



**HAL**  
open science

# Propositions de méthodes pour adapter le réseau aux contraintes d'applications temps-réel

Idriss Diouri

► **To cite this version:**

Idriss Diouri. Propositions de méthodes pour adapter le réseau aux contraintes d'applications temps-réel. Autre. Université Henri Poincaré - Nancy 1, 2010. Français. NNT : 2010NAN10091 . tel-01746353v1

**HAL Id: tel-01746353**

**<https://hal.univ-lorraine.fr/tel-01746353v1>**

Submitted on 29 Mar 2018 (v1), last revised 21 Jul 2011 (v2)

**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.



## AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : [ddoc-theses-contact@univ-lorraine.fr](mailto:ddoc-theses-contact@univ-lorraine.fr)

## LIENS

Code de la Propriété Intellectuelle. articles L 122. 4

Code de la Propriété Intellectuelle. articles L 335.2- L 335.10

[http://www.cfcopies.com/V2/leg/leg\\_droi.php](http://www.cfcopies.com/V2/leg/leg_droi.php)

<http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm>

U.F.R. : Faculté des Sciences et Technologies

École doctorale : IAEM Lorraine

Département de Formation Doctorale : Automatique

Thèse

présentée pour l'obtention du titre de

Docteur de l'Université Henri Poincaré, Nancy 1

en Sciences, spécialité Automatique,  
Traitement du Signal et Génie Informatique

par **Idriss DIOURI**

**Propositions de méthodes pour adapter le réseau  
aux contraintes d'applications temps-réel**

Soutenue publiquement le 15 octobre 2010

**Membres du jury :**

*Président :*

Michel Dambrine      Professeur, Université de Valenciennes et du Hainaut-Cambrésis, LAMIH

*Rapporteurs :*

Philippe Fraisse      Professeur, Université Montpellier 2, LIRMM

Daniel Simon      Chargé de Recherche, Habilité à Diriger des Recherches, INRIA Rhône-Alpes

*Examineur :*

Vincent Bombardier      Maître de conférences, Université Henri Poincaré Nancy 1, CRAN

*Directeur de thèse :*

Eric Rondeau      Professeur, Université Henri Poincaré Nancy 1, CRAN

*Co-encadrant :*

Jean-Philippe Georges      Maître de conférences, Université Henri Poincaré Nancy 1, CRAN







U.F.R. : Faculté des Sciences et Technologies

École doctorale : IAEM Lorraine

Département de Formation Doctorale : Automatique

Thèse

présentée pour l'obtention du titre de

Docteur de l'Université Henri Poincaré, Nancy 1

en Sciences, spécialité Automatique,  
Traitement du Signal et Génie Informatique

par **Idriss DIOURI**

**Propositions de méthodes pour adapter le réseau  
aux contraintes d'applications temps-réel**

Soutenue publiquement le 15 octobre 2010

**Membres du jury :**

*Président :*

Michel Dambrine      Professeur, Université de Valenciennes et du Hainaut-Cambrésis, LAMIH

*Rapporteurs :*

Philippe Fraisse      Professeur, Université Montpellier 2, LIRMM

Daniel Simon      Chargé de Recherche, Habilité à Diriger des Recherches, INRIA Rhône-Alpes

*Examineur :*

Vincent Bombardier      Maître de conférences, Université Henri Poincaré Nancy 1, CRAN

*Directeur de thèse :*

Eric Rondeau      Professeur, Université Henri Poincaré Nancy 1, CRAN

*Co-encadrant :*

Jean-Philippe Georges      Maître de conférences, Université Henri Poincaré Nancy 1, CRAN



# Remerciements

En premier lieu, je tiens à remercier M. Eric RONDEAU, Directeur de ma thèse et Professeur à l'Université Henri Poincaré Nancy 1, pour son encadrement et les nombreux conseils qu'il m'a prodigués tout au long de mon doctorat. Je remercie également M. Jean-Philippe GEORGES, Maître de conférences à l'Université Henri Poincaré Nancy 1, pour son encadrement et les nombreuses discussions qu'on a pu avoir durant ma thèse.

Je tiens aussi à remercier M. Vincent BOMBARDIER, Maître de conférences à l'Université Henri Poincaré Nancy 1, pour avoir accepté de participer à mon jury de thèse et pour les discussions intéressantes que j'ai eues avec lui.

J'adresse mes remerciements à M. Philippe FRAISSE, Professeur à l'Université de Montpellier 2, et M. Daniel SIMON, Chargé de Recherche Habilité à Diriger des Recherches à l'INRIA Rhône-Alpes, pour avoir accepté de rapporter sur ma thèse et pour l'intérêt qu'ils ont manifesté pour mon travail.

Je remercie M. Michel DAMBRINE, Professeur à l'Université de Valenciennes et du Hainaut-Cambrésis, pour avoir présidé mon jury de thèse.

Mes remerciements iront aussi à tous les membres du groupe thématique Systèmes de Production Ambiants pour la convivialité qui a régné tout au long de ma thèse.

Enfin, je tiens à exprimer ma gratitude et ma reconnaissance à mes parents pour leur soutien et leurs encouragements et cela depuis mon plus jeune âge. Et je tiens aussi à remercier ma femme, mon frère ainsi que toute ma famille pour les moments agréables que j'ai passés avec eux et pour leurs encouragements durant la période de ma thèse.



*À mes parents*

*À ma femme*

*À mon frère*

*À toute ma famille*



# Table des matières

<b>Introduction générale.....</b>	<b>1</b>
<b>Partie I : Positionnement .....</b>	<b>7</b>
<b>Chapitre 1 : Les systèmes contrôlés en réseaux.....</b>	<b>9</b>
<b>1.1. Les systèmes contrôlés en réseaux .....</b>	<b>9</b>
1.1.1. Introduction .....	9
1.1.2. Impact du délai sur les SCR .....	10
1.1.3. Conclusion.....	13
<b>1.2. Approches appréhendant la QdC des SCR.....</b>	<b>14</b>
1.2.1. Approche commande en réseau.....	14
1.2.2. Approche commande de réseau.....	17
1.2.3. Vers une conception conjointe .....	20
1.2.4. Conclusion.....	22
<b>1.3. Contexte de cette recherche et positionnement.....</b>	<b>23</b>
1.3.1. Contexte .....	23
1.3.2. Positionnement .....	24
<b>Chapitre 2 : Les systèmes contrôlés en réseaux utilisant le réseau Ethernet.....</b>	<b>27</b>
<b>2.1. Ethernet.....</b>	<b>27</b>
2.1.1. Présentation de l’Ethernet .....	27
2.1.2. Méthodes permettant de rendre Ethernet adapté à interconnecter des systèmes temps-réel.....	31
2.1.3. Classification de service.....	33
<b>2.2. Variantes des ordonnancements équitables basés sur GPS.....</b>	<b>34</b>
<b>2.3. Exemples illustrant l’impact des délais sur le comportement d’un SCR utilisant le réseau Ethernet.....</b>	<b>37</b>
2.3.1. Evaluation expérimentale (Diouri <i>et al.</i> , 2007b).....	37
2.3.2. Apport d’un ordonnancement WRR dans le cas de la commande d’un drone .....	41
2.3.3. Simulation d’architecture Ethernet commutée sous OPNET (OPNET, 2005) .....	47
<b>2.4. Conclusion.....</b>	<b>55</b>

<b>Partie II : Contributions .....</b>	<b>57</b>
<b>Chapitre 3 : Approche commande de réseau pour les SCR à contraintes temps-réel strictes.....</b>	<b>59</b>
<b>3.1. Introduction .....</b>	<b>59</b>
<b>3.2. Présentation du calcul réseau.....</b>	<b>60</b>
3.2.1. Quelques définitions du calcul réseau .....	60
3.2.2. Remarques .....	64
<b>3.3. Modèle du commutateur et modélisation de sa courbe de service.....</b>	<b>64</b>
3.3.1. De l'architecture en bus vers une architecture commutée .....	64
3.3.2. Modèle de commutateur considéré.....	65
3.3.3. Modèle de l'arrivée des données du flux temps-réel .....	66
3.3.4. Courbe de service .....	67
3.3.5. Calcul du délai maximal (cas de $m$ ordonnanceurs).....	72
3.3.6. Problématique de l'agrégation de flux.....	77
<b>3.4. Comparaison des résultats du délai avec ceux obtenus avec une <i>rate-latency</i> .....</b>	<b>82</b>
3.4.1. Courbe de service de type <i>rate-latency</i> offerte à un ordonnanceur WRR .....	82
3.4.2. Courbe de service de type <i>rate-latency</i> offerte à une série d'ordonnanceurs WRR .....	83
3.4.3. Pessimisme de la courbe de service de type <i>rate-latency</i> .....	84
3.4.4. Conclusions .....	85
<b>3.5. Comparaison entre les délais maxima obtenus en utilisant le calcul réseau et ceux obtenus en simulation.....</b>	<b>86</b>
<b>3.6. Recherche des poids optimaux .....</b>	<b>87</b>
<b>3.7. Utilisation des résultats obtenus par calcul réseau.....</b>	<b>90</b>
3.7.1. Résultats du calcul réseau appliqués sur le drone (Diouri <i>et al.</i> , 2008).....	90
3.7.2. Résultats du calcul réseau appliqués sur un SCR simulé sur OPNET .....	92
<b>3.8. Conclusion.....</b>	<b>94</b>
<b>Chapitre 4 : Approche commande de réseau pour les SCR à contraintes temps-réel souples .....</b>	<b>95</b>
<b>4.1. Introduction .....</b>	<b>95</b>
<b>4.2. Présentation de la logique floue .....</b>	<b>97</b>
4.2.1. Concept de sous-ensemble flou.....	97
4.2.2. Concept de relation floue.....	98
4.2.3. Notion de variable linguistique .....	98
4.2.4. Notion de proposition floue.....	98
4.2.5. Notion de règle floue .....	99

