusieurs autres connexions, moins prioritaires. Nous nous appuyons pour cela sur les protocoles de Transport configurables.

Pour aborder cette problématique, nous avons décidé d'étudier la mise en œuvre de l'approche consistant à améliorer le débit obtenu par une connexion en dégradant celui d'une ou plusieurs autres connexions dans un contexte filaire dans un premier temps, afin d'en évaluer la pertinence et la difficulté de mise en œuvre. La considération d'autres paramètres de QoS est une perspective de nos travaux, de même que la considération d'un contexte de réseaux sans fil.

L'approche classique, non coopérative, consistant à gérer la pénurie de ressources de manière équitable amène une dégradation de toutes les

connexions y compris des plus importantes. On cherche donc, comme par exemple avec le PHB des réseaux DiffServ qui traite de façon prioritaire les paquets les plus importants, à donner plus de bande passante aux connexions les plus prioritaires. Cette approche coopérative est décrite dans les sections suivantes.

Description informelle de l'approche proposée

On considère une topologie de réseau constituée de N nœuds et d'un nombre C de connexions. A chaque connexion sont associés des besoins en QoS que nous limitons ici au débit requis. Le débit d'émission de chaque connexion découle de la mise en œuvre d'un mécanisme de contrôle de congestion, qui de manière classique évite l'effondrement du réseau. Pour mettre en place notre approche de gestion coopérative de la QoS, nous faisons l'hypothèse que les débits d'émission sont des débits moyens qui restent stables sur une période de temps significativement longue, de l'ordre de la centaine de secondes, éventuellement même de quelques minutes. L'hypothèse d'une telle période de stabilité du réseau, est nécessaire pour pouvoir envisager de faire collaborer les connexions en dégradant la QoS de certaines au profit d'autres.

Le paramètre de QoS que nous souhaitons améliorer est le débit d'émission obtenu par une application. Si une connexion n'obtient pas une bande passante satisfaisante au regard de ses besoins en QoS, c'est-à dire que son débit d'émission est inférieur au débit requis par l'application, on cherche alors à augmenter ce débit d'émission de la quantité manquante. Nous appelons cette quantité manquante l'*incrément*. Pour pouvoir augmenter le débit de cette connexion et maintenir un niveau de congestion acceptable, il faut trouver quelles connexions doivent diminuer leur débit afin de libérer la bande passante nécessaire. La quantité de bande passante libérée par ces connexions est appelée le *décrément*.

La récupération des différentes informations relatives à la mise en œuvre de cette approche, la possibilité de trouver une solution et le déploiement de la solution, c'est-à-dire sa mise en application sur les hôtes d'extrémité reposent sur l'hypothèse de stabilité des débits moyens des applications dans le réseau pendant une période de temps suffisamment longue.

Priorités des connexions

On considère dans un premier temps deux types de connexions, aux priorités différentes :

– les connexions de type 0 (les moins prioritaires) qui ne pourront jamais bénéficier d'une augmentation de bande passante, mais qui seront en revanche amenées à collaborer, c'est-à-dire à donner une partie de leur bande passante à des connexions de type 1 (plus prioritaires) définies ci-dessous. Dans la suite de ce chapitre, nous appelons les connexions de type 0 *connexions donneuses* ; tel qu'indiqué ci-dessus, la quantité de bande passante que chacune donne est appelé le *décrément*.
– les connexions de type 1 (les plus prioritaires) qui ne céderont jamais

une partie de leur bande passante, mais qui seront amenées à demander aux connexions de type o de collaborer en donnant une partie de leur bande passante lorsque la valeur de la bande passante dont elles disposent passe en-deçà d'un certain seuil. Dans la suite de ce chapitre, nous appelons ces connexions *connexions receveuses* ; tel qu'indiqué ci-dessus, la quantité de bande passante que réclame chacune est appelé l'*incrément*.

Nous ne considérons ici que deux types de connexions afin de simplifier la complexité du problème. En effet, considérer simplement des connexions qui cèdent de la bande passante et des connexions qui reçoivent de la bande passante, permet d'éviter de traiter d'éventuels conflits. Par exemple, si une connexion peut à la fois donner et recevoir, il peut se trouver une situation dans laquelle la connexion donne et ce faisant obtient un débit d'émission insuffisant qui implique une coopération induisant une augmentation de débit. On peut s'attendre alors à un phénomène d'oscillations. Le but de notre étude étant de montrer la pertinence d'un mécanisme permettant d'augmenter le débit d'une connexion au détriment d'une ou plusieurs autres, nous évitons de traiter le type de problèmes liés à la gestion de priorités multiples en ne considérant que des connexions exclusivement donneuses ou receveuses.

Dans un deuxième temps, en perspective de ce travail, on peut envisager une gestion plus complexe des connexions qui pourraient endosser les deux rôles, donneuses et receveuses, à la fois. Il faudrait alors résoudre les conflits qui en résultent. De plus on peut envisager des priorités entre les connexions donneuses et/ou entre les connexions receveuses.

Routeurs, interfaces et ressources disponibles

On considère dans cette étude des liens full-duplex entre les routeurs et une capacité des interfaces en sortie des routeurs égale à la capacité des interfaces en entrée des routeurs. Dans les exemples qui suivent, les liens considérés autorisent des connexions dans un seul sens, donc les connexions qui sortent d'une interface se partagent la bande passante correspondant à la capacité de cette interface. Par conséquent, pour une connexion, le goulot d'étranglement qui fixe son débit d'émission est constitué par l'interface de sortie d'un des routeurs de son chemin de données. Dans la suite, on parlera donc d'interface sans préciser qu'il s'agit d'une interface de sortie.

Une connexion receveuse réclame un incrément de débit lorsqu'une ou plusieurs interfaces sur son chemin de données ne possèdent pas les ressources suffisantes pour lui permettre d'obtenir le débit qu'elle souhaite. Dans la suite de notre étude nous distinguons deux types d'interfaces : les interfaces *non chargées* et les interfaces *chargées*.

Définition 3.1 *Une interface non chargée est une interface pour laquelle la somme des débits des connexions sortantes est inférieure à la capacité en sortie de cette interface.*

Exemple : Si on considère trois flux émettant à un débit moyen de

2Mbits/s et passant par un routeur dont l'interface de sortie offre un débit de 10Mbits/s, on voit alors clairement que l'interface n'est pas chargée et qu'elle possède même une "marge".

Définition 3.2 *La marge μ d'une interface est définie par la différence entre la somme des débits sortants et la capacité en sortie de l'interface.*

Dans notre exemple, la marge est égale à 10 - (3 x 2) = 4 Mbits/s.

Définition 3.3 *Une interface est chargée lorsque la somme des débits sortants et la somme des incréments requis est supérieure à la capacité du lien en sortie de l'interface.*

Définition 3.4 *Un routeur est chargé lorsque l'une au moins de ses interfaces est chargée. On note R_{ch}, l'ensemble des routeurs chargés.*

3.1.4 Formalisation du problème

Dans cette section, nous formalisons les différentes notions décrites dans la section précédente présentant le problème de répartion de la bande passante.

Définition 3.5 *On appelle R l'ensemble des routeurs r_i du réseau, $R = \{r_i\}$. Un routeur r_i est défini par un ensemble d'interfaces $\mathcal{I}_j$.*

Définition 3.6 *On appelle "donneuse" une connexion qui peut donner une partie de sa bande passante au profit d'autres connexions plus prioritaires qui partagent avec elle des ressources communes (les routeurs et leurs interfaces). On note D, l'ensemble des connexions de ce type présentes dans le réseau considéré.*

$$D = \{d_i\} \text{ avec } d_i = \left| \begin{array}{l} t_{obt} \\ \Delta \\ R_t = \{r_j\} \subseteq R \qquad \text{tel que } r_j \text{ est un routeur par lequel passe } d_i \end{array} \right.$$

avec $d_i.t_{obt}$, la valeur du débit moyen obtenu par la connexion d_i, $d_i.\Delta$ le décrément de d_i et $d_i.R_t$, l'ensemble des routeurs du chemin de données de d_i.

Définition 3.7 *On appelle "receveuse" une connexion qui réclame un incrément de débit. On note P, l'ensemble des connexions de ce type présentes dans le réseau considéré.*

$$P = \{p_i\} \text{ avec } p_i = \left| \begin{array}{l} t_{req} \\ t_{obt} \\ \delta \\ R_t = \{r_j\} \subseteq R \qquad \text{tel que } r_j \text{ est un routeur par lequel passe } p_i \end{array} \right.$$

avec $p_i.t_{req}$, la valeur du débit moyen requis par la connexion p_i, $p_i.t_{obt}$, la valeur du débit moyen obtenu par p_i, $p_i.\delta$ l'incrément de débit souhaité et $p_i.R_t$, l'ensemble des routeurs du chemin de données de p_i.

Définition 3.8 *La quantité $p_i.\delta$ de bande passante à obtenir pour une application liée à une connexion receveuse est donnée par :*

$$p_i.\delta = p_i.t_{req} - p_i.t_{obt}$$

Définition 3.9 *Une interface $\mathcal{I}_i$ est définie par :*

$$\mathcal{I}_i = \left| \begin{array}{l} \{D_e, D_s\} \\ \{P_e, P_s\} \\ c \\ \mu \end{array} \right.$$

Avec :
- *$\mathcal{I}_i.D_e$, le sous-ensemble des connexions donneuses entrant sur cette interface et $\mathcal{I}_i.D_s$, le sous-ensemble des connexions donneuses sortant de cette interface ;*
- *$\mathcal{I}_i.P_e$, le sous-ensemble des connexions receveuses entrant sur cette interface et $\mathcal{I}_i.P_s$, le sous-ensemble des connexions receveuses sortant de cette interface ;*
- *$\mathcal{I}_i.c$, la capacité du lien en sortie de cette interface ;*
- *$\mathcal{I}_i.\mu$ la marge associée à cette interface (cf. la définition 3.2).*

Définition 3.10 *La marge μ d'une interface $\mathcal{I}_j$ est donnée par :*

$$\mathcal{I}_j.\mu = \mathcal{I}_j.c - \left(\sum_{d_i \in \mathcal{I}_j.D_s} \mathcal{I}_j.d_i.t_{obt} + \sum_{p_k \in \mathcal{I}_j.P_s} \mathcal{I}_j.p_k.t_{obt} \right)$$

Définition 3.11 *Une interface $\mathcal{I}_i$ est chargée lorsque la somme des débits sortants des connexions donneuses et receveuses sur cette interface, plus la somme des incréments des connexions receveuses est supérieure à la capacité en sortie de cette interface.*

$$\sum_{d_j \in \mathcal{I}_i.D_s} d_j.t_{obt} + \sum_{p_j \in \mathcal{I}_i.P_s} (p_j.t_{obt} + p_j.\delta) > \mathcal{I}_i.c$$

Remarque 3.1 *Dans l'expression ci-dessus, la somme des débits obtenus par les connexions donneuses et receveuses :*

$$\sum_{d_j \in \mathcal{I}_i.D_s} d_j.t_{obt} + \sum_{p_j \in \mathcal{I}_i.P_s} p_j.t_{obt}$$

ne pouvant pas excéder $\mathcal{I}_j.c$, alors si l'interface est chargée, c'est qu'il existe au moins un terme $p_j.\delta$ non nul. En d'autres termes, une interface chargée est forcément traversée par au moins une connexion receveuse.

Exemple : Considérons trois flux qui émettent à un débit moyen de 2 Mbits/s, souhaitant chacun une augmentation de débit de 0.5 Mbits/s et passant par un routeur dont l'interface de sortie offre un débit de 7 Mbits/s, on obtient :
$(3 \times 2) + (3 \times 0.5) = 7.5$ Mbits/s, ce qui est supérieur à la capacité en sortie de l'interface (7 Mbits/s). L'interface est donc chargée.

3.1.5 Calcul de la répartition de bande passante

Considérons une connexion p_j de type 1 (receveuse) qui voit le débit dont elle dispose atteindre une valeur inférieure au seuil acceptable de QoS. Pour augmenter le débit de cette connexion et procéder à une nouvelle répartition de la bande passante au niveau des autres connexions, il faut :

1) Identifier la valeur de $p_j.\delta$, telle que définie par la définition 3.8.
2) Identifier les routeurs, dont au moins une interface est chargée, du chemin de données de cette connexion (routeurs de l'ensemble R_{ch}).

Les interfaces chargées des routeurs vont permettre de définir l'ensemble des connexions qui peuvent collaborer (parmi celles qui traversent ces routeurs) en donnant tout ou partie de leur bande passante, tandis que pour les interfaces non chargées des routeurs, on ne tiendra pas compte des connexions qui les traversent. En effet, on considère qu'un routeur dont l'interface est non chargée, peut supporter l'augmentation de débit requise, sans qu'il soit nécessaire de réduire le débit de la même quantité pour les autres connexions qui traversent cette interface non chargée.

On note $\mathcal{D}$ l'ensemble des connexions donneuses passant par les routeurs de l'ensemble R_{ch} ($\mathcal{D} \subseteq D$) .

En se basant sur les définitions précédentes, nous cherchons à calculer les décréments des connexions donneuses $d_i \in \mathcal{D}$ en fonction des incréments requis par les connexions receveuses. Les décréments notés $d_i.\Delta$ correspondent à la diminution de débit que vont subir les connexions donneuses. On cherche donc à les minimiser en fonction du besoin sur chaque routeur. La section suivante présente une formalisation de ce problème.

3.1.6 Problème P_1 : minimiser la bande passante globale cédée

Le problème considéré, qui consiste à déterminer les décréments à appliquer aux connexions donneuses afin de satisfaire les besoins des connexions receveuses ne bénéficiant pas d'une bande passante suffisante, admet un ensemble de solutions, parmi lesquelles nous allons choisir celles qui optimisent un certain critère z_1. Le critère considéré concerne la quantité de bande passante libérée dans sa globalité qu'il va s'agir de minimiser. Nous posons donc le problème suivant :

$$\text{Minimiser } z_1 = \sum_{d_i \in \mathcal{D}} d_i.\Delta$$

avec les contraintes, $\forall r_k \in R_{ch}, \forall \mathcal{I}_l$ chargée :

$$\begin{cases} \sum_{d_i \in \mathcal{I}_l.D_s} d_i.\Delta \geq \sum_{p_j \in \mathcal{I}_l.P_s} p_j.\delta - \mathcal{I}_l.\mu & (1) \\ \text{et} & \\ d_i.\Delta \leq d_i.t_{obt} \qquad i = 1, \ldots, card(\mathcal{D}) & (2) \\ d_i.\Delta \geq 0 & (3) \end{cases}$$

La contrainte (1) exprime que pour tout routeur possédant une interface chargée, et pour toute interface chargée de ce routeur, la quantité de bande passante à libérer par les connexions donneuses doit au moins être égale à la quantité totale de bande passante requise par les connexions receveuses moins la marge restante sur cette interface.

La contrainte (2) exprime que la quantité de bande passante cédée par une connexion doit être inférieure au débit courant de cette connexion.

La contrainte (3) exprime que les quantités recherchées sont nécéssairement positives.

La section suivante présente deux exemples illustratifs du problème ainsi défini et de sa solution. Le deuxième exemple permet en particulier d'introduire un second problème que nous définissons en suivant.

3.1.6.1 Exemple 1

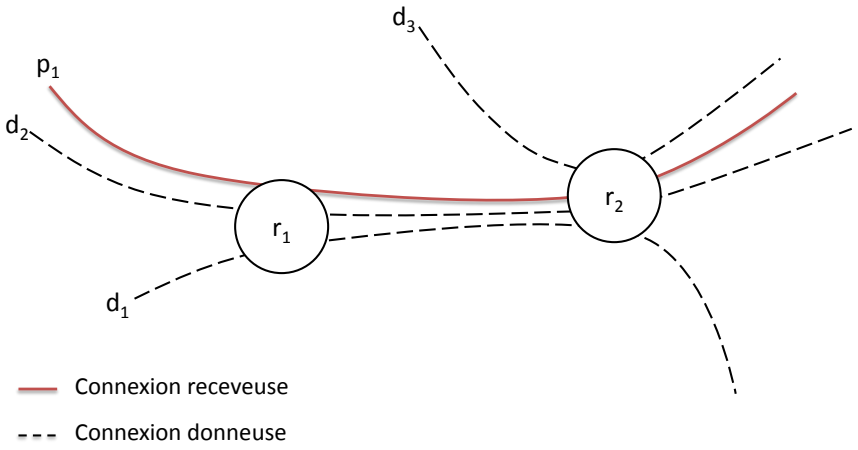


FIGURE 3.2 – *Réseau de connexions*

Soit le réseau simple illustré figure 3.2 faisant apparaître quatre connexions dont une receveuse et trois donneuses, et deux routeurs.

Les notations précédentes appliquées au réseau de la figure 3.2 donnent :

$R = \{r_1, r_2\}$ = ensemble de routeurs impliqués dans le chemin de la connexion receveuse ;
$P = \{p_1\}$ = ensemble des connexions receveuses, ici réduite à une seule ;
$D = \{d_1, d_2, d_3\}$ = ensemble des connexions donneuses.

$$p_1 = \left| \begin{array}{l} t_{req} = 17 \\ t_{obt} = 7 \\ \delta = 10 \\ R_t = \{r_1, r_2\} \end{array} \right. \qquad d_1 = \left| \begin{array}{l} t_{obt} = 6 \\ \Delta \\ R_t = \{r_1, r_2\} \end{array} \right.$$

$$d_2 = \left| \begin{array}{l} t_{obt} = 5 \\ \Delta \\ R_t = \{r_1, r_2\} \end{array} \right. \qquad d_3 = \left| \begin{array}{l} t_{obt} = 7 \\ \Delta \\ R_t = \{r_2\} \end{array} \right.$$

Deux interfaces nous intéressent : celles que traverse en sortie la connexion p_1 sur le routeur r_1 et sur le routeur r_2 :

$$r_1.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_1, d_2\} \\ P_s = \{p_1\} \\ c = 18 \\ \mu = 0 \end{array} \right. \qquad r_2.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_2, d_3\} \\ P_s = \{p_1\} \\ c = 19 \\ \mu = 0 \end{array} \right.$$

Dans cet exemple, les capacités associées aux deux interfaces à considérer, les débits des connexions qui les traversent et l'incrément requis par la connexion p_1 font que, d'après la definition 3.11, ces interfaces sont chargées et d'après la définition 3.10, leurs marges sont nulles.

Le problème P_1 consiste à déterminer les quantités $d_1.\Delta$, $d_2.\Delta$ et $d_3.\Delta$, décréments des connexions donneuses d_1, d_2 et d_3 compensant l'incrément $p_1.\delta = 10$ de la connexion receveuse p_1 et qui minimisent le critère z_1, la quantité de bande passante globale libérée.

Le problème P_1 s'exprime alors ainsi :

$$\text{Minimiser } z_1 = d_1.\Delta + d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes } \begin{cases} d_1.\Delta + d_2.\Delta \geq p_1.\delta \\ d_2.\Delta + d_3.\Delta \geq p_1.\delta \\ \\ d_1.\Delta \leq d_1.t_{obt} \\ d_2.\Delta \leq d_2.t_{obt} \\ d_3.\Delta \leq d_2.t_{obt} \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

Après instanciation, on obtient :

$$\text{Minimiser } z_1 = d_1.\Delta + d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes } \begin{cases} d_1.\Delta + d_2.\Delta \geq 10 \\ d_2.\Delta + d_3.\Delta \geq 10 \\ \\ d_1.\Delta \leq 6 \\ d_2.\Delta \leq 5 \\ d_3.\Delta \leq 7 \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

Le problème P_1 est posé sous la forme d'un ensemble d'inégalités linéaires sur n variables réelles et d'une fonction objectif, qui est elle aussi linéaire. L'algorithme du simplexe [DAN 55] permet de trouver une solution optimale pour la fonction objectif.

Pour notre exemple, on obtient alors une solution optimale donnée par : $d_1.\Delta = 5$, $d_2.\Delta = 5$, $d_3.\Delta = 5$.

Remarque 3.2 *L'algorithme du simplexe est une technique fondamentale pour les problèmes de programmation linéaire. Il est très efficace en pratique et il est implémenté dans la plupart des solveurs de programmes linéaires.*

En termes géométriques, l'ensemble des inégalités linéaires définit un polytope dans l'espace à n dimensions et il s'agit de trouver un sommet optimal pour la fonction de coût donnée. L'idée de l'algorithme consiste à partir d'un sommet quelconque du polytope et, à chaque itération, d'aller à un sommet adjacent s'il est possible d'en trouver un meilleur pour la fonction objectif. S'il n'y en a pas, l'algorithme s'arrête en concluant que le sommet courant est optimal.

On peut donc remarquer que s'il y a plusieurs sommets optimaux, on peut les trouver en réexécutant l'algorithme en partant d'un sommet différent à chaque fois. Nous utilisons pour notre problème un algorithme du simplexe ainsi modifié qui permet, à l'aide de permutations exhaustives, de trouver plusieurs sommets solutions pour la fonction objectif.

Définition 3.12 *On appelle **Simplexe avec permutations**, l'algorithme du simplexe modifié permettant de trouver plusieurs sommets solutions pour la fonction objectif.*

3.1.6.2 Exemple 2

Dans ce second exemple, nous considérons un réseau plus complexe qui fait apparaître plusieurs connexions receveuses.