---

<b>4.3. Identification du modèle du délai par classification floue .....</b>	<b>100</b>
4.3.1. Présentation du système étudié et des conditions de simulation.....	100
4.3.2. Identification du système par règles floues conjonctives .....	106
<b>4.4. La commande floue .....</b>	<b>112</b>
<b>4.5. Synthèse de la commande floue pour la régulation du délai .....</b>	<b>113</b>
<b>4.6. Résultats des simulations .....</b>	<b>116</b>
4.6.1. Adaptation en ligne des poids du WRR sur OPNET .....	116
4.6.2. Mise en place de la stratégie de reconfiguration dynamique des poids sur OPNET .....	119
<b>4.7. Conclusion.....</b>	<b>125</b>
<b>Conclusion générale .....</b>	<b>129</b>
<b>Liste des publications .....</b>	<b>133</b>
<b>Bibliographie.....</b>	<b>135</b>



# Table des figures

Figure 1. Réseau utilisé par un SCR et partagé avec d'autres types d'applications .....	2
Figure 1.1. Réseau support d'un SCR et partagé avec d'autres types d'applications.....	10
Figure 1.2. Organisation du SCR (Branicky <i>et al.</i> , 2000) .....	12
Figure 1.3. Système de commande numérique distribué avec des délais induits, $\tau_k^{sc}$ et $\tau_k^{ca}$ . Le délai de calcul dans le nœud du contrôleur, $\tau_k^c$ , est aussi indiqué (Nilsson, 1998).....	13
Figure 1.4. Schéma d'un prédicteur de Smith pour la compensation du délai induit par le réseau (Vatanski <i>et al.</i> , 2006).....	15
Figure 1.5. Stratégie prédictive basée sur les réseaux de neurones pour les SCR avec des délais (Lin <i>et al.</i> , 2008) .....	16
Figure 1.6. Boucle de conception globale d'un SCR (Branicky <i>et al.</i> , 2003) .....	21
Figure 1.7. Architecture du système global (Felicioni <i>et al.</i> , 2008) .....	22
Figure 1.8. Principes de base du projet NeCST ( <a href="http://www.strep-necst.org">http://www.strep-necst.org</a> ) .....	23
Figure 2.1. Résolution d'une collision (Divoux <i>et al.</i> , 2008).....	29
Figure 2.2. Files d'attente pour chaque classe de service au niveau d'un port de sortie d'un commutateur .....	33
Figure 2.3. Les services offerts par GPS et par le WFQ (Bennett et Zhang, 1996) .....	35
Figure 2.4. Le comportement du WRR (Diouri <i>et al.</i> , 2007a) .....	36
Figure 2.5. Plateforme expérimentale (Diouri <i>et al.</i> , 2007b) .....	38
Figure 2.6. Comportement du système (Diouri <i>et al.</i> , 2007b).....	38
Figure 2.7. Comportement du système (cas de la compensation de l'erreur) (Diouri <i>et al.</i> , 2007b).....	40
Figure 2.8. Réglage des poids (cas de la compensation de l'erreur) (Diouri <i>et al.</i> , 2007b) .....	40
Figure 2.9. Benchmark du drone (Diouri <i>et al.</i> , 2008).....	41
Figure 2.10. Architecture du drone incluant le réseau embarqué (Diouri <i>et al.</i> , 2008) .....	42
Figure 2.11. Impact sur la sortie du système (Diouri <i>et al.</i> , 2008).....	44
Figure 2.12. Les paramètres du bloc réseau étendu de TrueTime (Diouri <i>et al.</i> , 2008) .....	46
Figure 2.13. Ethernet commuté à 10 Mb/s, avec une CdS WRR et avec un trafic de fond non synchronisé (Diouri <i>et al.</i> , 2008) .....	47
Figure 2.14. Architecture Ethernet commutée étudiée dans la thèse.....	48
Figure 2.15. Architecture Ethernet commutée étudiée en utilisant le logiciel OPNET (OPNET, 2005) .....	49
Figure 2.16. Nœuds pour les SCR sur OPNET .....	50
Figure 2.17. Parties du code permettant de modifier les équations du contrôleur et du process .....	51
Figure 2.18. Evolution des délais aux niveaux du contrôleur et du process en présence de la charge .....	52
Figure 2.19. Comportement du SCR étudié dans la thèse en présence de la charge .....	52
Figure 2.20. Evolution des délais aux niveaux du contrôleur et du process en présence de la charge (cas du WRR avec $w_{Trafic_{TR}} = 30$ et $w_{Trafic_{TF}} = 1$ ) .....	54

Figure 2.21. Comportement du SCR considéré dans la thèse avec WRR ( $w_{Trafic_{TR}} = 30$ et $w_{Trafic_{TF}} = 1$ ).....	54
Figure 3.1. Majorant du délai $D_{max}$ et de l'arriéré de traitement $x(t)_{max}$ .....	62
Figure 3.2. Cas d'un paquet traversant $m$ commutateurs.....	63
Figure 3.3. Cas d'un paquet traversant $m$ commutateurs en utilisant le théorème de la concaténation des nœuds (Le Boudec et Thiran, 2004).....	63
Figure 3.4. Vers un modèle de commutateur 802.1D (Georges, 2005).....	65
Figure 3.5. Modèle du commutateur incluant le protocole d'ordonnancement WRR.....	66
Figure 3.6. Courbe d'arrivée du trafic temps-réel dans le cas d'un lissage par seuil percé.....	67
Figure 3.7. Courbe de service offerte par un ordonnanceur WRR.....	69
Figure 3.8. Les 3 cas à distinguer dans le calcul du délai.....	74
Figure 3.9. Phénomène d'agrégation de flux.....	78
Figure 3.10. Scénario de flux emboîtés interférents (Schmitt et Zdarsky, 2006).....	80
Figure 3.11. Un exemple d'application de la méthodologie LUDB (Lenzini <i>et al.</i> , 2008).....	82
Figure 3.12. Courbes de services de type <i>rate-latency</i> et de type en escalier non uniforme (cas où $n = 2$ ).....	83
Figure 3.13. Architecture considérée pour la comparaison.....	86
Figure 3.14. Ethernet commuté à 10 Mb/s, avec une CdS WRR et avec un trafic de fond non synchronisé (Diouri <i>et al.</i> , 2008).....	91
Figure 3.15. Comportement du SCR en utilisant la configuration des poids basée sur le calcul réseau.....	93
Figure 4.1. Architecture utilisée pour l'identification du délai du trafic temps-réel souple.....	100
Figure 4.2. Définition linguistique du délai.....	103
Figure 4.3. Délai en fonction de $w_2$ à $w_1$ constants et avec une charge moyenne de 5 Mb/s.....	105
Figure 4.4. Délai en fonction de $w_2$ à $w_1$ constants et avec une charge moyenne de 6 Mb/s.....	105
Figure 4.5. Délai en fonction de $w_2$ à $w_1$ constants et avec une charge moyenne de 7 Mb/s.....	106
Figure 4.6. Description générale du système à identifier en utilisant le FRC (Bombardier et Schmitt, 2010)....	107
Figure 4.7. Fuzzification du poids $w_1$ en 3 termes avec des courbes trapézoïdales-triangulaires.....	109
Figure 4.8. Fuzzification du poids $w_2$ en 3 termes avec des courbes trapézoïdales-triangulaires.....	109
Figure 4.9. Le contrôleur flou est obtenu en inversant le modèle flou. Pour les systèmes qui varient dans le temps, le modèle peut être mis à jour dans la boucle (Babuška et Verbruggen, 1996).....	113
Figure 4.10. Commande floue dans le cas d'un délai inférieur à 7,644 ms.....	115
Figure 4.11. Ajout de la « statistic wire » permettant de récupérer la valeur du délai.....	117
Figure 4.12. Process model bridge_mac_relay_entity_v3.pr.m.....	118
Figure 4.13. Commutateur permettant de mettre un WRR dynamique sur OPNET (Diouri <i>et al.</i> , 2010).....	119
Figure 4.14. Variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du contrôleur vers le process.....	121
Figure 4.15. Variation des poids du commutateur 2 (figure 2.14) en fonction des délais de traversée de celui-ci par le flux allant du contrôleur vers le process.....	121
Figure 4.16. Variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du process vers le contrôleur.....	122

---

Figure 4.17. Variation des poids du commutateur 2 (figure 2.14) en fonction des délais de traversée de celui-ci par le flux allant du process vers le contrôleur.....	122
Figure 4.18. Délais subis aux niveaux du contrôleur et du process.....	123
Figure 4.19. Sortie du système .....	123



# Introduction générale

De nos jours, la plupart des systèmes industriels repose sur des architectures distribuées. Dans les milieux industriels, ces architectures peuvent se composer de plusieurs sous-systèmes autonomes coopérant entre eux *via* des réseaux de terrain. Ces systèmes distribués peuvent aussi être interconnectés avec d'autres en utilisant plus largement le réseau Internet pour permettre des applications de contrôle, de supervision ou d'e-maintenance sur des longues distances. Par comparaison avec les systèmes basés sur des architectures centralisées, la distribution a permis d'améliorer les applications de contrôle/commande industrielles en termes de réduction des coûts de câblage, de modularité, de flexibilité (Pleinevaux et Decotignie, 1988) et aussi en termes d'aide au diagnostic et à la maintenance des systèmes (Iung, 2002).

Cependant, les réseaux de communication utilisés dans les systèmes distribués génèrent en particulier des retards durant les échanges de données voire des pertes d'informations. Pour ce genre de système, la rupture d'un lien du réseau peut aussi affecter de manière fatale le système dans certains cas. Ces perturbations dépendent alors fortement du réseau utilisé (topologie, protocole de communication et politique d'ordonnancement, etc.) et de la charge du réseau (inter-arrivée et taille des paquets). En effet, des flux de type multimédia (utilisés généralement pour la surveillance) peuvent charger considérablement un réseau de communication et rentrer en conflit avec des échanges ayant des besoins/exigences plus importants. Ces applications à exigences temps-réel peuvent être dédiées notamment à la commande d'un système (comme la commande d'un process industriel, la téléopération, etc.). Pour cela, nous allons devoir hiérarchiser l'information en fonction de sa criticité et des exigences auxquelles elle est soumise. De cette manière, il sera possible de prendre ces perturbations en compte dans la conception des systèmes distribués de façon à pouvoir les maîtriser voire les atténuer pour satisfaire les contraintes du système à commander.