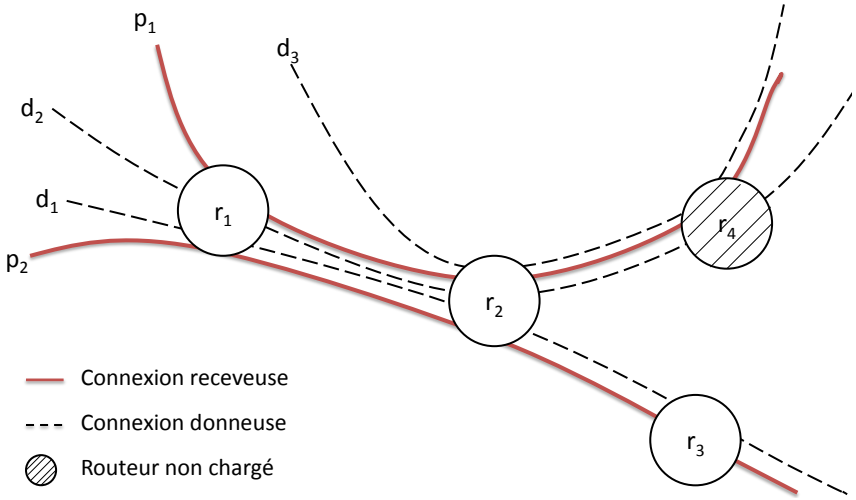


FIGURE 3.3 – *Réseau de connexions*

Les notations précédentes appliquées au réseau de la figure 3.3 donnent :
$R = \{r_1, r_2, r_3, r_4\}$
$P = \{p_1, p_2\}$
$D = \{d_1, d_2, d_3\}$

$$p_1 = \left| \begin{array}{l} t_{obt} = 10 \\ \delta = 5 \\ R_t = \{r_1, r_2, r_4\} \end{array} \right. \quad p_2 = \left| \begin{array}{l} t_{obt} = 10 \\ \delta = 5 \\ R_t = \{r_1, r_2, r_3\} \end{array} \right. \quad d_1 = \left| \begin{array}{l} t_{obt} = 10 \\ \Delta \\ R_t = \{r_1, r_2, r_3\} \end{array} \right.$$

$$d_2 = \left| \begin{array}{l} t_{obt} = 10 \\ \Delta \\ R_t = \{r_1, r_2, r_4\} \end{array} \right. \quad d_3 = \left| \begin{array}{l} t_{obt} = 10 \\ \Delta \\ R_t = \{r_2, r_3\} \end{array} \right.$$

Remarquons qu'au moins l'un des routeurs chargés a une marge nulle, puisqu'il existe forcément une interface du chemin de données qui constitue un goulot d'étranglement, fixant les débits des connexions.

La capacité des interfaces des différents routeurs et la marge restante sur ces interfaces (definition 3.2) sont les suivantes :

$$r_1.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_1, d_2\} \\ P_s = \{p_1, p_2\} \\ c = 40 \\ \mu = 0 \end{array} \right. \quad r_2.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_2, d_3\} \\ P_s = \{p_1\} \\ c = 34 \\ \mu = 4 \end{array} \right.$$

$$r_2.\mathcal{I}_2 = \left| \begin{array}{l} D_s = \{d_1\} \\ P_s = \{p_2\} \\ c = 22 \\ \mu = 2 \end{array} \right. \quad r_3.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_1\} \\ P_s = \{p_2\} \\ c = 23 \\ \mu = 3 \end{array} \right.$$

Dans cet exemple, les capacités associées aux quatre interfaces considérées, les débits des connexions qui les traversent et l'incrément requis par les connexions p_1 et p_2 font que, d'après la définition 3.11, ces interfaces sont chargées.

En revanche, l'interface $r_4.\mathcal{I}_1 = \left| \begin{array}{l} D_s = \{d_2, d_3\} \\ P_s = \{p_1\} \\ c = 40 \\ \mu = 10 \end{array} \right.$ est, d'après la définition 3.11, une interface non chargée. Les connexions donneuses qui la traversent, ne seront donc pas considérées pour le calcul des décréments, puisque cette interface peut supporter l'incrément demandé par p_1 sans devenir chargée.

Le problème P_1 consiste à déterminer les quantités $d_1.\Delta$, $d_2.\Delta$ et $d_3.\Delta$, décréments des connexions donneuses d_1, d_2 et d_3 compensant les incréments $p_1.\delta = 5$ et $p_2.\delta = 5$ des connexions receveuses p_1 et p_2, et qui minimise le critère z_1, la quantité de bande passante globale libérée.

P_1 s'exprime alors ainsi :

$$\text{Minimiser } z_1 = d_1.\Delta + d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes} \quad \begin{cases} d_1.\Delta + d_2.\Delta \geq p_1.\delta + p_2.\delta - r_1.\mathcal{I}_1.\mu \\ d_2.\Delta + d_3.\Delta \geq p_1.\delta - r_2.\mathcal{I}_1.\mu \\ d_1.\Delta \geq p_2.\delta - r_2.\mathcal{I}_2.\mu \\ d_1.\Delta \geq p_2.\delta - r_3.\mathcal{I}_1.\mu \\ \\ d_1.\Delta \leq d_1.t_{obt} \\ d_2.\Delta \leq d_2.t_{obt} \\ d_3.\Delta \leq d_2.t_{obt} \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

Par suite, on obtient :

$$\text{Minimiser } z_1 = d_1.\Delta + d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes} \quad \begin{cases} d_1.\Delta + d_2.\Delta \geq 10 \\ d_2.\Delta + d_3.\Delta \geq 1 \\ d_1.\Delta \geq 3 \\ \\ d_1.\Delta \leq 10 \\ d_2.\Delta \leq 10 \\ d_3.\Delta \leq 10 \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

En utilisant l'algorithme du simplexe avec permutations (définition 3.12), on obtient deux sommets solution :
$d_1.\Delta = 9$, $d_2.\Delta = 1$, $d_3.\Delta = 0$
et
$d_1.\Delta = 3$, $d_2.\Delta = 7$, $d_3.\Delta = 0$

3.1.7 Ajout d'un deuxième critère et définition du problème P_2 : minimiser l'excédent de bande passante libérée

L'exemple précédent met en évidence le fait que le critère utilisé par le problème P_1, c'est-à-dire z_1, peut aboutir à plusieurs solutions. Dans cette section, nous nous proposons de discriminer ces solutions au regard d'un critère supplémentaire (z_2) visant à minimiser la somme des quantités d'excédent de bande passante libérée au niveau de chaque interface.

En effet, le débit des connexions étant obtenu de bout-en-bout, il peut arriver que la quantité de bande passante libérée sur une interface soit plus importante que nécessaire, en fonction de la topologie des connexions. Il peut alors être intéressant de chercher à minimiser cette quantité excédentaire de bande passante libérée. De manière informelle, l'excédent de bande passante libérée au niveau d'une interface correspond à la marge de cette interface, ajoutée à la quantité de bande passante cédée par les connexions donneuses sortant par cette interface, auquelles on soustrait la quantité requise par les connexions receveuses sortant par cette interface.

L'excédent au niveau d'une interface $\mathcal{I}_l$, noté $\mathcal{I}_l.\chi$, est donné par :

$$\mathcal{I}_l.\chi = \sum_{d_i \in \mathcal{I}_l} d_i.\Delta + \mathcal{I}_l.\mu - \sum_{d_i \in \mathcal{I}_l} d_i.\delta$$

Les marges μ et les incréments δ sont des constantes qui peuvent donc être ignorées dans le critère. Si on cherche à minimiser globalement l'excédent sur les interfaces chargées, le critère s'exprime donc comme suit :

$$\text{Minimiser } z_2 = \sum_{d_i \in \mathcal{D}} k_i \times d_i.\Delta \qquad \text{avec } k_i = card(d_i.R_t \cap R_{ch})$$

k_i indique le nombre de fois où la connexion donneuse d_i traverse une interface chargée sur un routeur de son chemin de données.

Considérons l'exemple 2 de la section précédente. En évaluant le critère z_2 avec chacune des solutions optimales obtenues avec le critère z_1, on remarque que celles procurant la meilleure valeur pour z_2 constituent un sous-ensemble. En effet, si on calcule l'excédent libéré sur chaque interface chargée avec les 2 solutions du problème P_1, on obtient :

pour la solution $d_1.\Delta = 9$, $d_2.\Delta = 1$, $d_3.\Delta = 0$:
$r_1.\mathcal{I}_1 \Rightarrow \chi = 0$
$r_2.\mathcal{I}_1 \Rightarrow \chi = 0$
$r_2.\mathcal{I}_2 \Rightarrow \chi = 6$
$r_3.\mathcal{I}_1 \Rightarrow \chi = 7$

et pour la solution $d_1.\Delta = 3$, $d_2.\Delta = 7$, $d_3.\Delta = 0$:
$r_1.\mathcal{I}_1 \Rightarrow \chi = 0$
$r_2.\mathcal{I}_1 \Rightarrow \chi = 6$
$r_2.\mathcal{I}_2 \Rightarrow \chi = 0$
$r_3.\mathcal{I}_1 \Rightarrow \chi = 1$

On constate qu'avec la solution $d_1.\Delta = 3$, $d_2.\Delta = 7$, $d_3.\Delta = 0$, la somme des excédents est égale à 7 alors qu'avec la solution $d_1.\Delta = 9$, $d_2.\Delta = 1$, $d_3.\Delta = 0$, la somme des excédents est égale à 13. On préférera donc la solution $d_1.\Delta = 3$, $d_2.\Delta = 7$, $d_3.\Delta = 0$.

On peut à ce stade se demander si les solutions ainsi retenues sont également optimales par rapport au critère z_2. Pour cela, il faut poser le problème P_2 qui consiste à déterminer les quantités $d_1.\Delta$, $d_2.\Delta$ et $d_3.\Delta$, décréments des connexions donneuses d_1, d_2 et d_3 compensant les incréments $p_1.\delta = 5$ et $p_2.\delta = 5$ des connexions receveuses p_1 et p_2, et qui minimise le critère z_2, la somme des excédents de bande passante libérée au niveau de chaque interface.

La connexion d_1 passe par trois interfaces chargées ($r_1.\mathcal{I}_1$, $r_2.\mathcal{I}_2$, $r_3.\mathcal{I}_1$), la connexion d_2 passe par deux interfaces chargées ($r_1.\mathcal{I}_1$, $r_2.\mathcal{I}_1$) et la connexion d_3 par une interface chargée ($r_2.\mathcal{I}_1$). Les coefficients k_i sont donc : $k_1 = 3$, $k_2 = 2$ et $k_3 = 1$. Par suite, on obtient :

$$\text{Minimiser } z_2 = 3d_1.\Delta + 2d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes } \begin{cases} d_1.\Delta + d_2.\Delta \geq p_1.\delta + p_2.\delta - r_1.\mathcal{I}_1.\mu \\ d_2.\Delta + d_3.\Delta \geq p_1.\delta - r_2.\mathcal{I}_1.\mu \\ d_1.\Delta \geq p_2.\delta - r_2.\mathcal{I}_2.\mu \\ d_1.\Delta \geq p_2.\delta - r_3.\mathcal{I}_1.\mu \\ \\ d_1.\Delta \leq d_1.t_{obt} \\ d_2.\Delta \leq d_2.t_{obt} \\ d_3.\Delta \leq d_2.t_{obt} \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

Par conséquent, on obtient :

$$\text{Minimiser } z_2 = 3d_1.\Delta + 2d_2.\Delta + d_3.\Delta$$

$$\text{avec les contraintes } \begin{cases} d_1.\Delta + d_2.\Delta \geq 10 \\ d_2.\Delta + d_3.\Delta \geq 1 \\ d_1.\Delta \geq 3 \\ \\ d_1.\Delta \leq 10 \\ d_2.\Delta \leq 10 \\ d_3.\Delta \leq 10 \\ \\ d_1.\Delta, d_2.\Delta, d_3.\Delta \geq 0 \end{cases}$$

L'algorithme du simplexe modifié donne une solution unique :
$d_1.\Delta = 3$, $d_2.\Delta = 7$, $d_3.\Delta = 0$

qui correspond à la solution optimale du problème P_1, solution qui est préférée selon le critère z_2. Cette solution est donc optimale au regard des deux critères.

3.1.8 Conclusion sur l'utilisation des critères de minimisation globale (z_1) et de minimisation des excédents (z_2)

Notre approche consiste à considérer le critère z_1, à savoir minimiser la quantité de bande passante globale libérée, comme le critère prioritaire. Si le problème admet une solution unique, on ne cherche pas à utiliser le critère z_2, qui permet de minimiser la somme des excédents de bande passante sur chaque interface. En revanche, si le problème P_1 (critère z_1) admet plusieurs solutions, on évalue le critère z_2 pour les différentes solutions trouvées pour ne retenir que les meilleures par rapport à z_2.

3.2 Gestion coopérative de la QoS dans le système NETQoS

Dans cette section, nous présentons les extensions de l'architecture fonctionnelle du système NETQoS permettant de mettre en œuvre une gestion coopérative de la QoS et l'intégration du modèle présenté dans la section précédente 3.1.

Dans un premier temps (section 3.2.1 et 3.2.2), les principales modifications à apporter aux classes et composants initiaux sont exposées. La section 3.2.3 décrit ensuite le processus de coopération.

3.2.1 Identification des connexions

Dans le diagramme de classe présenté Figure 3.4, la classe *ConnectionId* contient deux attributs, les adresses IP sources et destination, et les ports sources et destination. Pour prendre en compte une gestion coopérative de la QoS, plusieurs autres informations doivent être associées à une connexion.

La classe *ConnectionId* est donc spécialisée en une classe *CollaboratingConnection* possédant un attribut *t_obt*, désignant le débit obtenu par une connexion, et un attribut *R_t*, désignant la liste des routeurs traversés par la connexion. Pour prendre en compte les informations spécifiques aux deux types de connexions, la classe *CollaboratingConnection* est spécialisée en une classe *GivingConnection* et en une classe *ReceivingConnection*.

La classe *GivingConnection* définit une connexion donneuse possédant un attribut Δ, correspondant au décrément. La classe *ReceivingConnecion* définit une connexion receveuse possédant un attribut *t_req*, désignant le débit requis pour cette connexion et un attribut δ, correspondant à l'incrément.

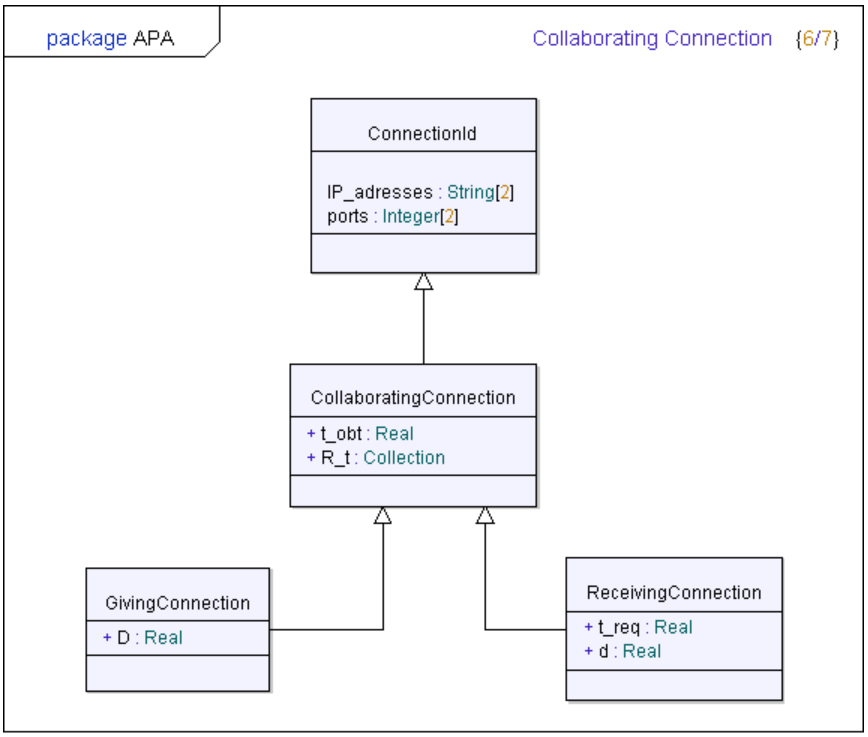


FIGURE 3.4 – *Classe CollaboratingConnection*

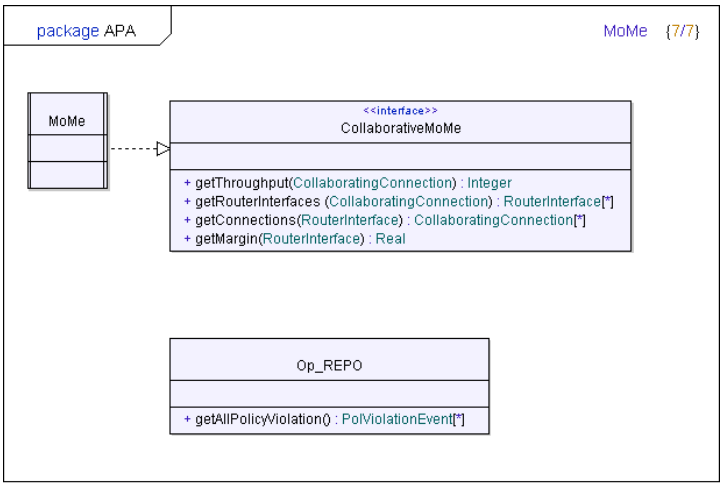


FIGURE 3.5 – *Classes Modifiées*

3.2.2 Classe MoMe

Le ***MoMe*** introduit dans le chapitre 2 est le composant du système NETQoS responsable de la supervision et de la collecte de données. La figure 3.5 est un diagramme de classe UML qui présente une extension de ce composant lui permettant d'être utilisé pour une gestion coopérative de la QoS.

Pour permettre à un composant tel que l'APA de récupérer les informations nécessaires à la prise de décision, le ***MoMe*** doit implémenter une interface *CollaborativeMoMe* dans laquelle sont définies plusieurs méthodes :

- *getThroughput ()* : Cette méthode prend en paramètre un objet *CollaboratingConnection* et renvoie son débit courant.
- *getRouterInterfaces ()* : Cette méthode prend en paramètre un objet *CollaboratingConnection* et renvoie une liste d'interfaces de routeurs traversées par cette connexion.
- *getConnections ()* : Cette méthode prend en paramètre une interface de routeur (objet *RouterInterface*) et renvoie une liste des connections (donneuses et receveuses) traversant cette interface.
- *getMargin ()* : Cette méthode prend en paramètre une interface de routeur et renvoie sa marge, telle qu'elle a été définie précédemment (definition 3.2).

Ces méthodes, invoquées notamment par le composant ***APA*** sur le ***MoMe*** permettront de récupérer les informations nécessaires pour le calcul de la répartition de la bande passante grâce au modèle présenté dans la section précédente. Dans la section 3.3, nous montrons comment ces méthodes pourraient être implémentées par le biais de certains outils et protocoles.

3.2.3 Mise en œuvre du processus de coopération

Le processus de coopération doit être initié lorsqu'une ou plusieurs connexions receveuses voient leur débit courant passer en dessous d'un certain seuil, inférieur au débit requis pour cette connexion. Dans le système NETQoS, il s'agit donc d'une violation de politique associée à une connexion.

Le diagramme présenté figure 3.6 réprésente les interactions et les messages échangés entre les différents composants du système NETQoS lors d'une violation de politique. Il s'agit d'une extension du diagramme présenté au chapitre précédent (Figure 2.11). Après avoir été notifié d'une violation de politique (message *notification(pve)*), l'APA récupère via les répertoires de stockage de politiques, les informations et les politiques opérationnelles de la connexion concernée par la violation de politique. Le processus de coopération débute ensuite et est décrit dans le sous-diagramme de séquence appelé *"PrepareCollaborativeAdaptation"* présenté dans le paragraphe suivant (Figure 3.7).

3.2.4 Diagramme de séquence pour la coopération

Pour chaque connexion receveuse qui voit son débit passer en dessous d'un certain seuil inférieur au débit requis par cette connexion, le composant APA stocke l'évènement violation de politique (*PolViolationEvent pve*) dans le composant Op_REPO. Cette action est effectuée pour chaque évènement de politique pendant une période T. A la fin de cette période, le composant OP_REPO contient l'ensemble des connexions receveuses qui ont vu leur débit passer en dessous d'un certain seuil (contenues dans l'objet PolViolationEvent). Le composant APA les récupère en utilisant la méthode *getAllPolicyViolation()* qui renvoie la liste des PolViolationEvent.

Puis, pour chaque connexion, le composant APA va récupérer les infor-

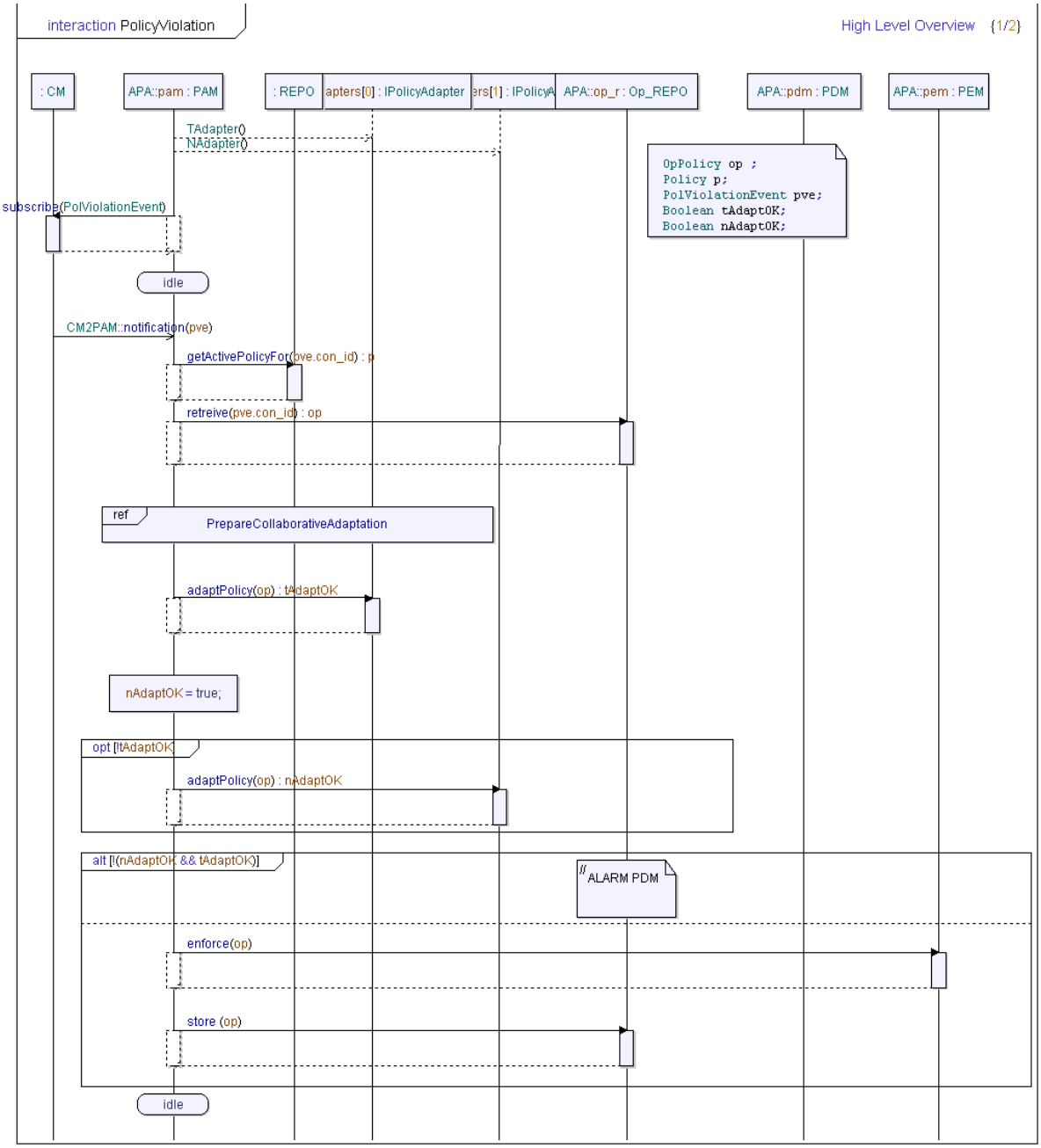


FIGURE 3.6 – *Diagramme de séquence Policy Violation*

mations nécessaires à la mise en œuvre du modèle de gestion coopérative présenté dans la section 3.1 du chapitre. Pour cela, il invoque auprès du MoMe les méthodes décrites au paragraphe 3.2.2. Pour chaque connexion, le composant APA récupère le débit moyen courant associé à cette connexion receveuse (méthode *getTroughput(cc)*), puis l'ensemble des interfaces de routeurs traversées par cette connexion (*getRouterInterfaces(cc)*). Pour chaque interface, la méthode *getConnections(ri)*, permet de récupérer l'ensemble des connexions donneuses qui traversent cette interface et la méthode *getMargin(ri)* la marge associée à cette interface.