Généralement les délais induits par les réseaux ont des caractéristiques difficiles à maîtriser car ils sont liés à la complexité des architectures de communication et des équipements informatiques ainsi qu'aux échanges pseudo aléatoires transitant sur le réseau. Ainsi les recherches menées sur les Systèmes Contrôlés en Réseaux (SCR) (Branicky *et al.*, 2000) (Richard et Divoux, 2007) définis comme des systèmes de contrôle/commande distribués *via* un réseau de communication partagé (c'est-à-dire utilisé aussi pour transporter des informations générées par d'autres applications) reposent sur deux approches. La première approche consiste à adapter la commande relativement aux performances du système réseau (approche commande en réseau ou « *control over network* »). La deuxième approche consiste à adapter le réseau pour répondre aux besoins applicatifs (approche commande de réseau ou « *control of network* ») (Juanole et Mouney, 2007) (Mascolo, 2006) (Diouri *et al.*, 2007a) (Diouri *et al.*, 2010). Une autre façon de faire consiste à concevoir le système en intégrant les

deux approches précédentes, c'est la co-conception (ou *co-design*) (Branicky *et al.*, 2003). Dans cette vision intégrée, les outils de co-simulation ont un rôle déterminant (Branicky *et al.*, 2003) (Brahimi *et al.*, 2008). Dans cette thèse l'intérêt se porte sur l'approche commande de réseau car c'est une problématique relativement nouvelle dans le cadre des SCR et car plusieurs travaux ont déjà été faits dans le cadre de la commande en réseau.

Ces approches reflètent l'aspect pluridisciplinaire des SCR. En effet pour les étudier, il faut avoir des connaissances en automatique, en diagnostic, en réseau, en informatique, en traitement de signal, etc. Les experts se réunissent régulièrement au niveau international au sein du comité technique 1.5 de l'IFAC sur les *Networked Systems* et au niveau national dans le groupe de travail ARC (Automatique et Réseaux de Communication) du GDR MACS. Ces dernières années, des projets de recherche ont été consacrés à cette problématique. On peut citer le projet Européen NeCST (Networked Control Systems Tolerant to fault) et le projet ANR Safe-NeCS (Safe-Networked Control Systems) auxquels j'ai participé. Dans le cadre du projet NeCST, j'ai participé à la rédaction d'un livrable (Slacky *et al.*, 2006).

### Contexte de la thèse

Mon sujet de thèse s'inscrit dans le cadre des Systèmes Contrôlés en Réseaux (SCR). Les SCR sont composés d'actionneurs, de contrôleurs et de capteurs connectés par un réseau. Le contrôleur envoie des signaux de commande à l'actionneur tandis que le capteur envoie des signaux de mesures au contrôleur. Le réseau utilisé par le SCR peut être utilisé par d'autres applications et peut être lui-même, dans certains cas, commandé en fonction du délai et de l'erreur du système pour améliorer les performances de la commande du système (figure 1).

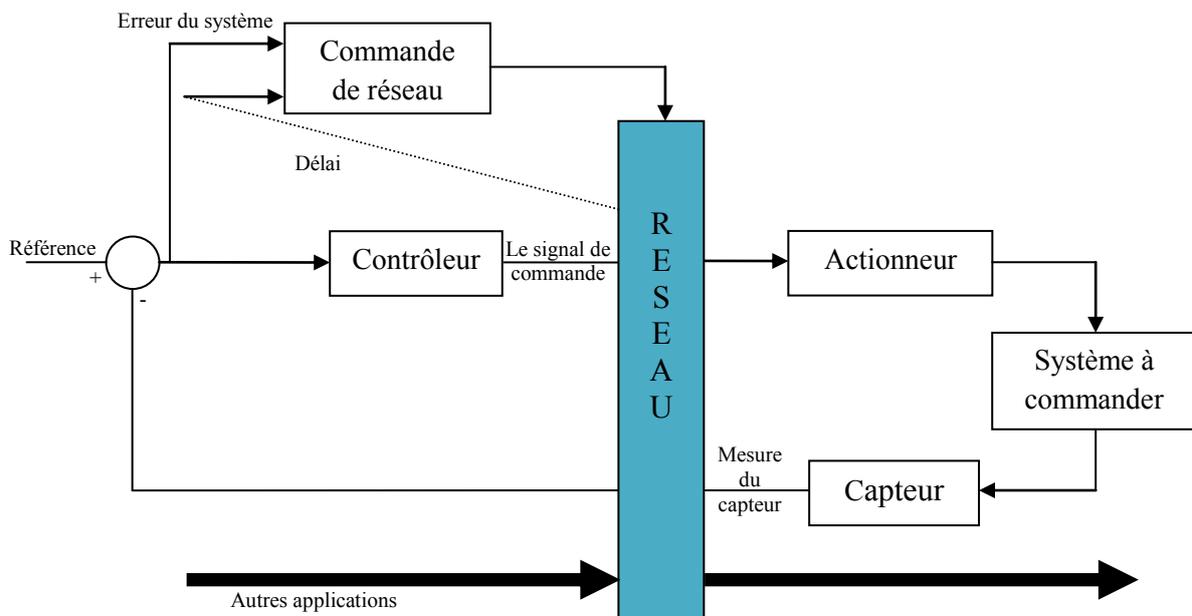


Figure 1. Réseau utilisé par un SCR et partagé avec d'autres types d'applications

Dans ces systèmes distribués temps-réel, il faut assurer la stabilité de la commande du process en tenant compte des performances du réseau utilisé. L'inconvénient majeur des SCR réside dans le fait que les performances du réseau (délai, perte de paquets, etc.) sont difficilement appréhendables, ce qui complique leur prise en compte dans la conception du système. De ce fait, le réseau peut affecter les performances du système à commander voire même conduire à son instabilité.

Initialement, les réseaux utilisés dans le domaine des SCR étaient des réseaux de terrain tels que FIP (Factory Instrumentation Protocol) (AFNOR, 1990) (EN50170), PROFIBUS (Process Field Bus (Nutzer Organisation e.v., 1992)) ou CAN (Controller Area Network) (ISO, 1993). Ces réseaux ont la particularité d'implémenter des protocoles adaptés aux contraintes temporelles des applications, mais présentent plusieurs inconvénients tels que leur coût et leur manque d'interopérabilité. Ainsi, depuis quelques années, il y a eu une tendance à remplacer les réseaux de terrain par le réseau Ethernet issu du monde informatique, malgré le non-déterminisme de son protocole d'accès au médium CSMA/CD (Carrier Sense Multiple Access with Collision Detection) (standard IEEE 802.3) (IEEE Computer Society, 2002). Ethernet offre plusieurs avantages : débits importants, utilisation de plus en plus répandue, etc. Actuellement, c'est plus précisément l'Ethernet commuté (norme IEEE 802.1D) (IEEE, 1998) qui devient de plus en plus utilisé car celui-ci a permis entre autres de réduire les domaines de collisions en comparaison avec l'Ethernet standard. Dans ma thèse, l'étude se concentrera donc sur les SCR s'appuyant sur des architectures Ethernet commutées d'autant plus que certains commutateurs Ethernet offrent la possibilité de mettre en place des protocoles d'ordonnancement qui permettent de satisfaire les contraintes temporelles des applications temps-réel.

Comme il a été précédemment dit, mes travaux s'inscrivent dans le cadre de l'approche commande de réseau (Kweon *et al.*, 1999) (Juanole et Mouney, 2007). L'objectif est d'optimiser, en fonction des trafics et en fonction des contraintes applicatives, les ressources de communication à allouer/réserver pour chaque trafic ou chaque type de trafic. Le réseau utilisé est l'Ethernet commuté et l'étude porte plus précisément sur l'ordonnanceur Weighted Round Robin (WRR) (Demers *et al.*, 1989) (Parekh, 1992) intégré dans les commutateurs Ethernet et qui est une implémentation pratique du Generalized Processor Sharing (GPS) (Parekh, 1992). Le WRR permet de partager la capacité d'un lien entre différentes classes de services et sera utilisé de manière plus générale pour optimiser le partage de la bande passante du réseau aux différents trafics tout en garantissant les contraintes temporelles de l'application.

De par la définition préalablement établie, il est clair que le réseau d'un SCR va héberger/supporter des applications présentant des contraintes temporelles plus ou moins variables. Dans cette thèse, l'intérêt se porte à la fois sur les applications ayant des contraintes temps-réel strictes ainsi que sur celles ayant des contraintes temps-réel souples. Pour chaque type d'applications temps-réel, il a fallu dans un premier temps définir un modèle mathématique du réseau, non nécessairement homogène avec les modèles de commande du

système distribué. Une originalité de ces travaux consistera à la poursuite d'une modélisation de tels réseaux. Deux techniques/théories seront utilisées pour la modélisation. À la recherche de la satisfaction de contraintes strictes, cette thèse propose l'application de la théorie du calcul réseau (Cruz, 1991a) (Cruz, 1991b) (Chang, 2000) (Le Boudec et Thiran, 2004) qui est utilisée pour définir des délais bornés. Lorsque les contraintes deviennent moins strictes, il sera proposé de recourir à la logique floue (Zadeh, 1965). Celle-ci permet d'établir des règles d'inférences qui mettent en relations les entrées sur lesquelles il est possible d'agir (le partage de la bande passante) avec la sortie, à savoir le délai dans notre cas. Les deux modèles ainsi établis serviront de base pour la définition d'une accommodation du réseau permettant de satisfaire les contraintes temporelles de l'application temps-réel tout en maximisant les ressources pour l'application non contrainte temporellement.

### Organisation du document

Ce document se compose de deux parties. La première partie, qui regroupe les deux premiers chapitres, présente le contexte de nos travaux à savoir les SCR (chapitre 1) ou plus précisément les SCR utilisant le réseau Ethernet (chapitre 2). La première partie comporte également un positionnement de nos travaux par rapport à la bibliographie. Pour le second chapitre, une attention particulière est apportée aux mécanismes de classification de service supportés par ce type de réseau. En effet, ce sont ces mécanismes qui vont permettre la construction d'une commande de réseau. Il s'agit là d'un point important des travaux de commande de réseau qui restent fortement liés/dépendants à la technologie supportée par le système réel. Dans ce cadre, nous nous intéresserons aux mécanismes d'ordonnancement équitables qui ont la faculté de rester légers, et ainsi, aisément implantables.