Après avoir effectué ces différentes "boucles" (Figure 3.7), le composant APA possèdent maintenant l'ensemble des informations nécessaires à la

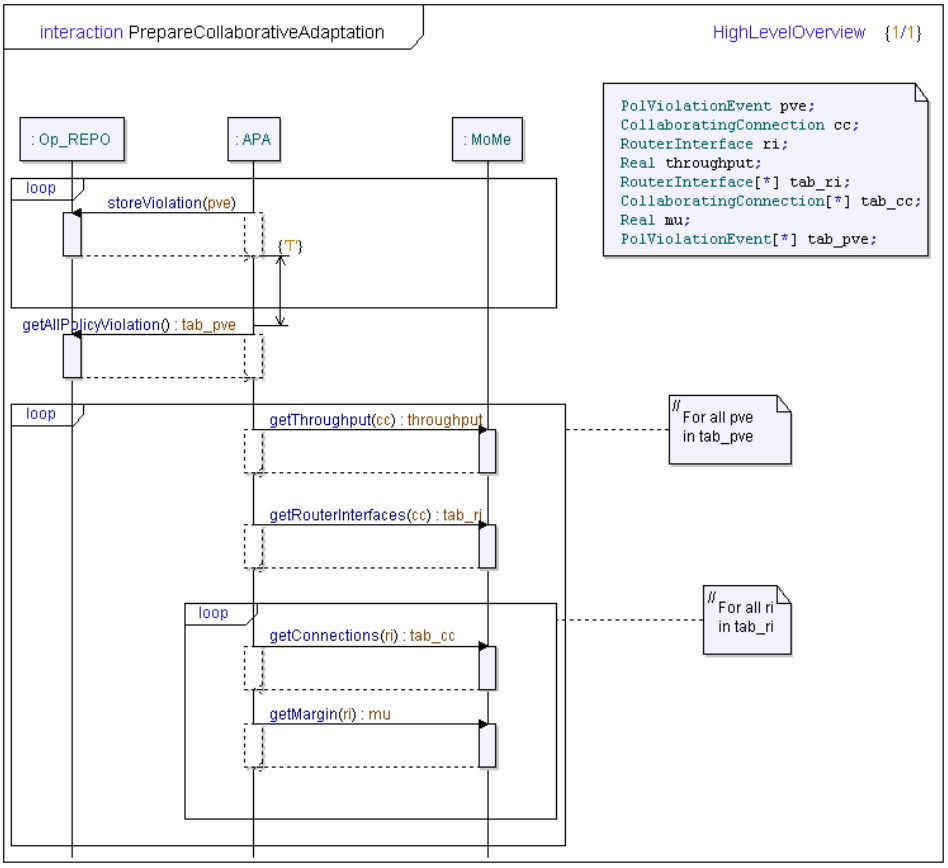


FIGURE 3.7 – *Diagramme de séquence pour une adaptation coopérative*

mise en œuvre du modèle. Il peut donc calculer la quantité de bande passante que chaque connexion donneuse doit donner pour satisfaire les incréments des différentes connexions receveuses.

3.3 Outils et protocoles pour la récupération des informations

Pour implémenter les différentes méthodes du composant MoMe permettant la récupération des informations décrites précédemment, plusieurs outils et protocoles sont envisageables. Nous les décrivons ci-après ; même si nous n'avons pas directement contribuer à cette facette de la problématique, nous pensons que ces outils pourraient être des briques de base pour l'implémentation des fonctions nécessaires d'un organe de monitoring tel que le MoMe.

3.3.1 SNMP

SNMP (Simple Network Management Protocol) [CAS 90]. Il s'agit d'un protocole qui permet aux administrateurs réseau de gérer les équipements du réseau et de diagnostiquer les problèmes de réseau. Le système de gestion de réseau est basé sur deux éléments principaux : un superviseur et

des agents. Le superviseur est la console qui permet à l'administrateur réseau d'exécuter des requêtes de management. Les agents sont des entités qui se trouvent au niveau de chaque interface connectant l'équipement managé au réseau et permettant de récupérer des informations sur différents objets.

Switchs, hubs, routeurs et serveurs sont des exemples d'équipements contenant des objets "gérables" (plus classiquement appelés "manageables"). Ces objets peuvent être des informations matérielles, des paramètres de configuration, des statistiques de performance et autres objets qui sont directement liés au comportement en cours de l'équipement en question. Ces objets sont classés dans une sorte de base de donnée appelée MIB ("Management Information Base"). SNMP permet le dialogue entre le superviseur et les agents afin de recueillir les objets souhaités dans la MIB.

L'architecture de gestion du réseau proposée par le protocole SNMP est donc fondée sur trois principaux éléments :
- les équipements managés (managed devices) sont des éléments du réseau (ponts, switchs, hubs, routeurs ou serveurs), contenant des « objets de gestion » (managed objects) pouvant être des informations sur le matériel, des éléments de configuration ou des informations statistiques ;
- les agents, c'est-à-dire une application de gestion de réseau résidant dans un périphérique et chargé de transmettre les données locales de gestion du périphérique au format SNMP ;
- les systèmes de gestion de réseau (network management systems notés NMS), c'est-à-dire une console à travers laquelle les administrateurs peuvent réaliser des tâches d'administration.

Le protocole SNMP offre donc la possibilité de récupérer des informations sur différents équipements. Dans le cadre d'un système de gestion coopérative de la QoS, on pourait s'en inspirer afin de récupérer les informations sur les routeurs (interfaces, marges, connexions traversantes...) ou sur les entités Transport d'extrémités (débit d'émission, débit requis,...).

3.3.2 Traceroute

Traceroute est un outil de diagnostic des réseaux, présent sur la plupart des systèmes d'exploitation, permettant de déterminer le chemin suivi par un paquet. La commande traceroute permet ainsi de dresser une cartographie des routeurs présents entre une machine source et une machine cible. La commande traceroute diffère selon les systèmes d'exploitation.

Traceroute appuie son fonctionnement sur le champ TTL des paquets IP. En effet chaque paquet IP possède un champ durée de vie (TTL, Time To Live) décrémenté à chaque passage d'un routeur. Lorsque ce champ arrive à zéro, le routeur, considérant que le paquet tourne en boucle, détruit ce paquet et envoie une notification ICMP à l'expéditeur.

Ainsi, Traceroute envoie des paquets à un port UDP non privilégié, réputé non utilisé (le port 33434 par défaut) avec un TTL valant 1. Le premier routeur rencontré va supprimer le paquet et renvoyer un paquet ICMP donnant notamment l'adresse IP du routeur ainsi que le temps de pro-

pagation en boucle. Traceroute va ainsi incrémenter séquentiellement le champ TTL, de manière à obtenir une réponse de chacun des routeurs sur le chemin, jusqu'à obtenir une réponse «port ICMP non atteignable» («ICMP port unreachable») de la part de la machine cible.

Dans le cadre d'un système de gestion coopérative de la QoS, le protocole Traceroute pourrait être utilisé afin de connaitre l'ensemble des routeurs traversés par une connexion receveuse.

3.3.3 Protocoles de routage

Les protocoles de routage, notamment les protocoles à état de liens (link-state protocol), permettent l'échange d'informations entre les routeurs. Chaque routeur établit des relations d'adjacence avec ses voisins immédiats en envoyant des messages à intervalle régulier. Chaque routeur communique ensuite la liste des réseaux auxquels il est directement connecté par des messages propagés de proche en proche à tous les routeurs du réseau. L'ensemble de ces messages forme la base de données des liens "Link-State Database" (LSDB), qui est identique pour tous les routeurs participants. Chaque routeur utilise ensuite un algorithme pour déterminer la route la plus courte vers chacun des réseaux connus dans la LSDB. En cas de changement de topologie, de nouveaux messages sont propagés de proche en proche.

Dans le cadre d'un système de gestion coopérative de la QoS, les protocoles de routage pourraient être utilisés pour récupérer les routes, et donc les différents routeurs, empruntés par les connexions.

3.3.4 Fonctions de Monitoring dans ETP

Le protocole ETP présenté dans le chapitre 1, (section 1.4.3), est le protocole de Transport configurable sur lequel nous nous appuyons, pour mettre en œuvre une gestion coopérative de la QoS.

La structure modulaire de ETP permet notamment d'instancier des micro-protocoles, sous la forme de composition de *processing modules*. On peut envisager d'instancier des modules permettant de calculer ou de récupérer, grâce au feedback d'ETP, divers paramètres de QoS et données de monitoring de niveau Transport, comme le délai, le débit, la gigue ou le taux de pertes.

Dans le chapitre précédent (section 2.3.4.3), nous avons montré comment utiliser la structure modulaire de ETP pour instancier dans le plan de Management un module *ETP Agent Interface* permettant de faire interagir le composant ETP Agent du système NETQoS avec une instance ETP en cours d'exécution.

De manière analogue, la figure 3.8 montre comment on peut instancier dans le plan de Management des modules de monitoring qui, utilisés conjointement avec le module NETQoS, permettraient de centraliser les valeurs des paramètres de QoS mesurés au sein du MoMe, par exemple. Les modules "Monitoring Header" du plan de données (IN et

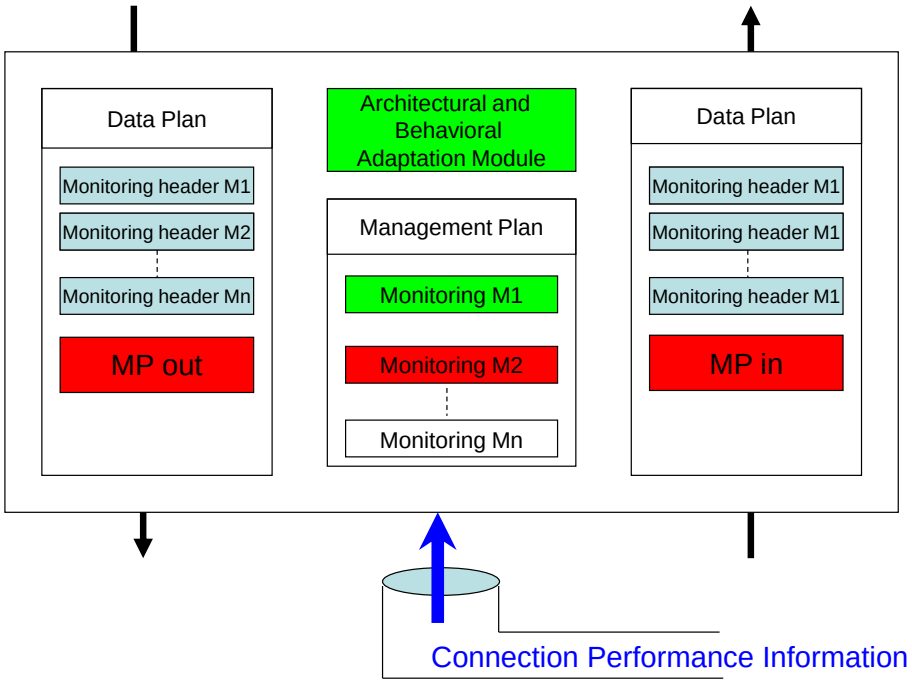


FIGURE 3.8 – *Fonctions de Monitoring dans ETP*

OUT) servent à encapsuler et désencapsuler les éventuels entêtes nécessaires à la mise en œuvre du monitoring.

CONCLUSION

Dans ce chapitre, nous avons proposé une contribution originale pour l'élaboration d'un système de gestion coopérative de la QoS. Dans le contexte des applications coopératives qui induit potentiellement des priorités entre les connexions, un tel système de gestion de la QoS peut être intéressant pour répondre au mieux aux pénuries de ressources induites par le réseau ou son utilisation. Pour mettre en place un tel système de gestion coopérative, trois aspects sont à prendre en compte : la répartition bande passante, la récupération des données de supervision, et la mise en œuvre au niveau des hôtes d'extrémité.

Dans la première partie de ce chapitre, nous avons défini puis formalisé les données nécessaires à la résolution du problème de répartition de la bande passante. Le principe est de donner une partie de la bande passante à certaines connexions au détriment d'autres. Pour déterminer la quantité de bande passante que doivent libérer les connexions pour satisfaire le besoin des connexions les plus prioritaires, notre approche consiste à poser le problème sous la forme d'un problème de programmation linéaire, c'est-à-dire d'une fonction objectif à satisfaire sous contraintes. Nous avons par ailleurs discuté la pertinence et l'utilisation de différents critères.

Dans la deuxième partie de ce chapitre, nous avons montré comment le modèle présenté pouvait être intégré au système NETQoS présenté au chapitre 2. Les extensions du système concernant les connexions et les

services offerts par le composant de mesure et de monitoring (MoME) ont été décrits. Les différentes actions à mettre en œuvre au niveau du composant de gestion de l'adaptation (APA) pour collecter les données nécessaires afin d'utiliser le modèle de calcul de la répartition de la bande passante ont été discutées.

Dans la dernière partie de ce chapitre, nous avons présenté différents éléments (outils et protocoles) concernant le deuxième aspect à prendre en compte pour mettre en place un système de gestion coopérative de la QoS : la récupération des informations de supervision en utilisant notamment des protocoles et des outils tels que SNMP, Traceroute, les protocoles de routage à états de lien ou le protocole de Transport configurable ETP.

Les perspectives de ce travail sont nombreuses :

Stabilité des débits. L'approche coopérative que nous avons élaborée s'appuie sur l'utilisation d'un contrôle de congestion et sur l'hypothèse que le débit des connexions restent stables sur une période de temps suffisamment longue. Si la dynamique des débits des connexions ne permet pas de vérifier cette hypothèse, il faudra éventuellement le prendre en compte pour modifier le modèle.

Collecte d'informations. La résolution du modèle d'optimisation suppose de pouvoir récupérer un certain nombre de données. Il faut pouvoir évaluer la faisabilité et le coût d'une telle collecte d'informations. Si certaines informations ne sont pas récupérables avec un coût raisonnable, il faudra peut-être envisager une modification du modèle, par exemple en relâchant certaines contraintes d'inégalité.

Gestion des priorités. Une autre perspective concerne la gestion des priorités, pour prendre en compte des connexions à la fois donneuses et receveuses, ou encore pour prendre en compte des priorités relatives entre les connexions ; le but pourrait être par exemple de permettre à des connexions donneuses de plus ou moins contribuer à la coopération en donnant un pourcentage de leur bande passante différent en fonction de leur priorité.

Amélioration de l'algorithme du simplexe. Pour utiliser au mieux les deux critères définis aux sections 3.1.6 et 3.1.7 (minimisation globale, critère $z1$ et minimisation des excédents, critère $z2$), il serait utile de pouvoir obtenir tous les sommets solution du problème. On pourrait même envisager de pouvoir trouver l'ensemble complet des solutions, permettant ainsi d'affiner les quantités de bande passante cédées par les connexions, en relation avec les priorités par exemple. Cette perspective porte sur l'amélioration de l'algorithme basé sur le simplexe ou sur l'utilisation d'un autre algorithme de résolution permettant de trouver l'ensemble complet de solutions.

Désynchronisation de la décision. Nos travaux pourraient également s'étendre pour couvrir la phase de déploiement et de mise en œuvre de la décision et profiter d'expérimentations pour affiner le modèle de décision. Par exemple, on pourra étudier l'influence de la synchronisation

(ou de la désynchronisation) de la décision sur les hôtes d'extrémités. En effet, pour mettre en œuvre cette approche de gestion coopérative, on suppose qu'au même instant, toutes les connexions de Transport appliquent la décision concernant la quantité de bande passante dont elles vont maintenant bénéficier. S'il y a désynchronisation ou s'il faut un certain temps pour arriver à un régime permanent, il est sans doute utile, voire nécessaire, de prendre en compte cette dynamique dans le modèle.

Décision de collaboration. Enfin, d'autres perspectives concernent la décision de collaboration, comme par exemple déterminer à partir de quel seuil on doit commencer le processus de collaboration et comment on le détecte. De même, lorsque la collaboration n'est plus nécessaire, la question de savoir comment l'on revient à la situation initiale se pose.

Dans le chapitre suivant, nous présentons une étude de faisabilité de l'approche de gestion coopérative de la QoS, réalisée avec le simulateur ns-2. Outre la démonstration de faisabilité, les simulations effectuées ont permis de soulever les questions relatives à la stabilité des débits (point 1 des perspectives énoncées ci-dessus).

Mise en œuvre et évaluation de performance

4

Sommaire

4.1 Evaluation de performance de l'APA 95
4.1.1 Spécification du contexte de mesure 95
4.1.2 Métriques mesurées 97
4.1.3 Scénarios de mesures, résultats et analyses 97
4.1.4 Conclusions 102
4.2 Gestion coopérative de la QoS et répartition de débit . 104
4.2.1 Contrôle de congestion utilisé : TFRC 105
4.2.2 Mise en œuvre de l'approche 105
4.2.3 Etude de faisabilité - Scénarios simples 106
4.2.4 Etude de faisabilité - Scénarios plus complexes 109
Conclusion 116

Dans ce chapitre, nous présentons les principaux résultats des campagnes d'évaluation de performance attenantes aux contributions proposées dans ce mémoire. Ces évaluations comportent deux volets, respectivement relatifs aux propositions faites dans les chapitres 2 et 3. Nous les introduisons ci-après.

Le chapitre 2 a présenté la proposition d'architecture à laquelle nous avons contribué dans le cadre du projet NETQoS. Rappelons que cette architecture vise à permettre (pour ce qui concerne directement nos travaux) une gestion adaptative des protocoles de Transport configurables pour optimiser la QoS requise par des applications distribuées dans un environnement de réseaux hétérogènes. La première partie de ce chapitre (section 4.1) s'attache à évaluer les performances du composant impliqué dans la décision (APA) auquel nous avons plus spécifiquement contribué. Les principes d'implémentation de ce composant et de ses sous-composants sont évalués sous l'angle de plusieurs métriques, notamment liées au temps de service des requêtes faites à l'APA. L'objectif est d'évaluer la capacité de l'APA à résister au facteur d'échelle vis-à-vis du nombre de requêtes à traiter.

Dans le chapitre 3, nous nous sommes intéressés à la décision d'adaptation de la QoS au niveau Transport, par le biais d'une proposition de

gestion dite collaborative. Le problème de la décision, qui s'inscrit pleinement dans le rôle de l'APA, nous a amenés à définir un modèle d'optimisation permettant de répartir de manière coopérative les ressources en bande passante attribuées aux connexions Transport considérées. La seconde partie de ce chapitre (section 4.2) présente les mécanismes mis en œuvre pour implanter la décision prise grâce au modèle. Nous nous attachons également à montrer la pertinence et l'efficacité du modèle proposé dans différents scénarios de collaboration.

Finalement, nous concluons sur les évaluations effectuées, et nous en dégageons des perspectives pour conforter nos propositions.

4.1 Evaluation de performance de l'APA

L'Automated Policy Adaptor (APA) est l'entité centrale du système de contrôle de l'adaptation défini dans le projet NETQoS (Figure 4.1). Partant de politiques dynamiques stockées par les acteurs dans le répertoire POLD (via l'APM), le rôle de l'APA est de décider et de faire appliquer les politiques opérationnelles, en particulier au niveau Transport, pour permettre au système de communication de fournir une QoS optimisée. Pour cela, l'APA s'appuie sur le composant de monitoring et de mesures (MoMe), chargé d'observer et de mesurer l'évolution du contexte applicatif et réseau, ainsi que l'efficience des politiques opérationnelles. Les requêtes que reçoit l'APA proviennent directement du composant CM (Context Manager) du MoMe. Ces requêtes sont soient des requêtes de provisionnement de (nouvelles) politiques, soient des requêtes d'adaptation de politiques existantes lorsque celles-ci ne répondent pas aux objectifs de QoS déduites des politiques exprimées par par acteurs.

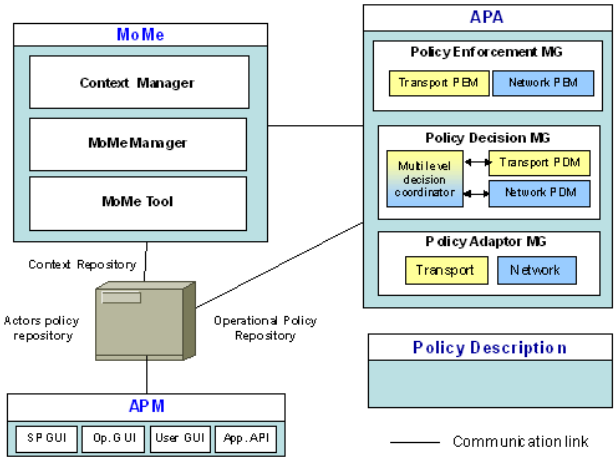


Figure 4.1 – *Les entités principales du système NETQoS*

Le but de cette section est d'évaluer la résistance de l'APA au facteur d'échelle vis-à-vis du nombre de requêtes de provisionnement et d'adaptation à traiter. Les mesures ont été effectuées sur une machine de type Dual Core Xeon 3050/2.13Ghz - 2GiB de RAM.

4.1.1 Spécification du contexte de mesure

L'APA est composé de trois composants principaux, le PDM (Policy Decision Manager), le PAM (Policy Adaptation Manager) et le PEM (Policy Enforcement Manager), dont l'implémentation JAVA est basée sur l'utilisation de "*Threads*" (mono and multi-threading).

Tel que présenté dans le chapitre 2, section 2.2.3.2.2, le PDM, qui est chargé de gérer les requêtes de provisionnement, est divisé en trois composants actifs. Dans les paragraphes suivants, nous rappelons les fonctionnalités de chacun de ces composants en précisant la nature des interaction en-

trantes, la nature des interactions sortantes et l'usage du multi-threading ou de files d'attentes dans leur implémentation.

Le PDMNotificationManager traite les évènements de type lancement d'application (ApplicationLaunch event) venant du Context Manager (CM), récupère les politiques intermédiaires correspondantes dans le répertoire (POLD) et les transmet au PDMWorkerManager.
– Il prend en entrée les évènements venant du Context Manager.
– Il retourne en sortie la politique intermédiaire à fournir.
– Il n'utilise pas le multi-threading : une file d'attente de longueur infinie est utilisée pour stocker les événements venant du Context Manager.

Le PDMWorkerManager prend la décision de provisionnement nécessaire et la transmet au PDMDispatcher.
– Il prend en entrée la politique intermédiaire à provisionnner.
– Il retourne en sortie la politique opérationnelle à déployer.
– Il utilise le multi-threading : un thread PDMWorker est ainsi créé pour chaque politique intermédiaire venant du PDMNotificationManager.

Le PDMDispatcher est chargé des interactions avec le PEM afin de déployer la politique opérationnelle. Il stocke également la politique dans le POLD, et rend la politique active dans le POLD.
– Il prend en entrée la politique opérationnelle à déployer.
– Il retourne en sortie cette même politique opérationnelle.
– Il n'utilise pas le multi-threading : une file d'attente de longueur infinie est utilisée pour stocker la politique opérationnelle venant du PDMWorkerManager.

De manière analogue au PDM, le gestionnaire de l'adaptation des politiques, le PAM, qui est chargé des requêtes d'adaptation, est divisé en trois composants actifs pour lesquels les mêmes choix ont été effectués concernant l'utilisation du multi-threading et les files d'attente.
– **Le PAMNotificationManager** traite les évènements de type violation de politique (PolicyViolation event) venant du Context Manager, récupère les politiques (intermédiaire et opérationnelle) dans le répertoire POLD, et les transmet au PAMWorkerManager.
– **Le PAMWorkerManager** prend les décisions d'adaptation nécessaires et les transmet au PAMDispatcher.
– **Le PAMDispatcher** est chargé des interactions avec le PEM afin de déployer la nouvelle politique opérationnelle, et de la stocker dans le POLD.

Le PEM est composé principalement du gestionnaire de l'application des politiques, appelé PEMEnforcementManager. En fonction des différents niveaux de politiques opérationnelles à déployer (niveau Transport, Réseau,...), le PEMEnforcementManager, délègue le déploiement de la politique pour chacun de ces niveaux vers le IPolicyEnforcer correspondant. Pour chaque niveau, par exemple le niveau Transport dans notre cas :
– il prend en entrée la politique opérationnelle à déployer ;
– il retourne en sortie la politique opérationnelle à déployer ;
– il n'utilise pas le multi-threading : une file d'attente de longueur infinie

est utilisée pour stocker la politique opérationnelle venant du PDMWorkerManager.

4.1.2 Métriques mesurées

Le comportement de l'APA est déclenché par la réception d'événements venant du Context Manager tel que l'évènement *ApplicationLaunchEvent* dans le cas d'un provisionnement et d'un événement *PolicyViolationEvent* lorsqu'une adaptation est requise.

Pour évaluer les performances de l'APA, deux métriques principales ont été observées pour différentes valeurs de taux d'arrivée des évènements *ApplicationLaunchEvent* et *PolicyViolationEvent* :
- le temps de service par requête (correspondant chacune à un événement) ;
- la longueur des files d'attente d'évènements pour les composants internes.

4.1.3 Scénarios de mesures, résultats et analyses

Deux types de scénarios ont été définis et réalisés :
- le premier vise à évaluer l'APA vis-à-vis des requêtes de provisionnement résultant de la réception des évènements *ApplicationLaunchEvent* ; dans ces scénarios, seuls le PDM et le PEM sont impliqués.
- le second vise à évaluer l'APA vis-à-vis des requêtes d'adaptation résultant de la réception des évènements PolicyViolation ; dans ces scénarios, seuls le PAM et le PEM sont impliqués.

4.1.3.1 Evaluation de l'APA vis-à-vis des requêtes de provisionnement

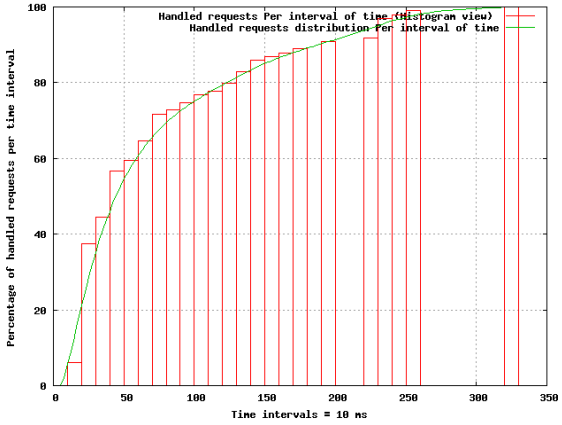
Cette section présente les performances de l'APA en termes de temps de traitement des requêtes et d'évolution des files d'attente internes. Le taux d'arrivée des requêtes *ApplicationLaunchEvent* varie de 20 évènements/s à 200 évènements/s.

Les courbes des figures 4.2, 4.3 et 4.4 montrent le temps de traitement des requêtes par le gestionnaire de décision (PDM) et le gestionnaire d'application (PEM) des politiques. Les courbes de la figure 4.5 présentent l'évolution des files d'attente des différents composants de l'APA qui sont impliqués dans le processus de provisionnement, c'est-à-dire le PDM Notification Manager, le PDM Dispatcher et le PEM.

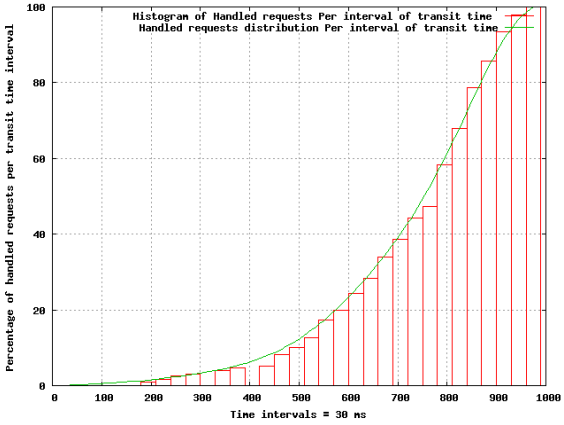
APA : Temps de traitement des requêtes de provisionnement

Trois types de courbes sont présentés :
- les courbes de la figure 4.2 présentent l'histogramme du pourcentage de requêtes servies par intervalle de temps, pour différents taux d'arrivée variant de 20 à 200 évènements/s ;
- les courbes de la figure 4.3 présentent le temps de transit des requêtes, pour différents taux d'arrivée variant de 20 à 200 évènements/s ;

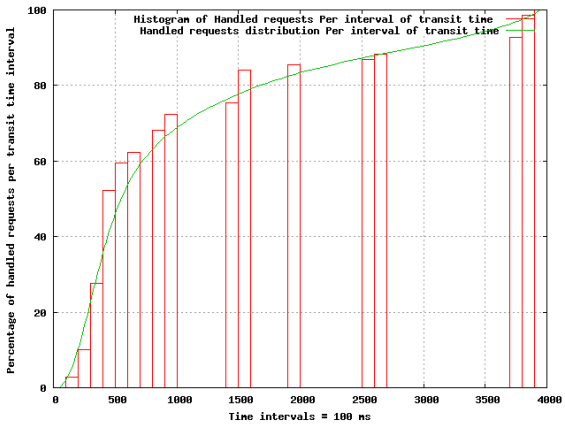
– les courbes de la figure 4.4 présentent la valeur moyenne du temps de transit d'une requête en fonction du taux d'arrivée variant de 20 à 200 évènements/s.



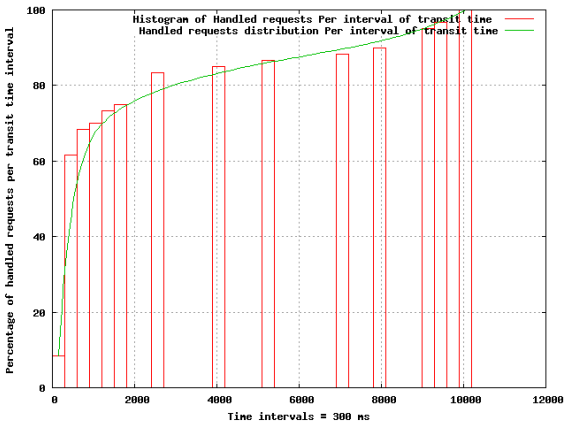
(a) $\lambda = 20$ *events/s*



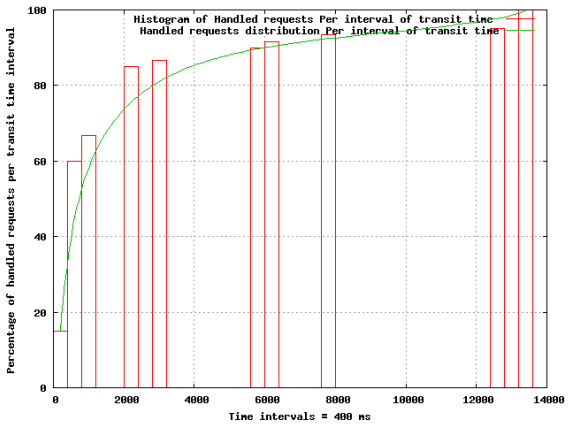
(b) $\lambda = 40$ *events/s*



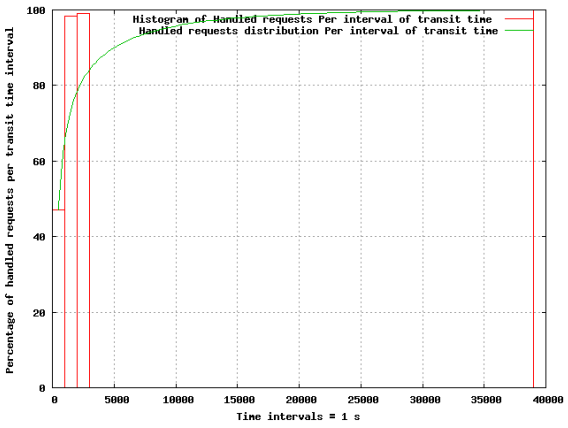
(c) $\lambda = 50$ *events/s*



(d) $\lambda = 70$ *events/s*



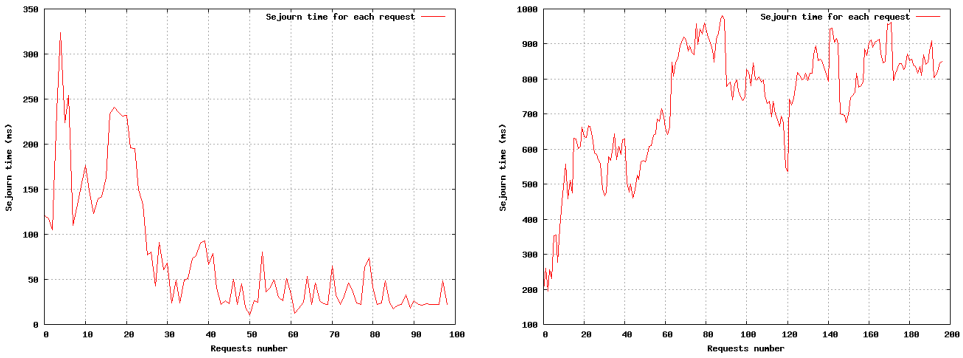
(e) $\lambda = 100$ *events/s*



(f) $\lambda = 200$ *events/s*

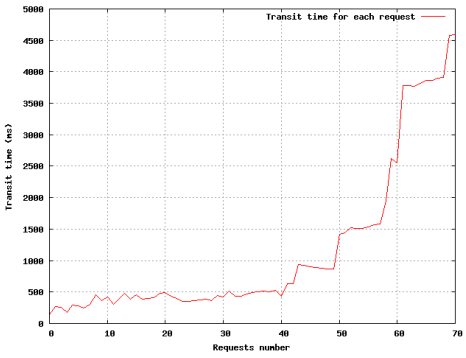
FIGURE 4.2 – *T-APA : Histogramme du % de requêtes de provisionnement servies par intervalle de temps*

APA : Evolution de la longueur des files d'attente

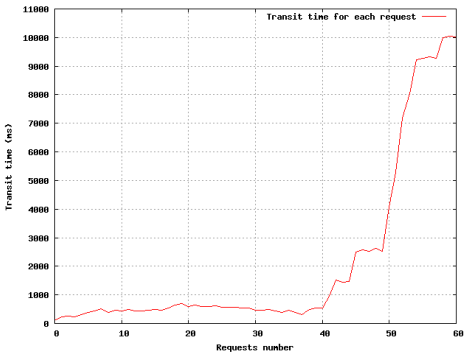


(a) $\lambda = 20$ *events/s*

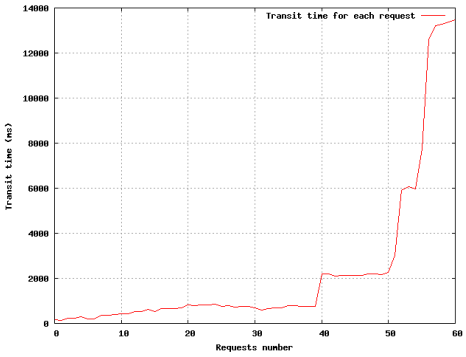
(b) $\lambda = 40$ *events/s*



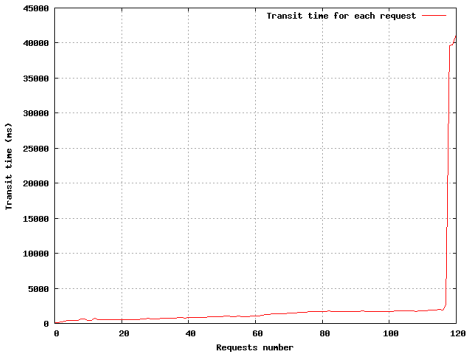
(c) $\lambda = 50$ *events/s*



(d) $\lambda = 70$ *events/s*



(e) $\lambda = 100$ *events/s*



(f) $\lambda = 200$ *events/s*

FIGURE 4.3 – *T-APA : Temps de transit pour une requête de provisionnement*

La figure suivante 4.5 montre l'évolution des différentes files d'attente gérées dans l'APA lorsqu'il doit traiter des requêtes de provisionnement.

Analyse

Deux comportements différents sont observés en fonction du taux d'arrivée des évènements *ApplicationLaunchEvent* :

– De 20 à 39 évènements/s, l'APA n'est pas chargé (Figure 4.2). Les meilleures performances sont obtenues et les files d'attentes restent vides pour tous les composants impliqués. Par exemple, pour un taux d'arrivée de 20 évènements/s, presque 60% des requêtes sont servies en moins de 50 ms qui est la valeur moyenne du temps inter-arrivées

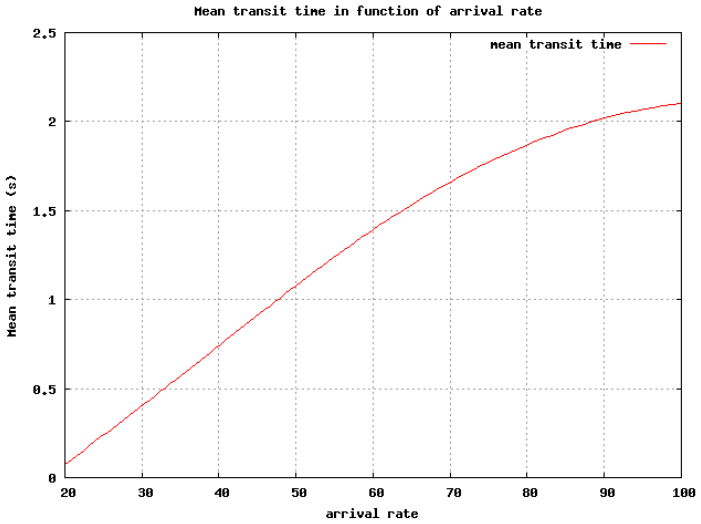


FIGURE 4.4 – *T-APA : Temps de transit moyen d'une requête de provisionnement vs. taux d'arrivée*

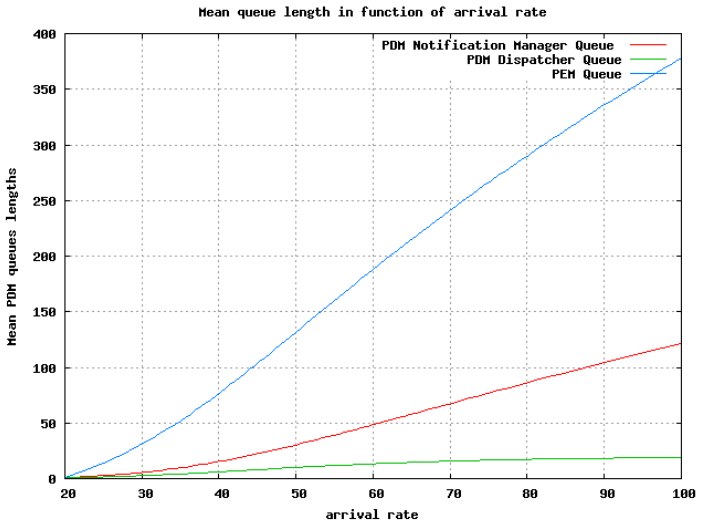


FIGURE 4.5 – *APA : Evolution de la longueur des files d'attente en fonction du temps*

des requêtes.

– Lorsque le taux d'arrivée dépasse les 40 requêtes/s, l'APA commence à être surchargé :
  - Ceci est montré par l'augmentation progressive des files d'attente interne (Figure 4.5).
  - On peut remarquer sur la figure 4.3 que l'APA peut résister à un nombre importants de requêtes par seconde seulement pendant une période courte (moins d'une seconde). Cela s'explique par l'utilisation massive du multi-threading dans le processus de décision (un PDMWorker est créé par requête). Après la création d'un nombre donné de threads, la JVM n'est pas capable de créer de nouveau threads et le système est bloqué jusqu'à ce que les anciens threads meurent.

Remarquons enfin que les histogrammes présentés sur la figure 4.3 et les temps moyens de transit des requêtes de provisionnement présentés figure 4.4 ne prennent pas en compte les valeurs mesurées après le blocage.

4.1.3.2 Evaluation de l'APA vis-à-vis des requêtes d'adaptation

Cette section présente les performances de l'APA vis-à-vis du temps de traitement des requêtes d'adaptation et de l'évolution des files d'attente internes. Le taux d'arrivée des requêtes de violation de politique *PolicyViolationEvent* varie toujours de 20 évènements/s à 200 évènements/s.

Les courbes des figures 4.6, 4.7 et 4.8 montrent (pour le PAM et le PEM) les temps de service moyen des requêtes. Les courbes de la figure 4.9 présente l'évolution de la file d'attente des différents composants de l'APA qui sont impliqués dans le processus d'adaptation, c'est-à-dire le PAM Notification Manager, le PAM Dispatcher et le PEM.

APA : Temps de traitement des requêtes d'adaptation

Trois types de courbes sont présentés :
- la figure 4.6 présente l'histogramme du pourcentage de requêtes servies par intervalle de temps, pour différents taux d'arrivée de requête ;
- la figure 4.7 présente le temps de transit des requêtes, pour différents taux d'arrivée de requête ;
- la figure 4.8 présente la valeur moyenne du temps de transit d'une requête en fonction du taux d'arrivée de requête.