La deuxième partie, quant à elle, concerne les contributions de la thèse. Cette deuxième partie se compose, à son tour, de deux sous-parties (chapitres 3 et 4). Le chapitre 3 traite les applications ayant des exigences strictes (chapitre 3). Le chapitre 4 traite, quant à lui, les applications ayant des contraintes temps-réel souples (chapitre 4).

Dans le chapitre 1, l'accent est mis sur l'impact des délais sur les SCR. Ce premier chapitre présente les différentes approches pour stabiliser les SCR, à savoir : l'approche commande en réseau et l'approche commande de réseau. Certains se sont intéressés à concevoir le système en intégrant les deux approches précédentes, c'est la co-conception (ou *co-design*) (Branicky *et al.*, 2003). Dans le chapitre 1, on explique également pourquoi l'utilisation du réseau Ethernet commuté est intéressante dans le cas des SCR.

Dans le second chapitre, l'intérêt se porte sur le réseau Ethernet et plus particulièrement sur le réseau Ethernet commuté. Ce chapitre correspond à la pré-étude de cette thèse de façon à mieux comprendre les effets du réseau sur les SCR en termes de bande passante, de protocoles d'accès au médium (CAN, Ethernet), d'ordonnanceur, de charge, avant d'engager des travaux plus théoriques présentés dans les chapitres suivants. Cette analyse est réalisée sur une plateforme expérimentale et sur plusieurs simulateurs qui ont par ailleurs nécessité le

---

développement de nouveaux modules pour pouvoir réaliser des évaluations de performances propres à notre étude.

Le chapitre 3 concerne les applications temps-réel strict. On entend par temps-réel strict, des systèmes qui doivent respecter impérativement les contraintes temporelles définies par l'application à la fois par rapport aux temps de remontée des informations des capteurs vers le contrôleur mais aussi par rapport aux temps de mise en action des commandes. Dans ce chapitre, nous faisons une présentation de la théorie du calcul réseau (Le Boudec et Thiran, 2004). Ensuite, le calcul réseau est utilisé pour modéliser les délais maxima de bout en bout pour des architectures Ethernet commutées à classification de service WRR. Cette modélisation du délai sert ensuite à établir une configuration du réseau (hors-ligne) permettant de maximiser la bande passante des flux non contraints temporellement tout en garantissant les contraintes temporelles des applications temps-réel strict.

Dans le chapitre 4, on se focalise sur les applications temps-réel souple qui sont considérées dans cette recherche comme des systèmes autorisant une « certaine » tolérance dans le non respect des contraintes temporelles applicatives. Cette tolérance n'est pas formellement étudiée dans cette thèse qui s'attache surtout à formaliser le délai de communication et les mécanismes de reconfiguration du réseau. Néanmoins, nous considérons que dans cette fenêtre de tolérance, on doit être capable de détecter un dépassement de seuil du délai, et de redescendre sous ce seuil en activant une reconfiguration du réseau. A noter que durant cette reconfiguration du réseau, celui-ci doit rester opérationnel. Dans ce chapitre, nous commençons par présenter la théorie de la logique floue (Zadeh, 1965) (Bouchon Meunier, 1995). La logique floue est ensuite utilisée pour modéliser le délai de traversée d'un commutateur Ethernet implémentant le protocole d'ordonnancement WRR. Cette modélisation du délai permet de mettre en place une commande de réseau pour des applications temps-réel souple. Enfin, une discussion sur la portée des deux stratégies (chapitres 3 et 4) proposées dans cette thèse sera également présentée.



**Partie I :**

**Positionnement**



# Chapitre 1 :

## Les systèmes contrôlés en réseaux

### 1.1. Les systèmes contrôlés en réseaux

#### 1.1.1. Introduction

Dans les années 70, Honeywell a initié le concept des SCR en utilisant le Distributed Control System (DCS). Les SCR se composent généralement de contrôleurs, de capteurs et d'actionneurs. Le capteur transmet la mesure au contrôleur, ce dernier la compare à la consigne (valeur souhaitée) et l'utilise comme entrée de sa loi de commande. Ensuite, le contrôleur calcule la commande et la transmet à l'actionneur. Le réseau utilisé par les SCR peut être partagé avec d'autres applications indépendantes de la commande du système et peut être lui-même, quelques fois et sous certaines conditions, commandé en fonction par exemple du délai et/ou de l'erreur du système pour améliorer les performances de la commande du système (figure 1.1). La figure 1.1 montre les deux types de commandes : commande en réseau et commande de réseau. Comme le montre la figure 1.1, il existe une autre manière qui consiste à concevoir le système en intégrant les deux approches précédentes, c'est la co-conception (Branicky *et al.*, 2003). Il existe dans la littérature d'autres variantes de ce schéma (figure 1.1).

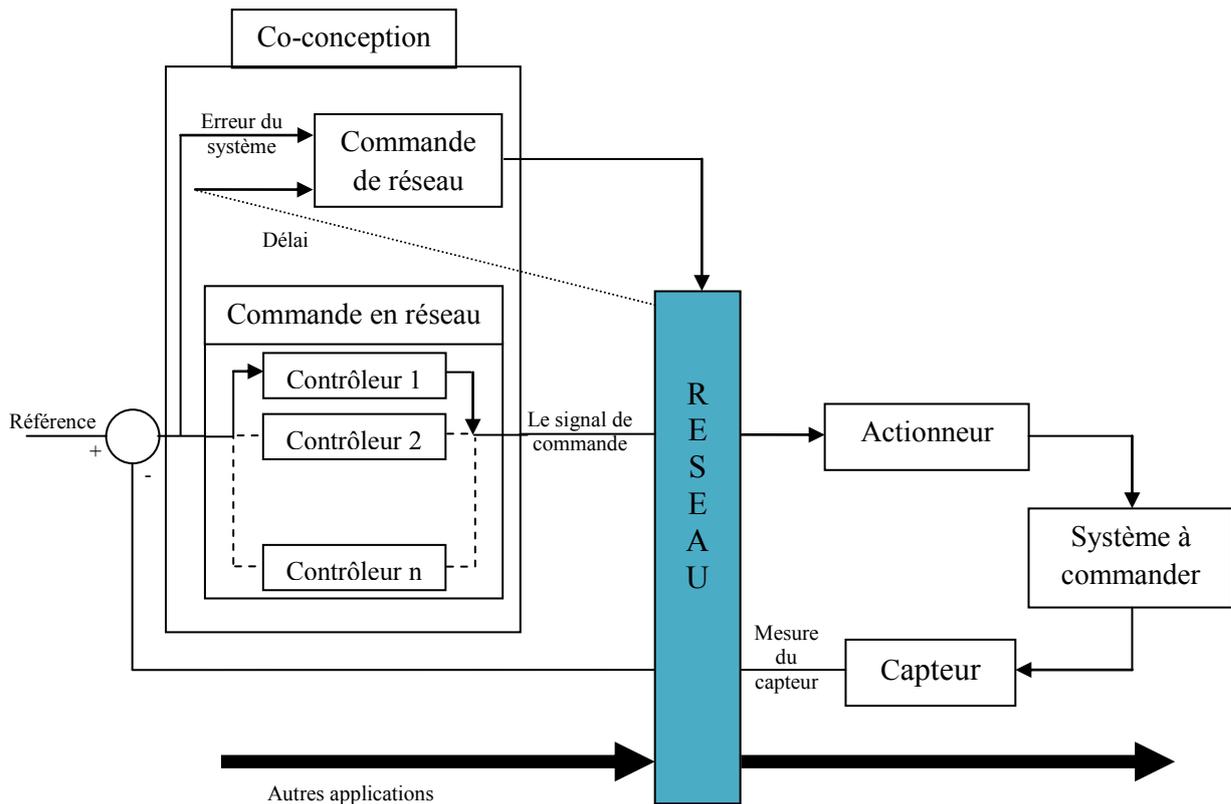


Figure 1.1. Réseau support d'un SCR et partagé avec d'autres types d'applications

Comme précédemment mentionné, l'utilisation des réseaux dans les SCR présente plusieurs avantages (réduction des coûts de câblage, etc.). De ce fait, on les rencontre de plus en plus dans les applications industrielles. On les trouve dans plusieurs domaines comme les systèmes embarqués (drone (Diouri *et al.*, 2008)), l'industrie automobile (Ben Gaid *et al.*, 2006), etc. Les SCR se retrouvent aussi dans le contexte de la téléopération (Meng *et al.*, 2004) (Tipsuwan et Chow, 2004) (Fraisse et Lelevé, 2003).

Les SCR peuvent utiliser les réseaux filaires et/ou les réseaux sans fil (Stubbs et Dullerud, 2001) (Boughnami *et al.*, 2009). De nos jours, dans les SCR utilisant les réseaux filaires, l'Ethernet (IEEE computer Society, 2002) et plus précisément l'Ethernet commuté (norme IEEE 802.1D) (IEEE, 1998) devient de plus en plus courant, remplaçant ainsi des réseaux industriels traditionnels tels que CAN (Controller Area Network (ISO, 1993)) ou PROFIBUS (Process Field Bus (Nutzer Organisation e.v. 1992)). De manière générale, le choix du réseau, sa topologie et ses caractéristiques auront un impact sur les performances du SCR que l'on appelle la Qualité de Service (QoS) (ou en anglais Quality of Service (QoS)). Les indicateurs de la QoS les plus étudiés sont le délai, la gigue ou encore le taux de perte de paquets.

### 1.1.2. Impact du délai sur les SCR

Les délais induits par le réseau constituent un problème majeur dans les SCR car ils peuvent dégrader les performances de la commande du système voire même conduire à l'instabilité de

celui-ci (Zhang *et al.*, 2001). De ce fait, les hypothèses idéales, telles que l'absence de délais aux niveaux des capteurs et des actionneurs, sur lesquelles se basent les théories de commande conventionnelles sont à reconsidérer afin qu'elles puissent être appliquées dans les SCR (Zhang *et al.*, 2001).

Les délais de bout en bout entre le contrôleur et le process et vice-versa se composent de plusieurs délais (le temps de transmission, le temps de traversée des couches du modèle OSI, le temps de calcul de la commande, etc.) dont certains peuvent être variables. Ces délais induits par le réseau dépendent fortement du protocole de communication ainsi que du trafic de fond. Concernant le délai de calcul, (Yang, 1990) a présenté une analyse et un exemple dans lequel le délai du contrôleur numérique peut causer l'instabilité du système.