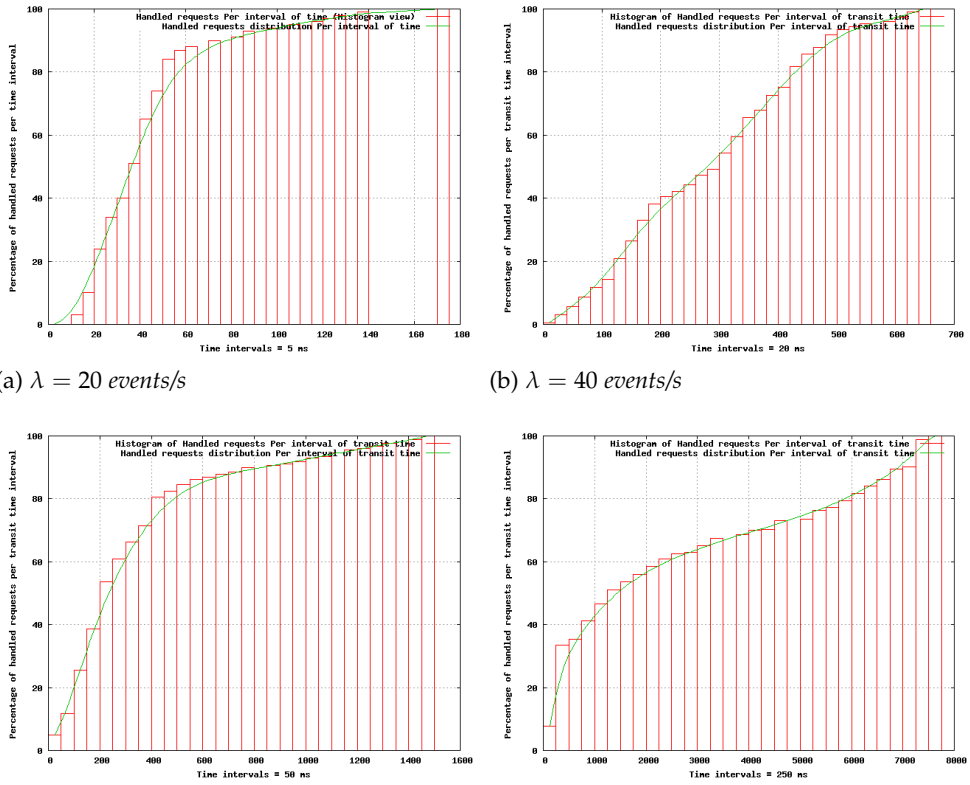
APA : Evolution de la longueur des files d'attente

La figure suivante 4.9 montre l'évolution des différentes files d'attente gérées dans l'APA lorsqu'il doit traiter des requêtes d'adaptation.

Analyse

Comme pour les mesures concernant les requêtes de provisionnement, deux comportements différents sont observés en fonction du taux d'arrivée des évènements *PolicyViolationEvent* :
- de 20 à 39 évènements/s, l'APA n'est pas chargé. Les meilleures performances sont obtenues et les files d'attentes restent vides pour tous les composants impliqués (Figure 4.9). Par exemple, pour un taux d'arrivée de 20 évènements/s, plus de 75% des requêtes sont servies en moins de 50 ms ;

- lorsque le taux d'arrivée dépasse les 40 requêtes/s (voir les figures avec un λ de 50, 70, 100, 200), l'APA commence à être surchargé :
  - ceci est montré par l'augmentation progressive des files d'attente interne (Figure 4.9) ;
  - de manière analogue aux mesures effectuées pour les requêtes de provisionnement, on peut remarquer sur la figure 4.7 que l'APA peut résister à l'arrivée d'un nombre important de requêtes par seconde pendant une période donnée puis que le délai de transit augmente très rapidement. Cela s'explique à nouveau par l'utilisation massive



(a) $\lambda = 20$ *events/s* (b) $\lambda = 40$ *events/s*

(c) $\lambda = 50$ *events/s* (d) $\lambda = 100$ *events/s*

FIGURE 4.6 – *APA : Histogramme du % de requêtes d'adaptation servies par intervalle de temps*

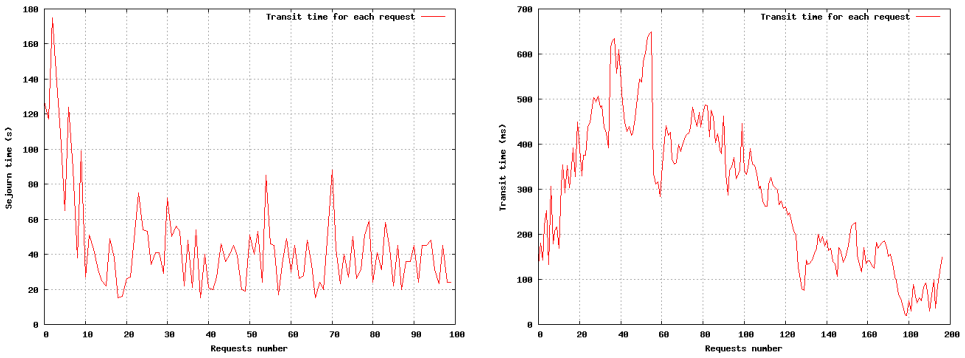
du multi-threading dans le processus de décision (un PAM worker créé par requête). Après la création d'un nombre donné de threads, la JVM n'est plus capable de créer de nouveaux threads et le système est bloqué jusqu'à ce que les anciens threads meurent.

Remarquons enfin que les histogrammes présentés sur la figure 4.7 et les temps moyens de transit des requêtes d'adaptation présentés figure 4.8 ne prennent pas en compte les valeurs mesurées après le blocage.

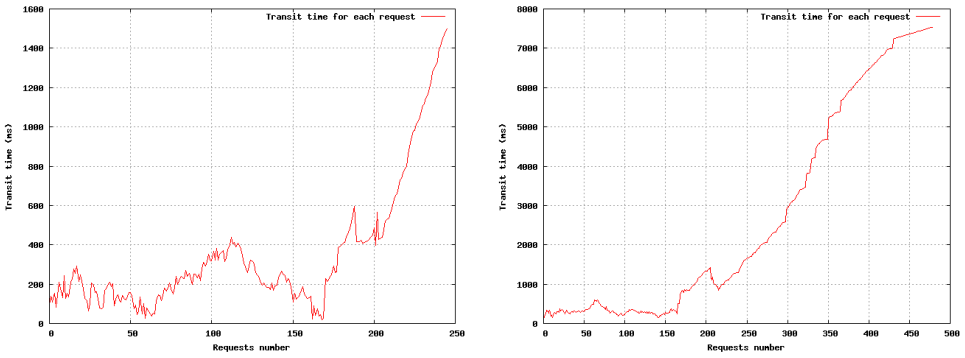
4.1.4 Conclusions

Cette section a présenté une campagne de mesures visant à évaluer les choix d'implémentation du composant de décision proposé dans le projet NETQoS (l'APA). L'objectif était d'évaluer la résistance au facteur d'échelle de l'APA face aux requêtes de provisionnement et d'adaptation susceptibles d'être reçues.

Plusieurs métriques liées au temps de service de l'APA ont été analysées afin de valider en particulier l'usage proposé du multi-threading. Les mesures présentées font apparaître que l'APA est capable de traiter dans un délai acceptable un nombre donné (de l'ordre de 40) de requêtes de provisionnement ou adaptation par seconde. Au delà, l'APA est toujours capable de traiter les requêtes mais seulement pendant une très



(a) $\lambda = 20$ *events/s* (b) $\lambda = 40$ *events/s*



(c) $\lambda = 50$ *events/s* (d) $\lambda = 100$ *events/s*

FIGURE 4.7 – *PAM : Temps de transit pour une requête d'adaptation*

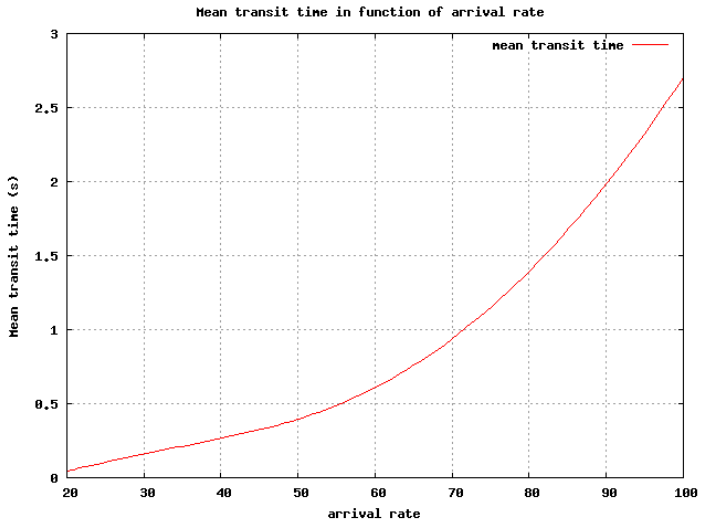


FIGURE 4.8 – *APA : Temps de transit moyen pour une requête d'adaptation vs. taux d'arrivée*

courte période. Ce phénomène découle d'une utilisation massive du multi-threading, qui conduit au blocage du système en raison des limitations de la JVM. Une solution à étudier pourrait être de mettre en place une solution hybride de mono et multi-threading, pour maintenir le nombre de threads PDM/PAM Worker créés sous une valeur donnée.

Dans le cadre d'une activité de groupe telle que celles ciblées par nos

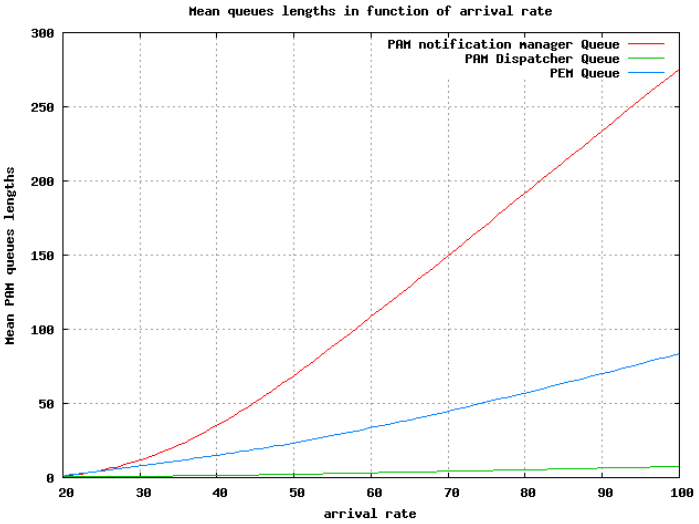


FIGURE 4.9 – *T-APA : Evolution de la longueur des files d'attente*

travaux, bien que nous n'ayons pas mené l'étude de façon formelle, le nombre restreint de participants et la nature fermée de l'environnement réseau permettent d'envisager des performances acceptables de l'APA vis-à-vis du nombre de requêtes de provisionnement ou d'adaptation.

La suite de ce chapitre présente les résultats de la campagne d'évaluation de performance attenante à notre proposition de gestion coopérative de la QoS.

4.2 Gestion coopérative de la QoS et répartition de débit

La première partie de ce chapitre a permis d'évaluer les principes d'implémentation d'un des composants du système de gestion de la QoS dans les réseaux autonomes, l'APA, tel que proposé dans projet NETQoS. Comme nous l'avons montré au chapitre 3, c'est par ce composant que la décision de gestion collaborative peut être prise, grâce à l'implémentation du modèle d'optimisation permettant d'augmenter la QoS d'une connexion au détriment d'une ou plusieurs autres connexions.

Dans la suite de ce chapitre, nous cherchons à évaluer la pertinence et l'efficacité du mécanisme de gestion collaborative décrit au chapitre 3. Pour cela, nous présentons d'abord deux mises en œuvre possibles de l'approche de gestion de collaboration proposée au niveau Transport. Ensuite, nous présentons la campagne d'évaluation organisée en deux parties : la première est une étude de faisabilité menée dans un contexte limité ; la deuxième s'appuie sur une topologie plus complexe en termes de nœuds et de connexions, et s'appuie sur la méthode d'optimisation proposée au chapitre 3.

4.2.1 Contrôle de congestion utilisé : TFRC

Comme explicité en section 3.1.3 du chapitre 3, notre approche de répartition de la QoS s'appuie sur l'hypothèse de la mise en œuvre d'un mécanisme de contrôle de congestion, qui de manière classique évite l'effondrement du réseau et permet d'envisager une relative stabilité des débits des connexions sur des périodes de temps significativement longues.

Le contrôle de congestion que nous avons retenu pour mettre en œuvre la coopération des connexions de Transport est le mécanisme TFRC pour les raisons suivantes :

- TFRC est un mécanisme de contrôle de débit à l'initiative de l'émetteur basé sur un modèle. Il a été conçu pour minimiser les variations brusques de débit afin de répondre aux besoins des applications multimédias. Il permet également de respecter les flux TCP vis-à-vis de l'utilisation de la bande passante ;
- TFRC est un mécanisme de contrôle de congestion basé débit qui offre une estimation de la bande passante pour les différentes connexions que nous allons pouvoir utiliser.

L'émetteur ajuste le débit d'émission des paquets en fonction de rapports (appelés feedback) envoyés par le récepteur. En fonction des informations contenues dans un feedback, l'émetteur calcule le débit d'émission. Dans l'algorithme du mécanisme TFRC, la bande passante pour une connexion est estimée explicitement par la formule (4.1) suivante :

$$TFRC = \frac{s}{R\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \tag{4.1}$$

avec $TFRC$ le débit en octets/s, s la taille d'un paquet en octets, R le RTT en secondes, t_{RTO} la valeur du timer de retransmission en secondes, $p \in [0..1]$ le taux d'évènements de pertes calculé par l'émetteur, b le nombre maximal de paquets acquittés par un acquittement TCP.

4.2.2 Mise en œuvre de l'approche

L'approche de gestion collaborative proposée dans le chapitre 3 est basée sur la capacité des entités de Transport à effectuer des incréments $+\delta$ (pour les connexions receveuses) ou des décréments $-\Delta$ (pour les connexions donneuses) sur leur débit d'émission.

Nous avons testé deux approches pour aboutir à ces incréments et décréments.

La première approche repose sur un algorithme consistant à modifier le débit obtenu par la formule (4.1) en ajoutant ou retranchant les quantités correspondantes. Ainsi, pour une connexion receveuse p_k, le débit calculé par le mécanisme TFRC, donné par l'équation 4.1 est noté $p_k.TFRC$. Ce débit est ensuite augmenté de la quantité $p_k.\delta$ requise par la connexion. De la même manière, une connexion donneuse d_k dont le débit est noté $d_k.TFRC$, sera diminué de la quantité $d_k.\Delta$.

Le débit des connexions est donc donné par :

$p_k.TFRC + p_k.\delta$ où p_k est une connexion receveuse
$d_k.TFRC - d_k.\Delta$ où d_k est une connexion donneuse.

Dans la suite, nous désignons par *contrôle de congestion collaboratif* ce nouveau mécanisme. La sections 4.2.3 suivante et la première partie de la section 4.2.4 présentent les résultats de mesures attenant à la mise en œuvre de ce mécanisme.

La deuxième approche testée pour pour aboutir à des incréments/décréments sur des connexions receveuses/donneuses, ne repose pas sur une modification de TFRC mais sur la mise en œuvre d'un contrôle de débit (*Shaper*) au-dessus de TFRC avec pour valeur seuil les valeurs obtenues grâce à notre modèle d'optimisation. Les résultats de la mise en œuvre de cette approche sont présentés en section 4.2.4.3.

4.2.3 Etude de faisabilité - Scénarios simples

Afin d'illustrer la mise en œuvre du contrôle de congestion collaboratif, nous considérons plusieurs scénarios mettant en jeu uniquement deux connexions « qui collaborent » en présence d'autres connexions « non collaborantes » car elles ne participent pas à l'activité. On y considère en particulier une connexion receveuse qui requiert un incrément de δ et une connexion donneuse sur laquelle on applique un décrément de $-\Delta$, de manière à ce que la collaboration se réalise.

Ces scénarios simples ne requièrent pas l'utilisation de la méthode d'optimisation proposée dans le chapitre 3.

Les scénarios ont été conçus pour étudier deux points particuliers :
– l'influence de la localisation des interfaces chargées que nous appellerons dans cette partie "*goulots d'étranglement*" ;
– l'absence d'effet perturbateur sur les connexions ne participant pas à l'activité collaborative.
Enfin, afin d'éviter les problèmes liés à la dynamique de TFRC, les périodes de simulations sont volontairement très importantes.

4.2.3.1 Présentation des scénarios

Nous avons considérés trois scénarios :

Dans le premier scénario, les deux connexions qui collaborent (une connexion receveuse et une connexion donneuse) sont exécutées de façon à ce qu'elles suivent la même route à travers le réseau. Afin d'assurer cette propriété, nous attribuons à ces connexions la même source ainsi que la même destination, et nous supposons un routage statique.

Le second scénario, qui vise à étendre le domaine de validité de notre approche, considère deux sources de trafic distinctes émettant à destination d'une même machine. Dans ce scénario, nous veillons à ce que le « goulot d'étranglement » (bottleneck) pour ces connexions soit situé sur la portion commune de leurs routes respectives.

Enfin, le troisième scénario permet de mettre en évidence les limites de validité du mécanisme proposé. En effet, dans ce scénario, les deux sources collaborantes (la connexion donneuse et la connexion receveuse) ne partagent plus le même « goulot d'étranglement ». L'effet du mécanisme sur les autres connexions présentes est étudié permettant ainsi de tirer les conclusions quant aux hypothèses de bon fonctionnement du mécanisme.

Dans chacun des scénarios, les routeurs constituant les goulots d'étranglement sont représentés par un cercle avec un point noir au centre.

4.2.3.2 Résultats et Analyse

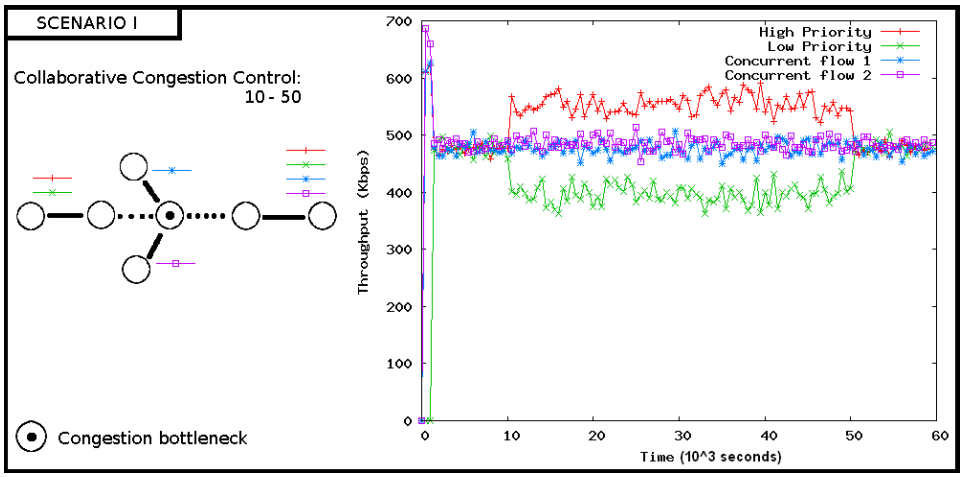


FIGURE 4.10 – *Scénario I : Flux suivant la même route*

Scénario 1. Les résultats obtenus pour le premier scénario sont présentés sur la Figure 4.10. Dans ce scénario, les deux connexions collaborantes partagent la même route à travers le réseau. Elles sont en concurrence avec d'autres connexions de telle sorte que l'un des routeurs situées sur le chemin de données soit en état de congestion.

Cette première étude vise à montrer dans un premier temps la faisabilité de l'approche. Si la topologie utilisée est spécifique, on peut néanmoins remarquer que beaucoup d'applications existantes, telles que la vidéo-conférence, présentent une configuration analogue (flux audio et vidéo à priorités relatives).

Sur la courbe Figure 4.10, on peut observer que chacune des quatre connexions en concurrence obtient en régime stationnaire une part de la bande passante totale disponible de 490Kbps environ entre les instants 3 et 10. La période collaborative intervient entre les instants 10 et 50. Dans cet intervalle, il est remarquable que la connexion receveuse obtient plus de ressources (550Kbps) que la connexion donneuse (430Kbps). De plus, il apparait que cette collaboration n'a aucune incidence sur les deux autres connexions présentes, qui conservent chacune une part de bande passante de 490Kbps comme précédemment. Enfin, une fois la période de collaboration terminée, l'algorithme standard du contrôle de congestion permet à nouveau l'obtention d'un débit d'environ 490Kbps pour chaque connexion.

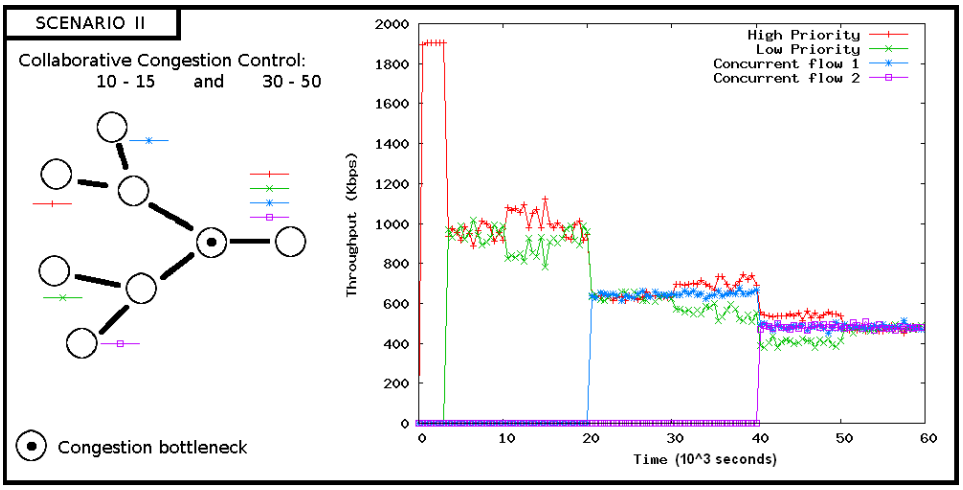


FIGURE 4.11 – *Scénario II : Flux de sources distinctes*

Scénario 2. Les résultats obtenus pour le second scénario sont présentés sur la Figure 4.11. Dans ce scénario, les deux connexions collaborantes proviennent de sources différentes. Elles sont en concurrence avec d'autres connexions de telle sorte que l'un des routeurs situés sur la partie commune du chemin de données soit en état de congestion.

Une première partie de l'étude entre les instants 10 et 15 sur la courbe Figure 4.11 vise à montrer, qu'en l'absence de flux concurrents, le contrôle de congestion collaboratif est possible entre deux flux provenant de sources différentes et partageant un routeur congestionné. Dans un second temps, un premier flux non collaboratif est introduit ; la présence d'une période de collaboration entre les instants 30 et 40 n'a aucun effet sur le flux non collaboratif qui obtient toujours une part équitable de la bande passante disponible. Enfin, l'arrivée d'un nouveau flux pendant la période de collaboration à l'instant 40 a pour effet de réduire le débit de toutes les connexions présentes. On peut noter que les connexions non collaboratives ont un débit inchangé pendant l'intervalle 40-50. Enfin, les connexions collaboratives réagissent à l'arrivée du nouveau flux dans le système en réduisant leur débit de façon à laisser place au nouvel arrivant ; elles conservent cependant les priorités propres à leur collaboration.