Actuellement, et par soucis économiques, de plus en plus de réseaux utilisés dans les SCR sont « partageables » c'est à dire qu'à n'importe quel moment, il est possible que d'autres flux rentrent en concurrence avec ceux qui sont contraints temporellement. De ce fait, ces réseaux, sont soumis aux mêmes contraintes que les « réseaux ouverts ». En effet, la présence de flux non maîtrisés qui sont générés par certaines applications dans le réseau peut dégrader de manière impromptue la qualité de service offerte aux messages temps-réel.

Dans le domaine des SCR, il est nécessaire de prendre en compte les délais induits par le réseau (Lian *et al.*, 2001) dès la conception de la loi de commande (Lian, 2001) (Cervin *et al.*, 2003). Dans ce contexte, (Lian *et al.*, 2001) ont comparé les délais induits par les réseaux Ethernet, ControlNet et DeviceNet. (Lian *et al.*, 2001) ont ensuite constaté que malgré le fait qu'Ethernet fournit un débit plus important, il reste un réseau non déterministe à cause de sa méthode d'accès au médium. Ainsi, (Lian *et al.*, 2001) ont conclu que pour les communications contraintes, l'utilisation de ControlNet et/ou de DeviceNet (à savoir des réseaux de terrain conventionnels) est plus adéquate.

Les études portant sur la conception et l'analyse des systèmes à retard (Richard, 2003) (Seuret *et al.*, 2004) (Michiels *et al.*, 2005) donnent des résultats qui peuvent être appliqués dans le contexte des SCR. Dans (Richard, 2003), l'auteur a proposé un état de l'art sur les systèmes à retard et il a indiqué les champs de recherche associés aux SCR comme les systèmes à retard avec des non-linéarités importantes ou les systèmes ayant des délais variables dans le temps.

Parmi les travaux remarquables dans le cadre des SCR, (Nilsson, 1998) a analysé les SCR dans le domaine discret. (Nilsson, 1998) a étudié deux modèles de délai : les délais aléatoires indépendants et les délais aléatoires avec des fonctions de distribution de probabilité régie par une chaîne de Markov sous-jacente. (Nilsson, 1998) a utilisé ces deux modèles de délais pour concevoir une loi de commande optimale. (Walsh *et al.*, 1999) ont introduit un nouveau protocole de commande de réseau, Try-Once-Discard (TOD), et ils se sont basés sur la simulation pour comparer la performance du nouveau protocole réseau avec les protocoles d'ordonnancement statiques. (Walsh *et al.*, 1999) ont défini le Maximum Allowable Transfer Interval (MATI) et leur objectif a été de définir le MATI qui garantit la performance désirée

du SCR. (Branicky *et al.*, 2000) ont tenu compte du délai induit par le réseau dans la boucle de commande en utilisant la figure suivante (figure 1.2).

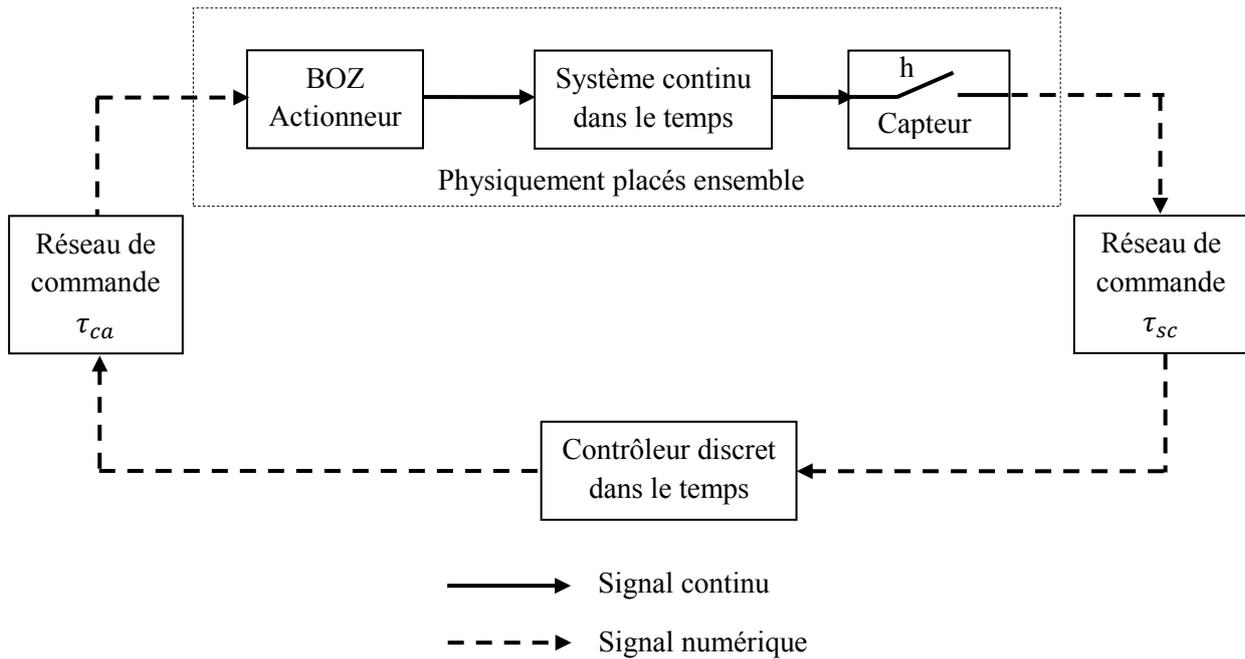


Figure 1.2. Organisation du SCR (Branicky *et al.*, 2000)

Comme le montre la figure 1.2, le SCR considéré par (Branicky *et al.*, 2000) se compose d'un système continu. Ils ont considéré deux configurations, la première utilise des capteurs à fonctionnement périodique, ainsi qu'un contrôleur et un actionneur de type événementiel. Dans la deuxième configuration, ils ont considéré un contrôleur périodique tel que dans (Halevi et Ray, 1988). (Branicky *et al.*, 2000) ont étudié le cas où le délai est inférieur à une période d'échantillonnage ainsi que le cas où le délai est supérieur ou égal à une période d'échantillonnage.

(Nilsson, 1998) ont étudié le système en boucle fermée en utilisant le schéma suivant (figure 1.3). Certains auteurs comme (Nilsson, 1998), tiennent compte du délai de calcul au niveau du contrôleur  $\tau_k^c$  (figure 1.3).

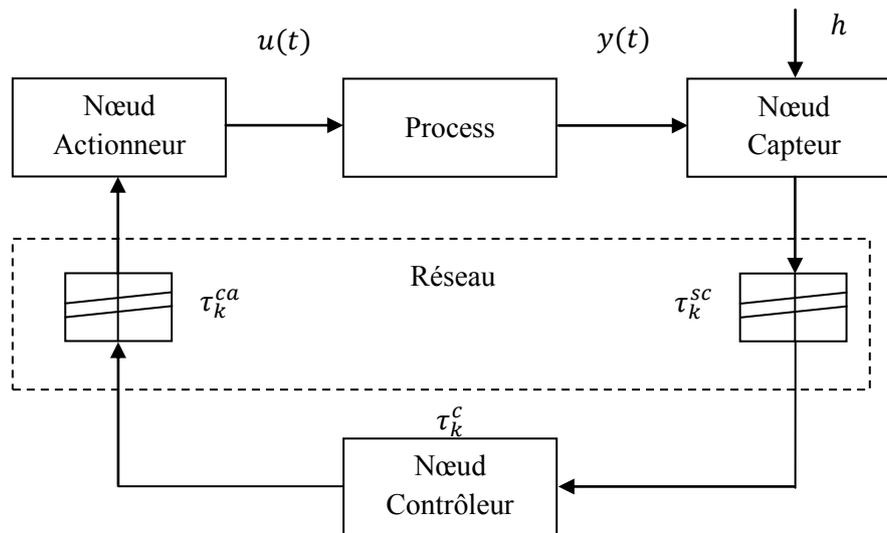


Figure 1.3. Système de commande numérique distribué avec des délais induits,  $\tau_k^{sc}$  et  $\tau_k^{ca}$ . Le délai de calcul dans le nœud du contrôleur,  $\tau_k^c$ , est aussi indiqué (Nilsson, 1998)

Les travaux présentés ci-dessus ne donnent qu'un aperçu des premières recherches dans la thématique des SCR. Aujourd'hui, un grand nombre de propositions sont apparues (on pourra d'ailleurs se référer aux états de l'art présentés dans (Richard, 2003) (Tipsuwan et Chow, 2003). Avant de continuer cette étude de l'état de l'art, une classification des propositions va alors être proposée.

### 1.1.3. Conclusion

L'étude préliminaire précédente nous montre que la majorité des travaux se focalisent sur la prise en compte des délais induits par le réseau en vue d'assurer la stabilité des process à commander. C'est d'ailleurs le paramètre de QdS que nous retiendrons dans la suite de la thèse. Pour cela, nous distinguerons deux types d'approches pour les SCR. Nous présenterons également une autre manière de traiter les SCR qui consiste à concevoir le système en intégrant les deux approches précédentes.

La première approche consiste à adapter la commande relativement aux performances du système réseau, c'est l'approche commande en réseau. La deuxième, consiste à considérer que dans la mesure où c'est le réseau qui est responsable de l'apparition de ces retards, et que l'on cherche à éviter une dégradation de la performance de la qualité de commande, l'adaptation portera sur le réseau, c'est l'approche commande de réseau (Juanole et Mouney, 2006) (Juanole et Mouney, 2007) (Mascolo, 2006) (Diouri *et al.*, 2007a). D'autres travaux ont porté sur la conception du SCR en intégrant les deux approches précédentes, c'est la co-conception (Branicky *et al.*, 2003) (Sename *et al.*, 2003). Dans (Branicky *et al.*, 2003) (Brahimi *et al.*, 2008), l'accent a été mis notamment sur le rôle déterminant des outils de co-simulation.

## 1.2. Approches appréhendant la QdC des SCR

Ce paragraphe vise à présenter comment chacune des approches définies précédemment permet d'améliorer les paramètres de QdC comme par exemple la stabilité du système. Les deux approches possibles pour assurer la stabilité des SCR ainsi que la co-conception sont détaillées successivement.