Scénario 3. Les résultats obtenus pour le troisième scénario sont présentés sur la Figure 4.12. Dans ce scénario, les deux connexions collaborantes proviennent de sources différentes et sont en concurrence avec d'autres connexions de telle sorte que les routeurs congestionnés ne soient à présent plus situés sur la partie commune du chemin de données.

Sur le courbe présentée Figure 4.12, un premier intervalle de collaboration entre les instants 10 et 15 rappelle les résultats obtenus précédemment lorsque le « goulot d'étranglement » est partagé. Néanmoins, l'arrivée d'un premier flux non collaboratif à l'instant 20 a pour effet de déplacer le « goulot d'étranglement » de la partie haute du réseau. Ainsi, les connexions collaboratives ne partagent plus de « goulot d'étranglement » après l'instant 20. La phase de collaboration entre les instants 30 et 50 permet de voir, dans un premier temps, que la connexion non collaborante est affectée par le comportement agressif de la connexion de forte priorité.

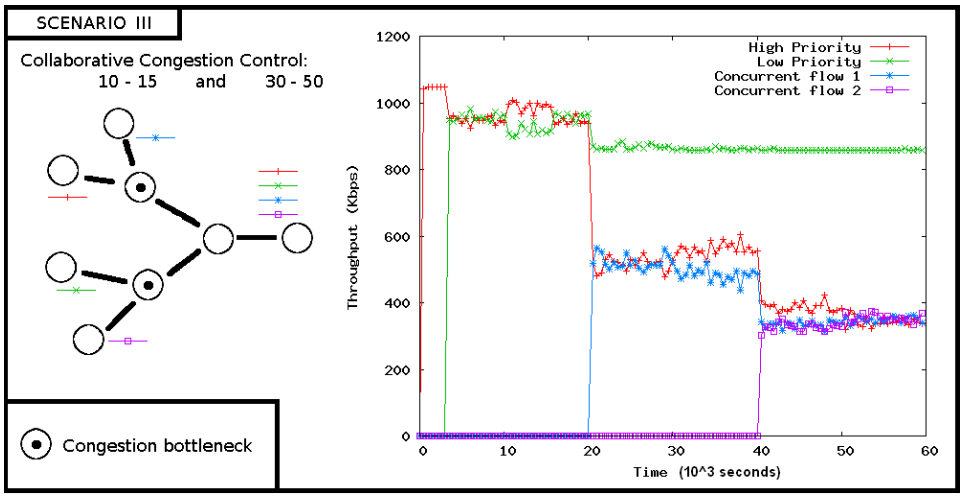


FIGURE 4.12 – *Scénario III : Flux de sources distinctes, bottleneck distincts*

De ce fait, le partage des ressources n'est plus équitable. De plus, l'arrivée d'une nouvelle connexion non collaborante partageant le même « goulot d'étranglement » montre que le comportement agressif de la connexion de forte priorité affecte toutes les autres connexions.

4.2.3.3 Conclusion

Ces trois scénarios montrent la faisabilité du mécanisme sur des périodes de collaboration longues, sans dépasser toutefois la dizaine d'heures. Cette échelle de temps permet d'illustrer le processus de coopération en faisant abstraction des fluctuations à des dynamiques temporelles plus élevées.

Les goulots d'étranglement permettent d'identifier les connexions qui doivent collaborer. Il ressort clairement que le mécanisme de collaboration peut être effectif sans perturber les connexions non collaborantes.

Dans la section suivante, nous étudions le phénomène de coopération sur des échelles de temps plus courtes, et sur une topologie de connexions plus complexe, en appliquant la méthode de calcul optimisé des décréments proposée dans le chapitre 3.

4.2.4 Etude de faisabilité - Scénarios plus complexes

Les simulations présentées dans cette section différent des précédentes en plusieurs points.

D'abord, la topologie réseau est plus complexe : elle présente un nombre de nœuds plus important ; nous considérons toujours une unique connexion receveuse mais plusieurs connexions potentiellement donneuses.

Ensuite, nous ne nous intéressons pas ici à l'influence de la coopération sur les connexions non collaborantes ou à la localisation des points de congestion. Il s'agit en effet de vérifier le comportement du contrôle de congestion collaboratif sur des échelles de temps courtes lorsqu'on injecte

les incréments et décréments donnés par le modèle d'optimisation pour plusieurs connexions donneuses.

Enfin, nous cherchons à expérimenter les deux méthodes proposées pour mise en œuvre du contrôle de congestion collaboratif :
- soit par la modification du mécanisme TFRC pour prendre en compte la possibilité d'augmenter ou de diminuer le débit de la quantité voulue, c'est-à-dire la même modification que pour les simulations précédentes sur des périodes de temps plus longues ;
- soit utilisation d'un mécanisme TFRC non modifié mais combiné à un mécanisme de contrôle de débit, de type *Shaper*.

Nous présentons d'abord le scénario avec la topologie réseau et les connexions mises en jeu, puis nous présentons les résultats obtenus.

4.2.4.1 Présentation du scénario

La topologie réseau utilisée en simulation est présentée figure 4.13. Le lien entre les nœuds n_2 et n_3 a une capacité de 2.4 Mbits/s. La capacité est de 2 Mbits/s pour tous les autres liens.

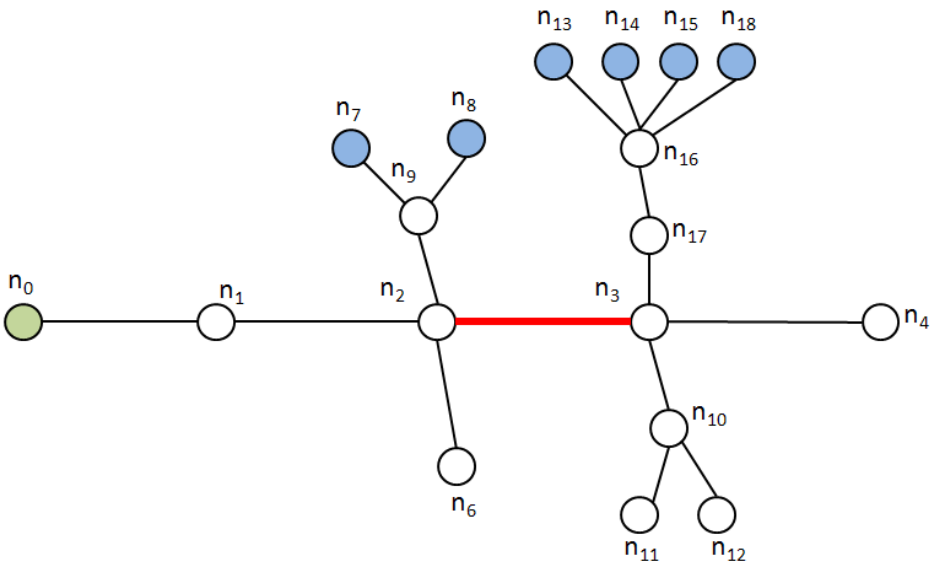


FIGURE 4.13 – *Topologie réseau de la simulation ns-2*

On considère sept connexions dont les caractéristiques sont données pour chacune par : le type de la connexion (donneuse ou receveuse), l'identité des nœuds source ou destination), la route empruntée et le débit moyen requis. Toutes les sources sont supposées émettre à un débit constant.

Connexion p_1
type : receveuse
source : n_0
destination : n_4
route : n_1, n_2, $n3$
débit moyen : 1.2 Mbits/s

Connexion d_2
type : donneuse
source : n_7
destination : n_{11}
route : n_9, n_2, n_3, n_{10}
débit moyen : 0.8 Mbits/s

Connexion d_3
type : donneuse
source : n_8
destination : n_{12}
route : n_9, n_2, n_3, n_{10}
débit moyen : 0.8 Mbits/s

Connexion d_4
type : donneuse
source : n_{13}
destination : n_4
route : n_{16}, n_{17}, n_3
débit moyen : 0.1 Mbits/s

Connexion d_5
type : donneuse
source : n_{14}
destination : n_4
route : n_{16}, n_{17}, n_3
débit moyen : 0.1 Mbits/s

Connexion d_6
type : donneuse
source : n_{15}
destination : n_4
route : n_{16}, n_{17}, n_3
débit moyen : 0.1 Mbits/s

Connexion d_7
type : donneuse
source : n_{18}
destination : n_4
route : n_{16}, n_{17}, n_3
débit moyen : 0.1 Mbits/s

D'après les débits des différentes connexions donnés ci-dessus et les capacités de chaque lien, on constate aisément que la seule interface chargée est l'interface du routeur allant de n_2 à n_3. Celle-ci a en effet une capacité de 2,4 Mbits/s mais doit écouler un total de 2,8 Mbits résultant de la traversée des paquets :

– de deux connexions donneuses, d_2 et d_3, qui obtiennent chacune un débit moyen de 0.8 Mbits/s correspondant au débit requis ;
– de la connexion receveuse (p_1), qui obtient un débit de 0.8 Mbits/s (pour 1,2 Mbits/s requis).

On se place donc dans l'analyse qui suit dans le cas où la connexion receveuse (p_1) réclame un incrément de $p_1.\delta = 0.4 Mbits/s$

4.2.4.2 Résultats et Analyse

Dans le scénario considéré, l'algorithme d'optimisation présenté dans le chapitre 3 produit les deux solutions triviales suivantes :

$d_2 = 0.4\,;\, d_3 = 0$

$d_3 = 0.4\,;\, d_2 = 0$

ce qui signifie que les quantités de débit que les connexions donneuses d_2 et d_3 vont devoir céder sont 0 et 0.4, ou inversement.

En d'autres termes, soit la connexion d_2 diminue son débit de 0.4 Mbits/s et la connexion d_3 garde son débit courant, soit la connexion d_3 diminue son débit de 0.4 Mbits/s et la connexion d_2 garde son débit courant.

La figure 4.14 présente les résultats obtenus pour la mise en œuvre de la solution, $d_2 = 0.4$; $d_3 = 0$. Le temps en seconde est représenté en abscisse et le débit en Mbits/s est représenté en ordonnée.

La collaboration ne concernant que les connexions p_1, d_2 et d_3, les débits des autres connexions ne sont pas représentés.

Jusqu'à l'instant 400, on constate que les trois flux en concurrence se partagent équitablement la bande passante, avec un débit de 0.8 Mbits/s. A partir de l'instant 400, la collaboration débute. La connexion receveuse p_1 voit son débit augmenter jusqu'à une valeur proche de 1.2 Mbits/s, tandis que le débit de la connexion d_2 est diminué de 0.4 Mbits/s. La connexion d_3 obtient toujours une bande passante de 0.8 Mbits/s. La collaboration cesse à l'instant 600 et chacune des trois connexions retrouve un débit équitable de 0.8 Mbits/s.

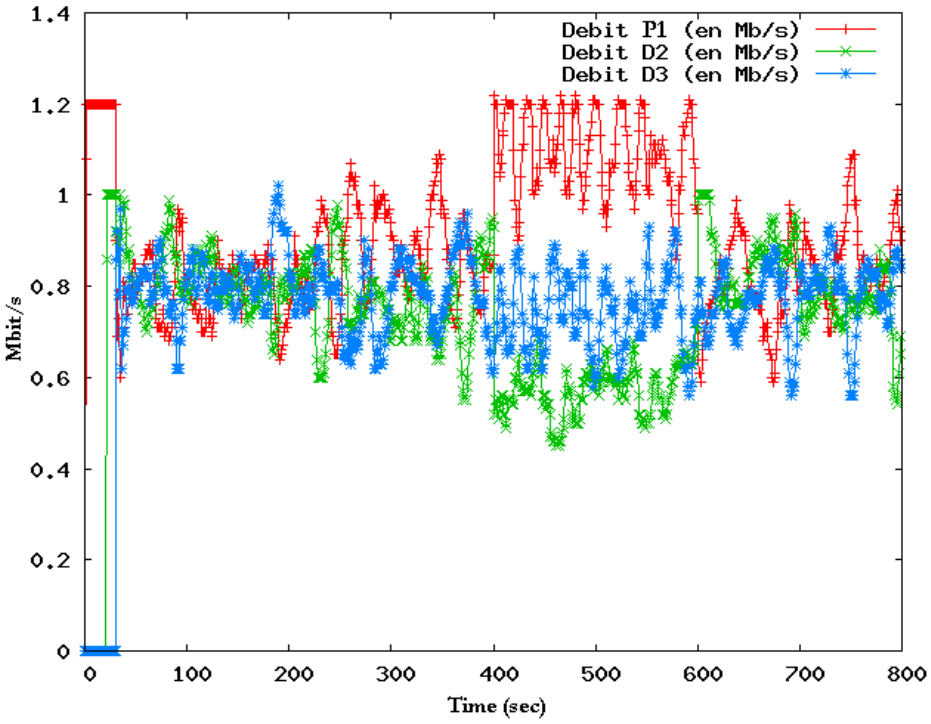


FIGURE 4.14 – *Débit TFRC modifié*

Les courbes présentées sur la figure 4.15 permettent de conforter le résultat précédent. Elles résultent du même scénario expérimental mais pour lequel les débits moyens obtenus ont été lissés en effectuant une moyenne à fenêtre glissante afin de supprimer les variations brusques des débits qui ne sont pas représentatives.

Les deux courbes précédentes (figures 4.14 et 4.15) montrent que le mécanisme de contrôle de congestion collaboratif fonctionne sur des périodes de temps plus restreintes que celles présentées dans la section 4.2.3.

Cependant, deux phénomènes sont à remarquer. D'une part, l'augmentation de débit obtenue par la connexion receveuse est un peu en-deçà de 0.4 Mbits/s en moyenne sur la période de collaboration entre les instants 200 et 400. D'autre part, la connexion d_2 subit une diminution elle aussi

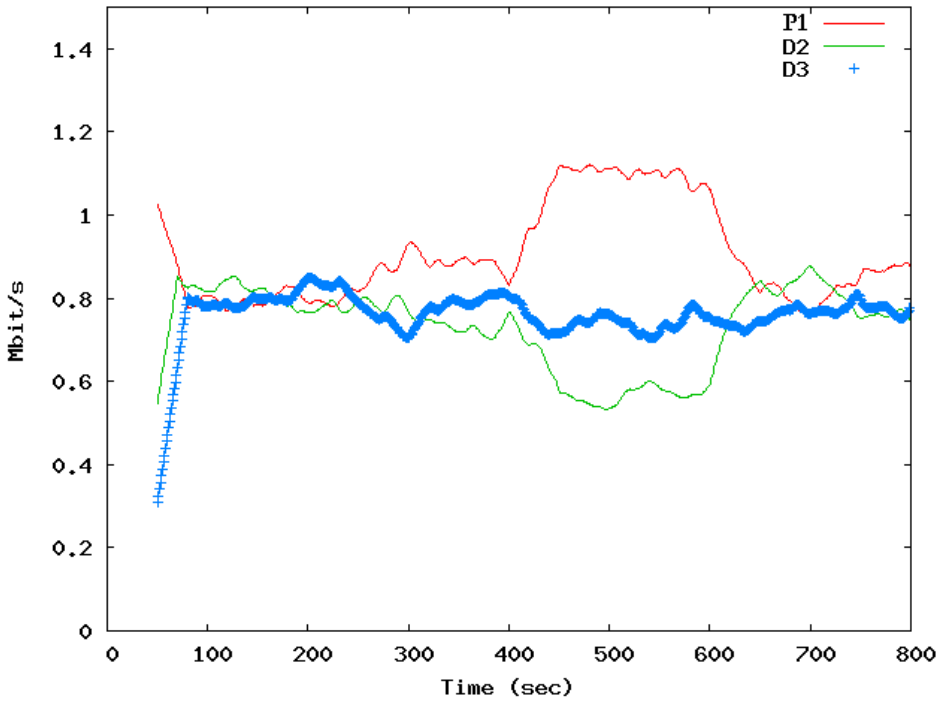


FIGURE 4.15 – *Débit TFRC modifié et lissé*

inférieure aux 0.4 Mbits/s attendus et la connexion d_3, qui devrait continuer d'obtenir un débit de 0.8 Mbits par seconde, semble subir une légère diminution.

Une explication de ces comportements réside sans doute dans la dynamique du mécanisme TFRC. En effet, si TFRC permet de limiter les variations trop brusques de débit résultant de l'application d'un mécanisme orienté fenêtre, il est en contre-partie sujet à des temps de convergence parfois importants. Lorsque le débit d'une connexion s'écarte du débit auquel elle peut normalement prétendre, il peut s'écouler un temps relativement long de plusieurs dizaines de secondes avant de retrouver le débit attendu. Ce phénomène permet d'expliquer les résultats obtenus en se basant sur TFRC.

Les résultats présentés sur les figures 4.16 et 4.17 illustrent un scénario de collaboration différent.

Dans le chapitre 3, nous avons évoqué l'intérêt d'obtenir l'ensemble des sommets solutions du problème d'optimisation permettant de trouver les quantités Δ des connexions donneuses. En effet, cela pourrait permettre dans certains cas d'obtenir, à partir de ces sommets, un ensemble plus important de solutions. Dans le scénario précédent, les sommets solution du polyèdre sont $d_2 = 0.4\,;\ d_3 = 0$ et $d_3 = 0.4\,;\ d_2 = 0$. Cependant, on constate que tous les points de l'arête du polyèdre formée par ces deux points sont également solution. Ainsi, la paire "équilibrée" $d_2 = 0.2\,;\ d_3 = 0.2$ est solution.

La période de collaboration se situe cette fois-ci entre les instants 100 et 200. On constate, comme dans les résultats précédents, une augmentation significative du débit de la connexion receveuse c_1 de presque 0.4 Mbits/s.

Les deux connexions receveuses voient chacune leur débit diminuer de 0.2 Mbits/s pour se rapprocher d'un débit de 0.6 Mbits/s.

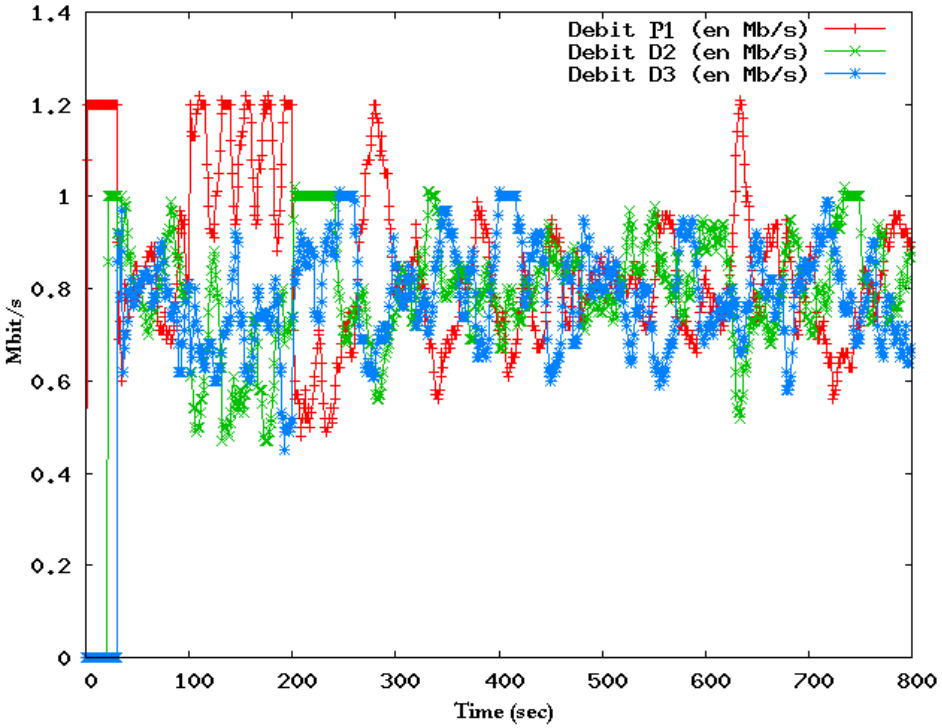


FIGURE 4.16 – *Débit TFRC modifié - version 2*

Les courbes de la figure 4.17, où les débits ont été lissés, confirment ces résultats.

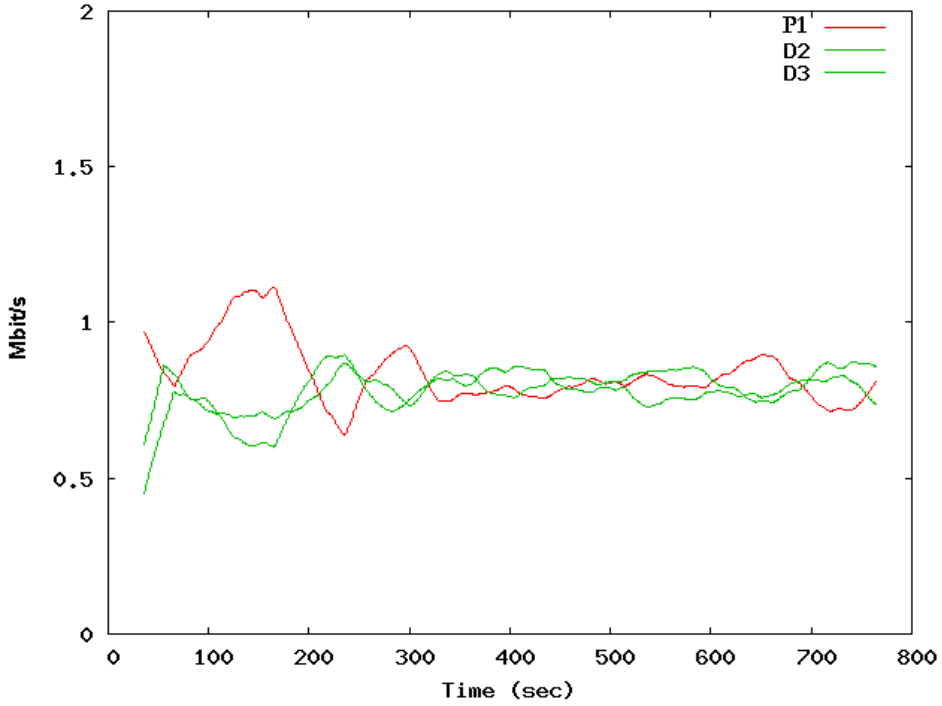


FIGURE 4.17 – *Débit TFRC modifié et lissé - version 2*

Le scénario de collaboration précédent mettant en œuvre une solution

équilibrée a montré des résultats probants, ce qui tend a montrer une certaine flexibilité dans l'application du modèle d'optimisation présenté dans le chapitre 3. Dans la perspective de l'utilisation de priorités, notamment entre les connexions donneuses, on constate que l'approche peut être étendue en disposant de toutes les solutions du problème d'optimisation. On pourrait alors envisager que les connexions puissent collaborer avec des poids différents, indiquant un choix particulier dans l'ensemble de solutions.