### 1.2.1. Approche commande en réseau

Les deux principaux problèmes des SCR sont les délais induits par le réseau et la perte de paquets. En effet, ces deux types de perturbations peuvent mener à l'instabilité du système commandé. L'approche commande en réseau consiste généralement à analyser la stabilité du process dans un SCR en considérant les performances du réseau, mais sans modifier ses paramètres : on subit les perturbations du réseau. Il s'agit donc de concevoir des lois de commande en tenant compte des performances du réseau utilisé (délai, gigue et/ou perte de paquets). Un état de l'art présentant les différentes méthodologies sur la commande en réseau a été présenté dans (Tipsuwan et Chow, 2003) (Zampieri, 2008).

De façon générale, les travaux menés dans le contexte des systèmes à retard (Richard, 2003) (Michiels *et al.*, 2005) peuvent être appliqués dans le contexte des SCR. La particularité des SCR repose alors sur les difficultés à mesurer ou à établir un modèle du délai induit par le réseau.

Dans la littérature, différentes propositions se sont différenciées sur la connaissance du retard. Ainsi, (Boukas et Liu, 2001) (Seiler et Sengupta, 2005) ont modélisé le délai comme étant un processus Markovien et cela dans le but de concevoir une loi de commande. (Goktas, 2000) (Yue *et al.*, 2005) (Vatanski *et al.*, 2006) ont considéré les bornes des délais induits par le réseau et ils ont proposé des méthodes basées sur la commande robuste. (Goktas, 2000) a basé son approche sur les bornes (maximale et minimale) du délai. (Yue *et al.*, 2005) ont utilisé l'information relative à la borne minimale du délai. (Vatanski *et al.*, 2006) ont conçu un contrôleur robuste basé sur la borne maximale qui a été calculée dans le cas des réseaux Ethernet commutés (Georges *et al.*, 2005).

(Hong, 1995), quant à lui, a proposé une méthode qui consiste à sélectionner la période d'échantillonnage appropriée et cela afin de compenser les effets du délai. Dans sa méthode nommée « *optimal stochastic control methodology* », (Nilsson, 1998) a traité les délais aléatoires comme étant un problème LQG (Linear Quadratic Gaussian).

Dans les travaux précédents, il est nécessaire que la modélisation des délais soit valable, ce qui peut en limiter la portée (notamment le type de réseau considéré). Dans le cas où des solutions de métrologie des délais sont possibles, il existe des méthodes utilisant entre autres le prédicteur de Smith (Bauer *et al.*, 2001) (Vatanski *et al.*, 2006). La figure 1.4 montre le

schéma du prédicteur de Smith adopté par (Vatanski *et al.*, 2006) afin de compenser les délais induits par le réseau. Ici, les retards sont supposés mesurables grâce à une solution de synchronisation des horloges PTP (standard IEEE 1588) (Eidson, 2006).

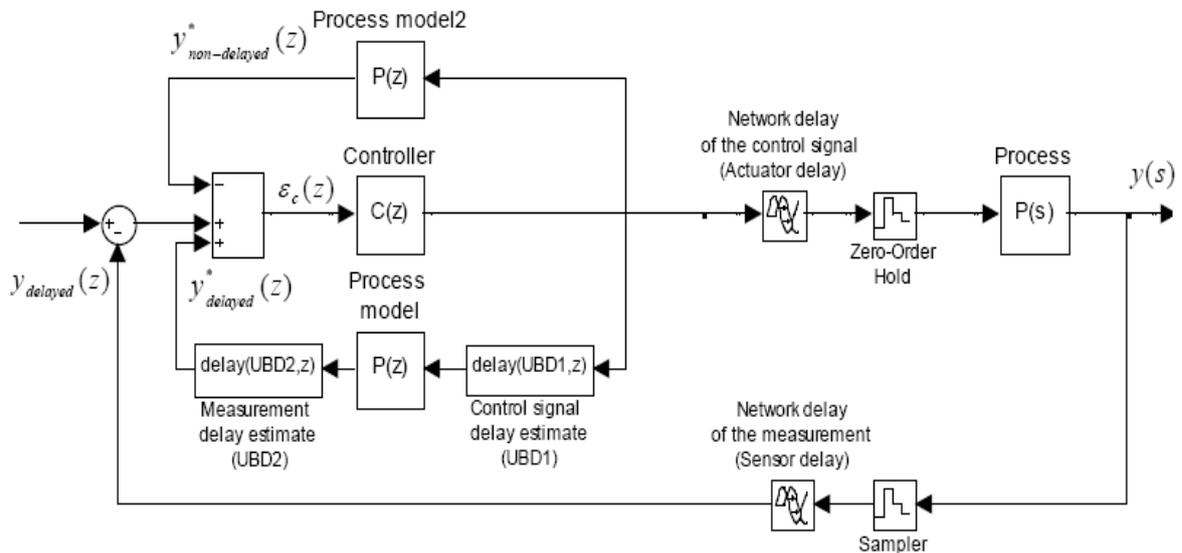


Figure 1.4. Schéma d'un prédicteur de Smith pour la compensation du délai induit par le réseau (Vatanski *et al.*, 2006)

Afin de faire face à la difficulté d'appréhension de la variation des retards, des approches par apprentissage ont été proposées. Ainsi, (Lin *et al.*, 2008), ont utilisé un schéma de compensation qui se base sur un réseau de neurones et sur un contrôleur PID flou dans le but d'atteindre la consigne. (Lin *et al.*, 2008) ont utilisé un prédicteur de Smith basé sur les réseaux de neurones (figure 1.5) et cela pour assurer une meilleure compensation des délais comparativement à un simple prédicteur de Smith.

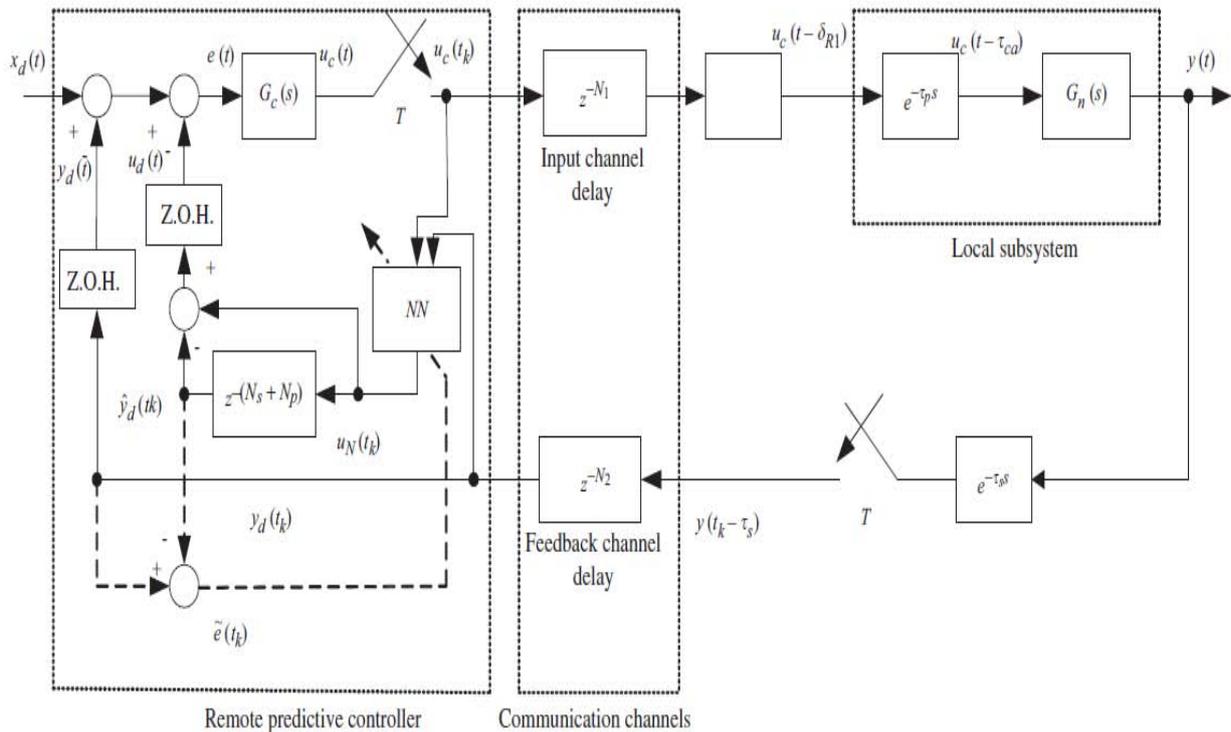


Figure 1.5. Stratégie prédictive basée sur les réseaux de neurones pour les SCR avec des délais (Lin *et al.*, 2008)

(Kim *et al.*, 2004) ont modélisé un SCR comme étant un système à commutation déterministe. (Kim *et al.*, 2004) ont proposé deux types de contrôleur : le contrôleur dépendant du mode précédent « *previous mode dependent controller* » et le contrôleur indépendant du mode précédent « *previous mode independent controller* », où un mode représente un des états auquel le contrôleur peut appartenir. (Kim *et al.*, 2004) ont basé leur étude sur un type de fonction de Lyapunov pour chaque type de contrôleur. Récemment, (Kim et Park, 2009) ont considéré des modèles multiples pour le trafic du réseau et ont construit un SCR efficace qui en dépend. (Kim et Park, 2009) ont amélioré les performances  $H_\infty$  obtenues par (Kim *et al.*, 2004) en employant des fonctions de Lyapunov multiples associées avec le mode courant au lieu du mode précédent ainsi qu'en diminuant l'aspect arbitraire dans la transition du mode et cela *via* le modèle de chaîne de Markov avec des modèles multiples.

Dans le cadre de l'approche commande en réseau, certains auteurs ont également considéré le système distribué comme étant un système multi-agents (Zampieri, 2008) (Olfati-Saber *et al.*, 2007). Leur proposition consiste à utiliser une commande distribuée afin de réduire l'impact des perturbations dues au réseau sur le système global.

Pour les approches qui tiennent compte de la perte de paquets, le lecteur peut se référer à (Zhang *et al.*, 2001) (Azimi-Sadjadi, 2003) (Yu *et al.*, 2004) (Xiong et Lam, 2007) (Berbra *et al.*, 2008). Dans (Zhang *et al.*, 2001), l'intérêt s'est porté sur la stabilité des SCR avec un certain taux de perte de paquets. (Azimi-Sadjadi, 2003) s'est basé sur le « principe du seuil incertain » pour montrer que sous certaines conditions, il existe un taux de perte de paquets

pour lequel le SCR est stable. Dans (Yu *et al.*, 2004), les auteurs ont étudié la stabilité des SCR en présence de délais et ils suggèrent d'écarter des paquets avec un certain taux pour préserver la bande passante du réseau tout en maintenant la stabilité des SCR. Dans (Xiong et Lam, 2007), les auteurs ont présenté les conditions de stabilité des SCR dans le cas d'une perte de paquets bornée. (Xiong et Lam, 2007) ont considéré le cas d'une perte de paquets arbitraire ainsi que le cas d'une perte de paquets qui suit le processus de Markov. Dans (Berbra *et al.*, 2008), l'aspect « diagnostic » a été pris en compte et une attention particulière a été accordée à la perte de paquets. Ainsi, ils ont proposé un nouvel indicateur sensible à ce problème afin de différencier les changements de résidus dus aux fautes ou à la perte de paquets. Dans (Berbra *et al.*, 2008), le réseau utilisé est le CAN (Controller Area Network) et le système (un drone) est commandé grâce à un observateur d'état non-linéaire qui maintient la valeur de la commande précédente dans le cas où il ne reçoit pas toutes les données nécessaires pour son calcul. Dans ce cas là, (Berbra *et al.*, 2008) ont proposé de procéder à une reconfiguration qui mène à un banc d'observateurs/estimateurs dont le choix dépend des données disponibles.

D'autre part, dans (Mechraoui *et al.*, 2009), l'influence de la perte de paquets sur la qualité de commande du système (robot Khepera) a été étudiée dans le cas du réseau sans-fil. (Mechraoui *et al.*, 2009) ont proposé une approche basée sur les réseaux Bayésiens qu'ils ont utilisés comme un critère de décision pour commuter entre deux types de contrôleurs.

Ce paragraphe donne un aperçu des travaux menés dans le contexte d'une commande en réseau. L'un des points de discrimination de chacune de ces approches concerne l'identification des retards : par modélisation ou par mesure. De fortes variations des retards peuvent alors conduire à la dégradation des performances de la commande.

### **1.2.2. Approche commande de réseau**

Dans le contexte des SCR, la commande de réseau consiste à adapter le réseau dans le but de satisfaire les contraintes de l'application. Pour être en mesure de régler les paramètres de communication, il est nécessaire de connaître les caractéristiques du trafic de chaque flux temps-réel relativement à l'application. Par exemple, les échéances d'un flux peuvent être fonction d'un temps de cycle automate.

Dans la suite, différents travaux sont présentés. Il est peut être d'ores et déjà à noter que chaque solution est extrêmement liée au contexte technologique. En effet, puisque cette approche consiste à modifier les paramètres d'un protocole ou d'un ordonnancement, chaque proposition requiert la présence d'un mécanisme d'actionnement spécifique.

Dans l'approche commande de réseau, certains auteurs ont basé leur étude sur le contrôle de la congestion pour adapter le débit d'entrée à la bande passante disponible (Jacobson, 1988) (Kelly *et al.*, 1998) (Floyd et Fall, 1999) (Altman *et al.*, 1999). (Altman *et al.*, 1999) ont

conçu un contrôleur stochastique de type LQG (Linear Quadratique Gaussian) pour le contrôle de congestion. Pour concevoir la loi de contrôle de congestion, (Mascolo, 1999) a utilisé le principe de Smith. La loi de commande proposée dans (Mascolo, 1999) garantit qu'il n'y a pas de pertes de paquets dans le réseau et une utilisation équitable et complète des liens que ce soit en régime transitoire ou en régime permanent. Dans (Mascolo, 2006), le contrôle du flux et de la congestion TCP ont été modélisés comme un système à retard et il a été montré que le principe de « *self-clocking* » (quand un paquet sort, un nouveau entre dans le réseau), correspond à l'implémentation d'un prédicteur de Smith plus un proportionnel simple.

(Floyd et Fall, 1999) ont justifié le besoin d'un contrôle de congestion de bout en bout et ils ont aussi montré l'importance de mécanismes dans le réseau pour détecter et restreindre les flux *best effort* durant les périodes de congestion.

(Tang *et al.*, 2007) ont proposé un modèle de flux fluide et précis, et ensuite ils ont utilisé leur modèle pour l'analyse de stabilité du FAST TCP. Dans des études récentes sur les réseaux sans fil, le contrôle de congestion (Zheng et Nelson, 2007) et le contrôle de flux (Abate *et al.*, 2006) sont analysés.

(Kweon *et al.*, 1999) ont proposé la technique de lissage de trafic qui permet de fournir des garanties de délai statistiques sur Ethernet. (Kweon *et al.*, 2000) ont proposé un mécanisme d'adaptation nommé « Harmonic-Increase and Multiplicative-Decrease » (HIMD) qui réagit avec la détection de collision sur une période de temps spécifique. Dans leur approche (Kweon *et al.*, 1999) (Kweon *et al.*, 2000), chaque nœud utilise une méthode de lissage de trafic pour contrôler le trafic.

(Carpinzano *et al.*, 2002) ont proposé une méthode de lissage basée sur la logique floue avec deux entrées (le nombre de collisions et le débit global) observées dans un intervalle de temps de référence donné et avec une seule sortie (la variation de la période de rafraîchissement). (Carpinzano *et al.*, 2002) ont montré que les délais aller-retour obtenus avec l'approche floue sont inférieurs à ceux obtenus avec la méthode HIMD (Kweon *et al.*, 2000). Afin d'améliorer leur méthode de lissage floue, (Lo Bello *et al.*, 2005) ont utilisé les algorithmes génétiques pour l'optimisation de celle-ci. La méthode de lissage de trafic floue proposée par (Lo Bello *et al.*, 2005) permet de fournir une borne de délai statistique sur le réseau Ethernet.

Dans (Song, 2009), quelques travaux récents dans le cadre de l'adaptation de la QoS ont été présentés. (Juanole et Mouney, 2006) ont montré l'impact de la QoS sur la stabilité de la commande du process dans les SCR. Afin de montrer l'influence de la QoS, (Juanole et Mouney, 2006) ont analysé l'ordonnancement des messages dans le réseau CAN (avec un schéma de priorité). Dans (Juanole et Mouney, 2007), le concept de priorité hybride a été proposé en utilisant le réseau CAN. Dans (Juanole et Mouney, 2007), les bits d'identification ont été séparés en deux champs (un pour la priorité statique et l'autre pour la priorité dynamique). (Juanole et Mouney, 2007) ont proposé d'utiliser la priorité statique dans le cas

normal et d'avoir recours au champ de priorité dynamique s'il y a des messages contraints temporellement. (Koubâa, 2004) a proposé une nouvelle politique d'ordonnancement nommée (m,k)-WFQ qui tient compte des contraintes temps-réel du (m,k)-firm (Hamdaoui et Ramanathan, 1995) dans le WFQ conventionnelle. Cette nouvelle politique d'ordonnancement proposée par (Koubâa, 2004) assure que m sur k paquets consécutifs doivent respecter leurs contraintes de délais (puisque le flux est sous contrainte (m,k)-firm) et donne aussi des garanties en termes de bande passante allouée à tous les flux et cela grâce au WFQ.

Dans (Boughnami *et al.*, 2009), l'intérêt s'est porté sur l'IEEE 802.15.4/ZigBee et l'étude a concerné deux mécanismes de gestion de la QoS utilisant la priorité probabiliste d'une part et déterministe d'autre part. Dans (Boughnami *et al.*, 2009), le mécanisme utilisant la priorité déterministe a été adopté afin de permettre d'adapter dynamiquement la QoS du réseau en donnant une priorité plus haute au flux de commande dans le cas où la qualité de commande du système contrôlé est insuffisante. (Boughnami *et al.*, 2009) ont réalisé leur adaptation en modifiant le paramètre macMinBE du protocole MAC de l'IEEE 802.15.4. L'approche proposée par (Habib *et al.*, 2009) consiste à adapter la période du *backoff* en fonction de la priorité, c'est-à-dire en fonction des contraintes temporelles.

La priorité stricte et l'Earliest Deadline First (EDF) (Liu et Layland, 1973) sont aussi des politiques d'ordonnancement classiques qui sont utilisées dans l'approche commande de réseau. (Pedreiras *et al.*, 2003) ont caractérisé le comportement du trafic temps-réel dans le cas d'un réseau Ethernet commuté à priorités. Dans (Georges, 2005) la borne maximale du délai a été calculée dans le cas des réseaux Ethernet commutés implémentant la priorité stricte.

Dans (Brahimi *et al.*, 2008) (Aubrun *et al.*, 2010), un SCR basé sur une architecture Ethernet commutée a été modélisé en utilisant les réseaux de Petri de haut niveau et a été simulé en utilisant le HTCPN (Hierarchical Timed Colored Petri Nets). Le modèle proposé dans (Brahimi *et al.*, 2008) tient compte du comportement du commutateur Ethernet et de l'environnement applicatif du SCR. Dans (Brahimi *et al.*, 2008), une stratégie basée sur le WRR a été proposée pour adapter la qualité de service en tenant compte de la qualité de commande requise par l'application. Le principe de la stratégie proposée dans (Brahimi *et al.*, 2008) est le suivant : si le délai des trames est supérieur au seuil MDT (Making Decision Threshold) évalué alors les mécanismes d'ordonnancement sont activés pour réduire le délai. Les résultats de simulations faites par (Brahimi *et al.*, 2008) ont montré que la stratégie d'adaptation de la qualité de service a permis d'améliorer les performances du système global.

(Lian *et al.*, 2004) ont proposé une commande de commutateur Ethernet dans le but de maintenir la QoS globale tout en garantissant les performances du système de commande. Dans (Lian *et al.*, 2004), quand le commutateur reçoit un ensemble de paquets ayant des exigences, celui-ci calcule l'utilisation totale nécessaire à tous les paquets soumis à des exigences. (Lian *et al.*, 2004) se sont basés sur ce calcul pour distinguer les deux cas

suivants : si l'ordonnancement est faisable alors la politique EDF est adoptée au niveau du commutateur Ethernet pour dispatcher l'ensemble des paquets qui viennent d'arriver, sinon une loi de commande active de la file du commutateur est utilisée pour répartir la bande passante disponible du réseau.