4.2.4.3 Perspectives : Shaper sur TFRC

Nous avons montré que la modification consistant à ajouter où retrancher une quantité $+\delta$ ou $-\Delta$ au débit calculé selon l'équation (4.1) par le mécanisme TFRC permettait de faire collaborer les connexions Transport. Cependant, les temps de convergence parfois longs de TFRC peuvent entrainer des instabilités et une inefficacité du mécanisme de collaboration.

Pour répondre à ce problème, une approche consistant à intégrer le calcul de la collaboration dans l'équation de TFRC peut être envisagée. Une autre approche, plus simple à mettre en œuvre et que nous avons suivie, consiste à utiliser un *Shaper*. Il faut pour cela contrôler le débit provenant de l'application en lui imposant le débit que l'on souhaite voir attribuer à la connexion pendant la période de collaboration.

Dans la perspective d'une implémentation utilisant un protocole de Transport configurable basé sur une architecture modulaire (tel qu'ETP), on peut envisager la mise en œuvre de cette approche par une composition de modules telle que la composition d'un module ***Shaper*** au-dessus d'un module ***TFRC***. Pour cela le mécanisme TFRC à utiliser est un TFRC classique et non un contrôle de congestion collaboratif.

De cette façon, on laisse le mécanisme TFRC classique en charge de la régulation des débits lorsque la collaboration des connexions n'est pas nécessaire et on utilise une mise en forme du trafic par le biais d'un Shaper permettant de limiter le débit des connexions au débit requis par la collaboration.

Les courbes de la figure 4.18 montrent les résultats issus d'une simulation de la mise en œuvre d'un Shaper combiné à TFRC.

Afin de diversifier les scénarios de collaboration, nous nous intéressons ici à un scénario où la connexion receveuse demande un incrément de $\delta = 0.2 Mbits/s$ et la période de collaboration s'étend entre les instants 100 et 200.

Pour les connexions donneuses d_2 et d_3, la résolution du problème d'optimisation donne $d_2 = 0.2\,;\, d_3 = 0$.

Jusqu'à l'instant 100, les trois connexions se partagent équitablement la bande passante avec un débit oscillant autour de 0.8 Mbits/s. Entre les instants 100 et 200, qui correspond à la période de collaboration, le débit des connexions donneuses est mis en forme. La connexion d_2 est limitée à un débit de 0.6 Mbits/s et la connexion d_3 est limitée à un débit

de 0.8 Mbits/s. On constate alors que le débit de la connexion receveuse augmente de 0.2 Mbits/s et atteint 1 Mbits/s. La collaboration est donc effective. A partir de l'instant 200, on stoppe la mise en forme du trafic et le mécanisme TFRC conduit à nouveau à une répartition des débits autour de 0.8 Mbits/s.

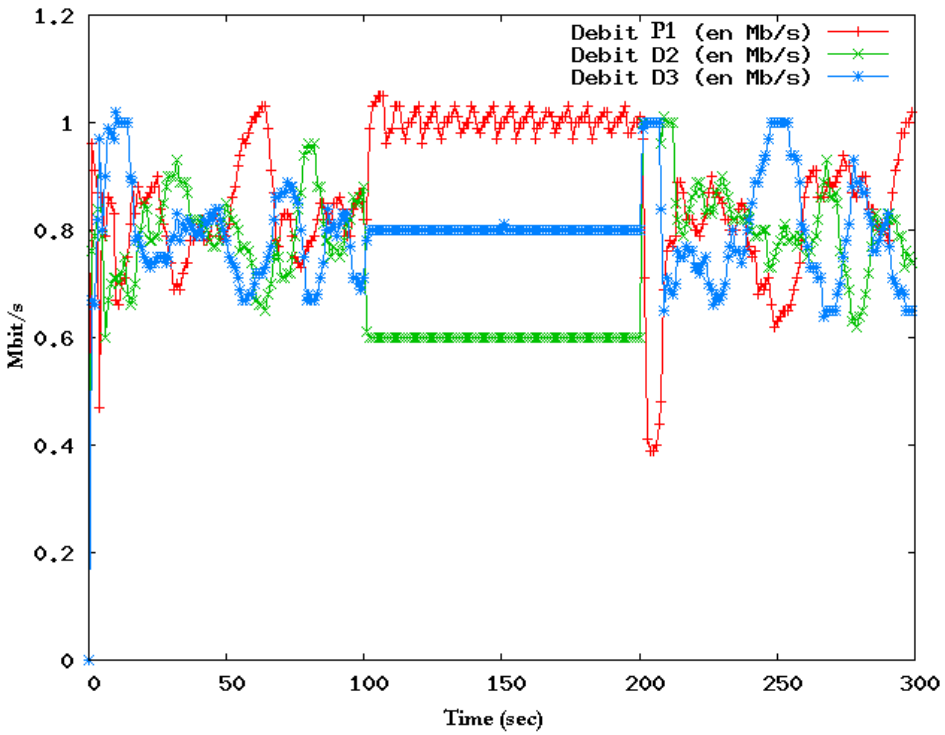


FIGURE 4.18 – *Débit TFRC + Shaper avec connexions donneuses non contraintes*

CONCLUSION DU CHAPITRE

La première partie de ce chapitre a présenté l'évaluation des performances du composant APA de gestion de l'adaptation du système de gestion adaptative de la QoS que nous avons élaboré dans le projet NETQoS. Les résultats, qui concernent les métriques liées au temps de service de l'APA, ont montré la pertinence des choix d'implémentation de ce composant pour un nombre acceptable de requêtes de service. Dans un contexte d'activités de groupe tel que celui des OIU, le nombre limité de participants et en conséquence le nombre limité de connexions de Transport en jeu, permettent d'envisager une adéquation de nos choix à ce contexte.

La deuxième partie du chapitre a montré la faisabilité du mécanisme de contrôle de congestion collaboratif. Une première étude a montré comment, sur des périodes de temps longues, le mécanisme de gestion coopérative de la bande passante pouvait être mis en œuvre en soulignant l'importance de la localisation des points de congestion. Le mécanisme proposé a également été évalué sous l'angle des perturbations qu'il était susceptible de susciter au niveau des connexions ne participant pas au mécanisme de collaboration. Une deuxième étude a permis d'illustrer la

mise en œuvre du modèle d'optimisation proposé au chapitre 3, proposant une répartition de la bande passante en fonction des connexions plus ou moins prioritaires. Les résultats ont permis de montrer l'efficacité de l'approche, y compris sur des périodes de temps courtes où la stabilité des débits n'est pas toujours évidente à acquérir.

Les perspectives de ce travail sont nombreuses. Si notre étude a permis de valider l'approche basée sur le modèle d'optimisation que nous avons proposé en montrant sa faisabilité, elle a également soulevé plusieurs interrogations, nous amenant à dire que l'étude doit être poursuivie.

Contrôle de congestion collaboratif. La première remarque concerne l'utilisation du mécanisme de contrôle de congestion TFRC et la modification qui lui a été apportée. Nous avons constaté des comportements divergents dans certains scénarios et certaines configurations de réseau. Une étude plus approfondie des comportements du mécanisme doit être menée afin d'améliorer le mécanisme proposé, soit en intégrant les augmentations et diminutions de débits requises directement dans l'équation 4.1 de TFRC, soit en combinant un mécanisme de Shaping à un TFRC classique par le biais d'un protocole de Transport configurable tel que ETP par exemple.

Impact de différents paramètres. La première interrogation, mentionnée ci-dessus, n'ayant pas été clarifiée, l'impact de plusieurs autres paramètres n'a pas été évalué dans nos simulations. Il s'agit ainsi de poursuivre l'étude en évaluant l'influence sur la mise en œuvre du mécanisme :

- du nombre de nœuds dans le réseau et des différents types de topologies
- du nombre de connexions donneuses et receveuses impliquées dans la collaboration
- du débit des connexions, ainsi que des valeurs de Δ/δ appliquées
- de la capacité des liens et des délais associés à ces liens.

Le mécanisme de collaboration basé sur le contrôle de congestion TFRC pourra ainsi être modifié, amélioré et validé.

Décision de collaboration. Pour la même raison, les questions liées à la prise de décision de la collaboration n'ont pas pu être abordées et doivent être considérées en perspective de ce travail. Parmi les questions possibles, celles qui nous semblent pertinentes sont par exemple les suivantes : à quel moment décide-t-on qu'une collaboration est nécessaire ? à partir de quel seuil de débit ? pendant combien de temps ce débit reste-t-il en dessous du seuil pour qu'une collaboration soit envisagée ? comment détecter que l'on doit mettre fin à la collaboration ?

Désynchronisation de la décision. Enfin, l'évaluation des conséquences d'une application asynchrone des décisions reste un travail à mener. En d'autres termes, il s'agit d'évaluer l'impact sur la répartition des ressources d'une mise en œuvre non synchronisée des diminutions et des augmentations de débit issues de la décision de collaboration.

Conclusion

Les travaux que nous avons présentés dans ce mémoire s'inscrivent dans la problématique générale de l'adaptabilité des systèmes de communication pour répondre aux besoins en QoS des applications amenées à être distribuées dans une vison future d'un Internet ambiant.

Dans cette optique, l'objectif de notre thèse était d'aborder le problème de l'adaptation dynamique de protocoles de Transport configurables, pour répondre aux besoins en QoS d'applications multimédias distribuées dans un contexte d'activités coopératives de groupe telles que les opérations d'intervention d'urgence.

Le chapitre 1 a présenté le panorama des solutions de gestion de la QoS dans l'Internet et a positionné nos travaux. Les différentes solutions présentées ont été classées en deux approches : l'une basée réservation pour la garantie de QoS, et l'autre basée adaptation pour l'optimisation de la QoS. Nos travaux s'inscrivent dans cette deuxième approche et visent plus spécifiquement à contribuer au développement de protocoles de Transport auto-configurables. L'adaptation ciblée touche les mécanismes du protocole qu'il s'agit de pouvoir paramétrer, voire remplacer, dynamiquement.

Le premier problème à traiter pour contribuer à l'auto-adaptation des protocoles est lié au besoin d'une architecture de contrôle de l'adaptation. Le chapitre 2 a présenté notre proposition d'architecture, développée dans le cadre du projet européen NETQoS. Les différentes fonctionnalités et les composants de cette architecture permettant de décider et de faire appliquer des actions d'adaptation au niveau Transport ont ainsi été présentés au travers de plusieurs cas d'utilisation. Nous avons également présenté de façon plus détaillée nos choix de conception pour les composants relevant de nos contributions spécifiques au projet, qui concernent le niveau Transport.

Le second problème à traiter pour contribuer à l'auto-adaptation des protocoles concerne les actions d'adaptation ciblées. Dans le chapitre 3, nous avons proposé une contribution originale pour l'élaboration d'un système de gestion coopérative de la QoS. Pour des activités coopératives de groupe qui induisent potentiellement des priorités entre communications, un tel système de gestion de la QoS peut être intéressant pour répondre au mieux aux pénuries de ressources induites par le réseau ou par son utilisation. Le chapitre 3 traite plus spécifiquement du problème de la répartition de la bande passante entre les connexions de Transport. Notre approche, basée modèle, nous a conduit à poser le problème sous la forme d'un problème de programmation linéaire, c'est-à-dire d'une fonction objectif à satisfaire sous contraintes. Nous avons par ailleurs discuté la pertinence et l'utilisation de différents critères. L'intégration de cette solu-

tion dans l'architecture présentée au chapitre 2 a également été proposée. Par ailleurs, le problème de la récupération des informations de supervision nécessaires à la mise en œuvre d'un système de gestion coopérative de la QoS, a été abordé.

Enfin, dans le but de montrer la faisabilité des choix d'architecture et du concept de gestion coopérative de la QoS proposés dans ce mémoire, le chapitre 4 a présenté une évaluation des contributions exposées aux chapitres 2 et 3. La première partie de ce chapitre a présenté l'évaluation des performances du composant de gestion de l'adaptation de niveau Transport (APA) du système NETQoS. La deuxième partie du chapitre a montré la faisabilité et la pertinence du mécanisme de contrôle de congestion coopératif. Les résultats ont permis de montrer l'efficacité de l'approche, y compris sur des périodes de temps courtes où la stabilité des débits n'est pas toujours évidente à acquérir.

Perspectives

Pour aboutir à une gestion coopérative de la QoS Transport dans un contexte d'activités de groupe de type OIU, les perspectives de nos travaux concernent tant la partie architecture de contrôle de l'adaptation, que la partie modèle de décision et actions d'adaptation au niveau des protocoles. Nous les présentons ci-après en commençant par la partie architecture.

La proposition d'architecture faite dans le cadre du projet NETQoS a été conçue indépendamment d'un environnement réseau spécifique. Pour aboutir à une architecture opérationnelle dans un contexte d'OIU (déployées dans des environnements de réseaux potentiellement mobiles et ad-hoc en tout ou partie), notre proposition est donc à raffiner afin de prendre en compte les caractéristiques et les contraintes spécifiques à ce type d'environnement.

De plus, la mise en œuvre d'une gestion coopérative de la QoS, suivant un modèle de répartition de la bande passante tel que celui proposé dans ce mémoire, suscite un besoin en monitoring des routeurs et des besoins applicatifs, pour lequel nous n'avons indiqué que des pistes. Le travail est donc à poursuivre en termes d'architecture de monitoring, en étudiant en particulier l'overhead induit en termes de trafic et en le confrontant au problème de la mise à l'échelle de la solution retenue vis-à-vis du nombre de connexions et du nombre de nœuds.

Concernant les aspects liés au modèle d'optimisation et aux actions opérationnelles pour la décision ainsi que la mise en œuvre de notre proposition de gestion coopérative de la QoS, plusieurs perspectives s'ouvrent également en suite directe de nos travaux.

Vis-à-vis du modèle de décision pour la répartition de la bande passante, un travail immédiat est d'étudier la complexité liée à la résolution du problème posé, en fonction d'un nombre de connexions ou de nœuds. Cette étude contribuerait à évaluer l'application du modèle dans un contexte défini.

Les perspectives à plus long terme attenantes au modèle d'optimisation sont les suivantes.

Notre proposition de gestion coopérative de la QoS repose sur une notion de priorité que nous avons instanciée de façon simple, distinguant ainsi les connexions dites donneuses (moins prioritaires) et les connexions dites receveuses (plus prioritaires).

Dans un contexte d'activités de groupe plus fortement hiérarchisées, considérer en conséquence une notion de priorité plus complexe amènerait à une extension de notre modèle. Parallèlement, la considération d'un modèle de réseau non plus filaire mais sans fil, voire mobile et ad hoc, nécessiterait également de revoir le modèle de façon conséquente. Plus théorique, une dernière perspective d'amélioration du modèle réside dans une étude approfondie d'un algorithme basé sur le simplexe qui permettrait de trouver l'ensemble de tous les sommets solutions, et l'ensemble complet des solutions, de la fonction objectif.

Vis-à-vis des actions d'adaptation, celles que nous avons naturellement testées reposent sur une utilisation particulière du mécanisme protocolaire TFRC, directement impliqué dans la répartition de la bande passante entre les connexions. Une étude plus approfondie de TFRC pourrait conduire à une meilleure utilisation de TFRC pour aboutir à un contrôle de congestion coopératif amélioré. Dans un contexte de réseaux ad-hoc, cette étude serait à mener sur la base des propositions de modification de TFRC faites pour ce nouveau contexte.

Parallèlement, plusieurs questions demeurent concernant par exemple la décision des dates de collaboration, notamment celles liées à la détection des seuils de débit qui doivent déclencher le début et la fin d'une collaboration. De même, l'impact sur le partage de la bande passante de l'application d'une décision de collaboration de manière asynchrone doit être évalué.

Bibliographie

[AKA 04] O. Akan and I. Akyildiz, "ATL : an adaptive transport layer suite for next-generation wireless internet", *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 5, 2004, pages 802–817, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA,. (Cité pages 14, 15 et 16.)

[AME 94] P. Amer, C. Chassot, T. Connolly, M. Diaz and P. Conrad, "Partial-order transport service for multimedia and other applications", *IEEE/ACM Transactions on Networking (TON)*, vol. 2, no. 5, 1994, pages 440–456, IEEE Press Piscataway, NJ, USA. (Cité pages 2 et 30.)

[BAL 01] H. Balakrishnan and S. Seshan, "The Congestion Manager", RFC 3124 (Proposed Standard), june 2001. (Cité page 32.)

[BAL 04] S. Balasubramaniam and J. Indulska, "Vertical handover supporting pervasive computing in future wireless networks", *Computer Communications*, vol. 27, no. 8, 2004, pages 708–719, Elsevier. (Cité pages 14 et 16.)

[BHA 98] N. Bhatti, M. Hiltunen, R. Schlichting and W. Chiu, "Coyote : A system for constructing fine-grain configurable communication services", *ACM Transactions on Computer Systems (TOCS)*, vol. 16, no. 4, 1998, pages 321–366, ACM New York, NY, USA. (Cité page 22.)

[BIR 00] K. Birman, B. Constable, M. Hayden, J. Hickey, C. Kreitz, R. Van Renesse, O. Rodeh and W. Vogels, "The horus and ensemble projects", proceedings of *Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEXoo)*, 2000. (Cité page 21.)

[BLA 98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Service", RFC 2475 (Informational), december 1998, Updated by RFC 3260. (Cité page 11.)

[BRA 97a] R. Braden and L. Zhang, "Resource ReSerVation Protocol (RSVP) – Version 1 Message Processing Rules", RFC 2209 (Informational), september 1997. (Cité page 10.)

[BRA 97b] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification", RFC 2205 (Proposed Standard), september 1997, Updated by RFCs 2750, 3936, 4495. (Cité pages 10 et 11.)

[BRI 01] P. Bridges, W. Chen, M. Hiltunen and R. Schlichting, "Supporting coordinated adaptation in networked systems", proceedings of *Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on*, 2001. (Cité page 17.)

[CAL 08] M. Callejo-Rodríguez, J. Enríquez-Gabeiras, W. Burakowski, A. Beben, J. Sliwinski, O. Dugeon, E. Mingozzi, G. Stea, M. Diaz, L. Baresse and others, "EuQoS : end-to-end QoS over heterogeneous networks", *Proceedings of the K-INGN*, , 2008. (Cité page 13.)

[CAS 90] J. Case, M. Fedor, M. Schoffstall and J. Davin, "Simple Network Management Protocol (SNMP)", RFC 1157 (Historic), mai 1990. (Cité page 86.)

[CHA 96] C. Chassot, M. Diaz and A. Lozes, "From the partial order concept to partial order multimedia connection", *Journal for high speed networks*, vol. 5, no. 2, 1996, pages 181–191. (Cité pages 3, 31 et 64.)

[CHE 05] L. Chen and W. Heinzelman, "QoS-aware routing based on bandwidth estimation for mobile ad hoc networks", *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 3, 2005, pages 561–572, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA,. (Cité page 16.)

[CRA 98] E. Crawley, R. Nair, B. Rajagopalan and H. Sandick, "A Framework for QoS-based Routing in the Internet", RFC 2386 (Informational), august 1998. (Cité page 7.)

[DAN 55] G. Dantzig, A. Orden and P. Wolfe, "The generalized simplex method for minimizing a linear form under linear inequality restraints", *Pacific Journal of Mathematics 8*, , 1955, pages 183-195. (Cité page 77.)

[DAS 04] L. DaSilva, S. Midkiff, J. Park, G. Hadjichristofi, N. Davis, K. Phanse and T. Lin, "Network mobility and protocol interoperability in ad hoc networks", *IEEE Communications Magazine*, vol. 42, no. 11, 2004, pages 88–96, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA,. (Cité page 16.)

[DUR 00] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748 (Proposed Standard), january 2000, Updated by RFC 4261. (Cité page 18.)

[ETS 94] E. ETSI, "003 :" Network Aspects (NA)", *General aspects of Quality of Service (QoS) and Network Performance (NP)*, , 1994. (Cité page 7.)

[EXP 03a] E. Exposito, "Specification and implementation of a QoS oriented Transport protocol for multimedia applications", PhD thesis, Institut National Polytechnique de Toulouse, 2003. (Cité pages 2, 22, 29 et 31.)

[EXP 03b] E. Exposito, P. Senac and M. Diaz, "FPTP : the XQoS aware and fully programmable transport protocol", proceedings of *ICON2003. The 11th IEEE International Conference on Networks*, 2003, pages 249–254. (Cité pages 2, 14, 15 et 16.)

[FAI 02] G. Fairhurst and L. Wood, "Advice to link designers on link Automatic Repeat reQuest (ARQ)", RFC 3366 (Best Current Practice), august 2002. (Cité page 27.)

[FAR 06] K. Farkas, O. Wellnitz, M. Dick, X. Gu, M. Busse, W. Effelsberg, Y. Rebahi, D. Sisalem, D. Grigoras, K. Stefanidis and others, "Real-time service provisioning for mobile and wireless networks", *Computer Communications*, vol. 29, no. 5, 2006, pages 540–550, Elsevier. (Cité page 14.)

[FLO 06] S. Floyd, E. Kohler and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3 : TCP-Friendly Rate Control (TFRC)", RFC 4342 (Proposed Standard), march 2006, Updated by RFC 5348. (Cité page 15.)

[FLO 08] S. Floyd, M. Handley, J. Padhye and J. Widmer, "TCP Friendly Rate Control (TFRC) : Protocol Specification", RFC 5348 (Proposed Standard), september 2008. (Cité pages 15, 17, 20 et 29.)

[GAR 01] T. Gary, M. Hitunen and R. Schlichting, "A configurable and extensible transport protocol", proceedings of *IEEE Infocom*, 2001, pages 22–26. (Cité page 23.)

[HAR 01] W. Hardy, *QoS : measurement and evaluation of telecommunications quality of service*, John Wiley & Sons Inc, 2001. (Cité page 7.)

[HEI 99] J. Heinanen, F. Baker, W. Weiss and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597 (Proposed Standard), june 1999, Updated by RFC 3260. (Cité page 12.)

[HEI 03] G. Heinz II, "Priorities In Stream Transmission Control Protocol (SCTP) Multistreaming", PhD thesis, University of Delaware, 2003. (Cité page 31.)

[HUT 91] N. Hutchinson and L. Peterson, "The x-kernel : An architecture for implementing network protocols", *IEEE Transactions on Software engineering*, vol. 17, no. 1, 1991, pages 64–76. (Cité page 21.)

[IBM 06] IBM, "An architectural blueprint for autonomic computing.", *white paper*, , 2006. (Cité pages 3 et 19.)

[ISO94] "Quality management and quality assurance – Vocabulary", second edition, 1994. (Cité page 7.)

[ISO00] "Quality management principle", 2000. (Cité page 7.)

[ITU 93] E. ITU, "800", *ITU-T Recommendation E*, vol. 800, 1993, pages 1–12. (Cité page 7.)

[ITU01] "ITU-T G.1010 - End-user multimedia QoS categories", november 2001. (Cité pages 9 et 38.)

[JAC 99] V. Jacobson, K. Nichols and K. Poduri, "An Expedited Forwarding PHB", RFC 2598 (Proposed Standard), june 1999, Obsoleted by RFC 3246. (Cité page 12.)

[KAL 06] A. Kaloxylos, G. Lampropoulos, N. Passas and L. Merakos, "A flexible handover mechanism for seamless service continuity in heterogeneous environments", *Computer Communications*, vol. 29, no. 6, 2006, pages 717–729, Elsevier. (Cité pages 14 et 16.)

[KHA 01] I. Khalil and T. Braun, "Implementation of a bandwidth broker for dynamic end-to-end capacity reservation over multiple diffserv domains", *Lecture Notes in Computer Science*, , 2001, pages 160–174, Springer. (Cité page 12.)

[KOH 06] E. Kohler, M. Handley and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340 (Proposed Standard), march 2006, Updated by RFCs 5595, 5596. (Cité pages 2 et 20.)

[LAN 04] R. Landry, K. Grace, A. Saidi and M. Bedford, "On the design and management of heterogeneous networks : a predictability-based perspective", *IEEE Communications Magazine*, vol. 42, no. 11, 2004, pages 80–87, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA,. (Cité page 17.)

[LI 98] T. Li and Y. Rekhter, "A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)", RFC 2430 (Informational), october 1998. (Cité page 11.)

[LUB 02] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley and J. Crowcroft, "Forward Error Correction (FEC) Building Block", RFC 3452 (Experimental), december 2002, Obsoleted by RFCs 5052, 5445. (Cité page 26.)

[MAR 01] I. Marshall and C. Roadknight, "Provision of quality of service for active services", *Computer Networks*, vol. 36, no. 1, 2001, pages 75–85, Elsevier. (Cité page 14.)

[MIR 01] H. Miranda, A. Pinto and L. Rodrigues, "Appia : A flexible protocol kernel supporting multiple coordinated channels", proceedings of *INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS*, vol. 21, IEEE Computer Society ; 1999, 2001, pages 707–710. (Cité page 21.)

[MOC 05] J. Mocito, L. Rosa, N. Almeida, H. Miranda, L. Rodrigues and A. Lopes, "Context Adaptation of the Communication Stack", proceedings of *In ICDCSW '05 : Proceedings of the Third International Workshop on Mobile Distributed Computing (MDC) (ICDCSW'05). IEEE Computer Society*, 2005. (Cité pages 15 et 16.)

[NIC 98] K. Nichols, S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474 (Proposed Standard), december 1998, Updated by RFCs 3168, 3260. (Cité page 11.)

[NIC 99] K. Nichols, V. Jacobson and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", RFC 2638 (Informational), july 1999. (Cité page 12.)

[OTT 07] D. Ott and K. Mayer-Patel, "An open architecture for transport-level protocol coordination in distributed multimedia applications", *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, vol. 3, no. 3, 2007, page 17, ACM. (Cité page 32.)

[OWE 98] P. Owezarski, M. Diaz and C. Chassot, "A time-efficient architecture for multimedia applications", *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 3, 1998, pages 383–396, Citeseer. (Cité pages 3, 30, 31 et 64.)

[POS 80] J. Postel, "User Datagram Protocol", RFC 768 (Standard), august 1980. (Cité page 20.)

[POS 81] J. Postel, "Transmission Control Protocol - DARPA Internet Program Protocol Specification–RFC 793", *Network Information Center. SRI International, Menlo Park. Calif*, , 1981. (Cité pages 20 et 25.)

[RAC 08] S. Racaru, *Conception et validation d'une architecture de signalisation pour la garantie de qualité de service dans l'Internet multi-domaine, multi-technologie et multi-service*, 2008. (Cité page 13.)

[REC 02] I. Rec, "Y. 1541", *Network performance objectives for IP-based services*, , 2002. (Cité pages 9 et 38.)

[RIT 84] D. Ritchie, "A stream input-output system", *AT&T Bell Laboratories Technical Journal*, vol. 63, no. 8, 1984, pages 1897–1910, Citeseer. (Cité page 23.)

[SAM 05] N. Samaan and A. Karmouch, "An automated policy-based management framework for differentiated communication systems", *IEEE Journal on Selected Areas in communications*, vol. 23, no. 12, 2005, pages 2236–2247, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA,. (Cité pages 16 et 18.)

[SCH 93] D. Schmidt, D. Box and T. Suda, "ADAPTIVE : A Dynamically Assembled Protocol Transformation, Integration, and eValuation Environment", *Concurrency : Practice and Experience*, , 1993, Citeseer. (Cité pages 22 et 23.)

[SHA 01] N. Shaha, A. Desai and M. Parashar, "Multimedia content adaptation for QoS management over heterogeneous networks", proceedings of *International Conference on Internet Computing*, Citeseer, 2001. (Cité page 15.)

[SHE 97] S. Shenker, C. Partridge and R. Guerin, "Specification of Guaranteed Quality of Service", RFC 2212 (Proposed Standard), september 1997. (Cité pages 10 et 11.)

[SKA 04] A. Skarmeta and G. Perez, "Policy-based dynamic provision of IP services in a secure VPN coalition scenario", *IEEE Communications Magazine*, vol. 42, no. 11, 2004, pages 118–124. (Cité pages 14 et 16.)

[STE 99] M. Stevens, W. Weiss, H. Mahon, B. Moore, J. Strassner, G. Waters, A. Westerinen and J. Wheeler, "Policy Framework (Draft)", *Internet Engineering Task Force*, , 1999. (Cité pages 3 et 18.)

[STE 00] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang and V. Paxson,

"Stream Control Transmission Protocol", RFC 2960 (Proposed Standard), october 2000, Obsoleted by RFC 4960, updated by RFC 3309. (Cité page 20.)

[STE 04] R. Stewart, M. Ramalho, Q. Xie, M. Tuexen and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758 (Proposed Standard), mai 2004. (Cité pages 3, 21 et 64.)

[STE 07] R. Stewart, "Stream Control Transmission Protocol", RFC 4960 (Proposed Standard), september 2007. (Cité page 15.)

[TER 99] A. Terzis, L. Wang, J. Ogawa and L. Zhang, "A two-tier resource management model for the internet", *GLOBECOM-NEW YORK-*, vol. 3, 1999, pages 1779–1791, Citeseer. (Cité page 13.)

[VAN 09] N. Van Wambeke, "Une approche pour la composition autonome de services de communication orientés QoS : application aux protocoles de transport configurables", PhD thesis, Institut National des Sciences Appliquées - Université de Toulouse, 2009. (Cité pages 21 et 23.)

[WON 01] G. Wong, M. Hiltunen and R. Schlichting, "A configurable and extensible transport protocol", proceedings of *IEEE INFOCOM*, vol. 1, Citeseer, 2001, pages 319–328. (Cité pages 15 et 16.)

[WRO 97a] J. Wroclawski, "Specification of the Controlled-Load Network Element Service", RFC 2211 (Proposed Standard), september 1997. (Cité pages 10 et 11.)

[WRO 97b] J. Wroclawski, "The Use of RSVP with IETF Integrated Services", RFC 2210 (Proposed Standard), september 1997. (Cité page 11.)

[WU 00] D. Wu, Y. Hou and Y. Zhang, "Transporting real-time video over the Internet : Challenges and approaches", *Proceedings of the IEEE*, vol. 88, no. 12, 2000, pages 1855–1875, Institute of Electrical and Electronics Engineers. (Cité pages 8, 25 et 29.)

[WU 01] D. Wu, Y. Hou, W. Zhu, Y. Zhang and J. Peha, "Streaming video over the internet : Approaches and directions", *IEEE transactions on circuits and systems for video technology*, vol. 11, no. 3, 2001, pages 282–300, Citeseer. (Cité pages 14, 15 et 16.)

[YU 03] F. Yu, Q. Zhang, W. Zhu and Y. Zhang, "QoS-adaptive proxy caching for multimedia streaming over the Internet", *IEEE transactions on circuits and systems for video technology*, vol. 13, no. 3, 2003, pages 257–269, Citeseer. (Cité page 14.)

[ZHA 01] Z. Zhang, Z. Duan and Y. Hou, "On scalable design of bandwidth brokers", *IEICE TRANSACTIONS ON COMMUNICATIONS E SERIES B*, vol. 84, no. 8, 2001, pages 2011–2025, INSTITUTE OF ELECTRONICS, INFORMATION &. (Cité page 13.)

[ZIM 80] H. Zimmermann, "OSI reference model–The ISO model of architecture for open systems interconnection", *IEEE Transactions on communications*, vol. 28, no. 4, 1980, pages 425–432. (Cité page 14.)

Bibliographie de l'auteur

Revues Internationales avec comité de lecture

[CHA 06a] C. Chassot, K. Guennoun, K. Drira, F. Armando, E. Exposito and A. Lozes, "Towards autonomous management of QoS through model-driven adaptability in communication-centric systems", *International Transactions on Systems Science and Applications*, vol. 2, no. 3, 2006, pages 255–264.

[GUE 08] K. Guennoun, K. Drira, N. Van Wambeke, C. Chassot, F. Armando and E. Exposito, "A framework of models for QoS-oriented adaptive deployment of multi-layer communication services in group cooperative activities", *Computer Communications*, vol. 31, no. 13, 2008, pages 3003–3017, Elsevier.

[VAN 08a] N. Van Wambeke, F. Armando, C. Chassot and E. Expósito, "A model-based approach for self-adaptive Transport protocols", *Computer Communications*, vol. 31, no. 11, 2008, pages 2699–2705, Elsevier.

[VAN 09b] N. Van Wambeke, F. Armando, A. Abdelkafi, C. Chassot, K. Guennoun and K. Drira, "Models and Architecture for Autonomic Network Management", *Int. Journal of Business Data Communications and Networking*, vol. 5, no. 2, 2009, pages 35–51.

Conférences Internationales avec comité de lecture

[ARM 08] F. Armando, N. Van Wambeke and C. Chassot, "Introducing a collaborative congestion control based on TFRC", proceedings of *the World Congress on Engineering and Computer Science (WCECS)*, San Francisco, USA, October 2008, IAENG.

[VAN 08b] N. Van Wambeke, F. Armando, C. Chassot, K. Guennoun, K. Drira and E. Exposito, "Towards the Use of Models for Autonomic Network Management", *IFIP International Conference on Personal Wireless Communications (PWC'2008) - Toulouse, France*, September 30, October 2, 2008.

[ARM 07] F. Armando, N. Van Wambeke, C. Chassot, E. Expósito and K. Drira, "A framework for the dynamic configuration of adaptive transport protocols", *Proceedings of the International Conference on Wireless Information Networks and Systems (WINSYS 2007)*, Barcelona (Spain), 28-31 July 2007, pages 275-283.

[VAN 07] N. Van Wambeke, F. Armando, C. Chassot and E. Expósito, "Architecture and Models for self-adaptability of Transport protocols", *Proceedings of the 21st IEEE International Conference on Advanced Information Networking and Applications (AINA-07) Workshops*, Niagara Falls, Canada, May 21-23 2007.

[CHA 06b] C. Chassot, K. Drira, F. Armando, E. Exposito and A. Lozes, "A model-based coordinated adaptability framework for QoS management in cooperative mobile and wireless applications", *Proceedings of the 9th ACM international symposium on Modeling analysis and Simulation of Wireless and Mobile System*, (ACM/IEEE MSWiM ?06), Torremolinos, Spain, October 2-6, 2006.

Rapports Projet européen IST NETQoS

[CHA 07a] C. Chassot, E. Exposito, N. Van Wambeke, F. Armando, I. Bouassida, K. Drira, C. Brandauer, F. Strohmeier, P. Dorfinger, P. A. A. Gutierrez, A. Flizikowski, K. Samp, I. Miloucheva, S. Avallone, L. Vollero, S. P. Romano, M. Michalak, M. Roth and S. Rao, "D2.1 - QoS solution Analysis and User/application requirements and business scenarios", Tech. report, 2007, IST FP6 - NetQoS.

[CHA 07b] C. Chassot, E. Exposito, N. Van Wambeke, F. Armando, C. Brandauer, T. Fichtel, S. Kavthasi, S. Rao, I. Miloucheva, D. Wagner, A. Flizikowski, R. Marcinczak, S. Avallone, S. P. Romano and A. Dainotti, "D2.3 - Policy specification", Tech. report, 2007, IST FP6 - NetQoS.

[CHA 07c] C. Chassot, E. Exposito, N. Van Wambeke, F. Armando, C. Brandauer, T. Fichtel, S. Kavthasi, S. Rao, I. Miloucheva, D. Wagner, A. Flizikowski, R. Marcinczak, S. Avallone, S. P. Romano and A. Dainotti, "D2.4 - Specification of policy-oriented measurement, learning, and context identification", Tech. report, 2007, IST FP6 - NetQoS.

[CHA 07d] C. Chassot, E. Exposito, N. Van Wambeke, F. Armando, C. Brandauer, J. Neuhaus, M. Roth, S. Rao, I. Miloucheva, A. Flizikowski, R. Marcinczak, P. A. A. Gutierrez, S. Avallone and S. P. Romano, "D2.2 - Design of the NetQoS policy architecture", Tech. report, 2007, IST FP6 - NetQoS.

[CHA 08] C. Chassot, E. Exposito, N. Van Wambeke, F. Armando, C. Brandauer, S. Khavatsi, S. Rao, S. Avallone, S. Romano, A. Pescape, A. Botta, A. Dainotti, W. D. Donato, L. Vollero and P. A. Gutierrez, "D2.6 : Implementation of policy-oriented measurement, learning, and context identification", Tech. report, 2008, IST FP6 - NetQoS.

[FLI 08] A. Flizikowski, R. Marcinczac, I. Miloucheva, S. Khavatsi, S. Rao, S. Avallone, S. Romano, N. Van Wambeke, F. Armando, E. Exposito and C. Chassot, "D2.5 : Policy implementation", Tech. report, 2008, IST FP6 - NetQoS.

[GUT 07] P. A. A. Gutierrez, C. Chassot, E. Exposito, F. Armando, N. Armando, N. Van Wambeke, C. Brandauer, S. Khavtasi, S. Rao, D. Wagner and A. Flizikowski, "D3.1 - Specification of policy-oriented measurement, learning, and context identification", Tech. report, 2007, IST FP6 - NetQoS.

[GUT 08a] P. A. Gutierrez, F. Trivunac, I. Miloucheva, D. Wagner, C. Niephaus, C. Chassot, N. Van Wambeke, C. Brandauer, A. Flizikowski, F. Armando, S. Rao and S. P. Romano, "D3.4 : Performance Assessment", Tech. report, 2008, IST FP6 - NetQoS.

[GUT 08b] P. A. A. Gutierrez, S. Rao, S. Khavtasi, C. Brandauer, A. Flizikowski, C. Chassot, N. Van Wambeke, F. Armando, I. Miloucheva, D. Wagner and S. P. Romano, "D3.2 : System integration (first prototype) and test report", Tech. report, 2008, IST FP6 - NetQoS.

[MIL 08a] I. Miloucheva, D. Wagner, C. Niephaus, C. Chassot, N. Van Wambeke, P. A. Gutierrez, F. Trivunac, C. Brandauer, A. Flizikowski, S. Khavtasi, F. Armando, S. Rao and S. P. Romano, "D3.3 : Trials execution", Tech. report, 2008, IST FP6 - NetQoS.

[MIL 08b] I. Miloucheva, D. Wagner, C. Niephaus, C. Chassot, N. Van Wambeke, P. A. Gutierrez, F. Trivunac, C. Brandauer, A. Flizikowski, S. Khavtasi, F. Armando, S. Rao and S. P. Romano, "D3.5 : Validation and Exploitation report", Tech. report, 2008, IST FP6 - NetQoS.

Glossaire

AF	Assured Forwarding
APA	Automated Policy Adaptor
API	Application Programming Interface (interface de programmation)
APM	Actor Preference Manager
ARQ	Automatic Repeat Request
AS	Autonomous System
CL	Controlled Load
CM	Context Manager
COPS	Common Open Policy Service
DCCP	Datagram Congestion Control Protocol
EF	Expedited Forwarding
ETP	Enhanced Transport Protocol
FEC	Forward Error Correction
FIFO	First In, First Out
GUI	Graphical User Interface
GS	Guaranted Service
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
ITU	International Telecommunication Union
JVM	JavaVirtual Machine
MAC	Media Access Control
MoMe	Monitoring and Measurement
OIU	Opérations d'Intervention d'Urgence
OSI	Open Systems Interconnection
PAM	Policy Adaptation Manager
PBNM	Policy Based Network Management
PDA	Personal Digital Assistant
PDP	Policy Decision Point
PDM	Policy Decision Manager
PDU	Protocol Data Unit
PEP	Policy Enforcement Point
PEM	Policy Enforcement Manager
PHB	Per Hop Behavior
PIB	Policy Information Base
POC	Partial Order Connection
POLD	Policy Description
RSVP	Resource ReSerVation Protocol
RTP	Real time Transport Protocol

QoS	Quality of Service (Qualité de Service)
SCTP	Stream Control Transmission Protocol
SLA	Service Level Agreement
SLS	Service Level Specification
TFRC	TCP Friendly Rate Control
TCP	Transmission Control Protocol
TOS	Type of Services
UDP	User Datagram Protocol
UML	Unified Modeling Language
XML	eXtensible Markup Language

AUTHOR : François ARMANDO

TITLE : Cooperative QoS for Transport protocols adaptability within group activities

SUPERVISORS : Christophe CHASSOT, Khalil DRIRA

Abstract : The work presented in this thesis fits into the overall theme of adaptability of communication systems for QoS. The proposed contribution focuses on the dynamic adaptation of configurable transport protocols to meet the QoS requirements of distributed multimedia applications in the context of cooperative activities such as emergency operations. The first part was conducted in the framework of the NetQoS European project and concerns the proposal of a conceptual architecture for implementing adaptation at the Transport layer. Within a policy-based approach, our contribution has addressed the development of the architecture of the main component. This component is in charge of designing, adapting and enforcing operational policies to the entities responsible for direct QoS management, typically at Transport level. The second part focuses on the decision problem and proposes a novel approach for collaborative management of transport protocols for supporting adaptive QoS in cooperative group activities. The approach explored is to focus on transport connections with the highest priority to the detriment of lower priority connections, within the allowed sending rate. We consider the problem of allocating bandwidth between these connections. After defining the problem and the data required for its resolution, we formulate it as a linear programming problem solved by the simplex method. Two minimization criteria are considered and discussed. Finally, measures of performance achieved in simulation highlight the benefits and limitations of proposed solutions.

Keywords : Quality of Service, Transport protocol, Adaptability, Group activities

AUTEUR : François ARMANDO

TITRE : QoS coopérative pour l'adaptabilité des protocoles de Transport dans le contexte des activités de groupe

DIRECTEURS DE THESE : Christophe CHASSOT, Khalil DRIRA

LIEU ET DATE DE SOUTENANCE : LAAS-CNRS, Toulouse, le 17 février 2010

Résumé : Les travaux présentés dans cette thèse s'inscrivent dans le thème général de l'adaptabilité des systèmes de communication pour la QoS. La contribution proposée porte sur l'adaptation dynamique de protocoles de Transport, pour répondre aux besoins en QoS d'applications multimédia distribuées dans un contexte d'activités coopératives de groupe de type opérations d'intervention d'urgence. La première partie a été réalisée dans le cadre du projet européen projet IST NETQoS et porte sur une proposition d'architecture conceptuelle pour la mise en œuvre de l'adaptation au niveau Transport. Suivant une approche basée politique, les contributions exposées portent sur l'élaboration de l'architecture du composant central du système, chargé de concevoir, d'adapter et de faire appliquer des politiques opérationnelles aux entités en charge de la gestion directe de la QoS. La deuxième partie porte sur le problème de la décision et propose une approche originale de gestion collaborative des protocoles de Transport pour la QoS dans les activités coopératives de groupe. L'approche explorée vise à privilégier les connexions de Transport les plus prioritaires au détriment des connexions les moins prioritaires. Nous nous intéressons au problème de la répartition de la bande passante entre ces connexions. Après avoir défini le problème et les données nécessaires à sa résolution, nous le formulons sous la forme d'un problème de programmation linéaire résolu grâce à la méthode du simplexe. Deux critères de minimisation sont envisagés et discutés. Enfin, des mesures de performance réalisées en simulation permettent de mettre en évidence le bien fondé et les limites des solutions proposées.

Mots-clés : Qualité de Service, Protocole de Transport, Adaptabilité, Activités coopératives

---

Discipline Administrative : Informatique et Télécommunications

---

Intitulé et adresse du Laboratoire : Laboratoire d'Analyse et d'Architectures des Systèmes - 7 avenue du colonel roche - 31077 Toulouse