(Hoang *et al.*, 2002) ont également utilisé le protocole d'ordonnancement EDF dans le cadre des réseaux Ethernet commutés pour permettre des garanties de délais. Dans (Hoang *et al.*, 2002), le contrôle d'admission se fait au niveau du commutateur et le protocole d'ordonnancement EDF est adopté au niveau du commutateur et des stations finales. Dans (Hoang *et al.*, 2002), les nœuds connectés au commutateur peuvent satisfaire leurs contraintes temporelles même en présence de communications Internet, ce qui peut permettre des applications telles que la maintenance à distance.

### 1.2.3. Vers une conception conjointe

La co-conception (ou *co-design* en anglais) consiste à concevoir le système en intégrant les deux approches précédentes. C'est-à-dire qu'il faut pouvoir agir à la fois sur le réseau pour qu'il satisfasse les contraintes applicatives (en termes de délais, de taux de pertes de paquets, etc.) et à la fois sur la commande du process pour que le système reste stable même en présence de perturbations au niveau du réseau.

Pour concevoir de telles solutions, un environnement de simulation commun doit permettre de déterminer le gain de chaque stratégie. Dans les travaux de (Branicky *et al.*, 2003), l'idée de recourir à la co-simulation pour faire de la co-conception des SCR a été introduite. Dans ce sens, la boîte à outils TrueTime (TrueTime *Toolbox*) a été développée (Andersson *et al.*, 2005) dans l'environnement Matlab et cela afin d'étudier l'impact du réseau sur le système de contrôle/commande. Dans (Brahimi *et al.*, 2008), l'accent a aussi été mis sur le rôle des outils de co-simulation pour l'étude de cette approche en se basant sur les réseaux de Petri. Des travaux de co-simulation ont également été conduits au sein de notre équipe de recherche au CRAN pour simuler le comportement du SCR sur le logiciel OPNET (OPNET, 2005).

Dans la figure 1.6, (Branicky *et al.*, 2003) montre que la co-conception commence par l'expression des contraintes relatives à la commande et au réseau.

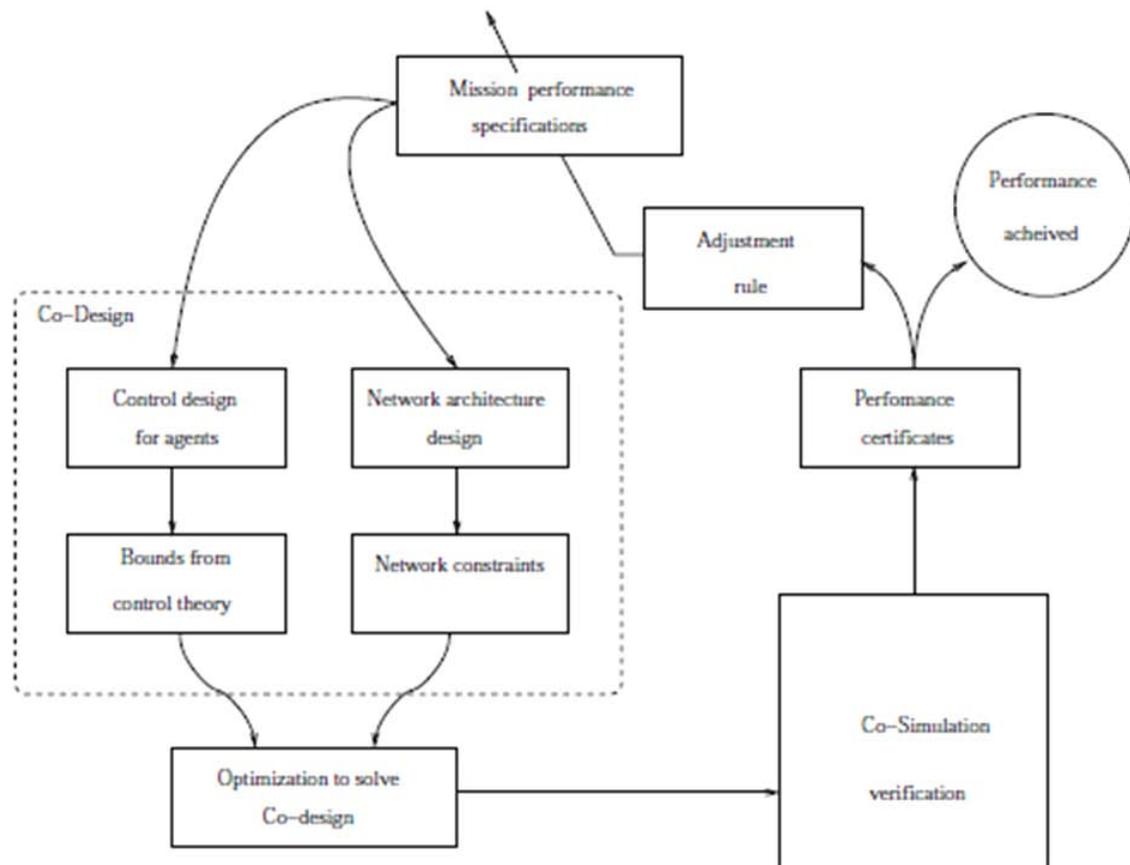


Figure 1.6. Boucle de conception globale d'un SCR (Branicky *et al.*, 2003)

La figure 1.6 montre clairement que la co-conception fait apparaître un étage supplémentaire permettant d'établir le gain de chaque approche (commande en/de réseau), et de sélectionner ainsi de manière appropriée l'action à accomplir. Actuellement, les travaux menés consistent à compléter les propositions de commande de réseau afin de lier dynamiquement la reconfiguration du réseau en fonction de paramètre de qualité de commande. Quelques travaux ont suivi ce sens et ont permis d'aller vers une co-conception.

Un premier axe que l'on peut dégager consiste à lier la commande de réseau (ou de ressources partagées) à la dégradation de la qualité de commande. Ainsi (Seto *et al.*, 1996) (Simon *et al.*, 2005) se sont basés sur la variation de la période d'échantillonnage dans le but d'optimiser le partage d'un processeur utilisé par différents contrôleurs. Dans (Simon *et al.*, 2005), la commande  $H_\infty$  a été utilisée et le gain du contrôleur est adapté en ligne en utilisant le nombre d'exécutions estimé des tâches de commande.

Pour réduire la charge du réseau, (Ramanathan, 1999) a proposé une approche qui se base sur la sélection de certains échantillons qui peuvent être écartés durant la période de surcharge. (Ramanathan, 1999) a proposé un schéma qui combine cette approche avec une commande optimale permettant ainsi de réduire la charge du réseau sans qu'il y ait une dégradation significative en termes de performance du système.

(Felicioni *et al.*, 2008) ont considéré  $N$  systèmes à commander avec pour chacun un contrôleur dédié (figure 1.7). (Felicioni *et al.*, 2008) ont considéré une implémentation centralisée de tous les contrôleurs (figure 1.7). L'approche proposée par (Felicioni *et al.*, 2008) se base sur les états des systèmes contrôlés pour déterminer la contrainte  $(m,k)$ -firm pour chaque tâche de contrôle dans le but de garantir à la fois un bon ordonnancement des tâches de commande et un haut niveau de performance pour la commande.

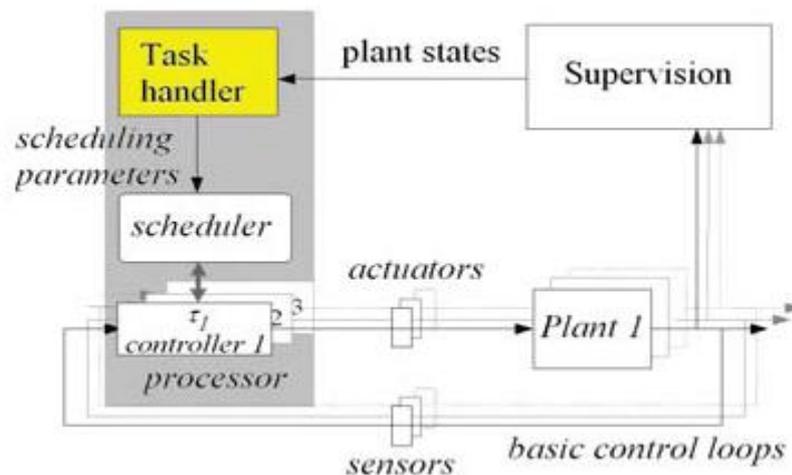


Figure 1.7. Architecture du système global (Felicioni *et al.*, 2008)

(Felicioni *et al.*, 2009) ont considéré l'approche qui se base sur la sélection de certains échantillons qui peuvent être écartés durant la période de surcharge (Ramanathan, 1999). (Felicioni *et al.*, 2009) se sont concentrés sur l'analyse de l'impact de séquences d'écartements de paquets sur la stabilité et la performance de la boucle de commande.

Toujours dans le cadre d'une prise en compte de la dégradation de la QdC induite par des actions de commande de réseau, on peut citer (Chen *et al.*, 2009) où les auteurs ont introduit une nouvelle approche de commande qui se base sur la commande de réseau. Dans leur cas, ils agissent sur la probabilité de distribution des délais de transmission, et sur la commande du système. Dans (Chen *et al.*, 2009), l'approche de commande est basée sur un système linéaire à saut Markovien (*Markovien jump linear system*) dépendant du délai.

#### 1.2.4. Conclusion

Comme on l'a vu dans ce paragraphe, il existe deux approches pour l'étude de la stabilité des SCR. De nombreux travaux ont d'ores et déjà été menés dans le cadre de la commande en réseau, alors que la commande de réseau est une problématique relativement jeune dans le contexte des SCR. C'est dans cette direction que cette thèse se place, avec comme double objectif, de définir une commande de réseau respectant les exigences de QdC tout en offrant le plus de bande passante pour le trafic de fond non associé à la commande du SCR.

### 1.3. Contexte de cette recherche et positionnement

#### 1.3.1. Contexte

Cette thèse a débuté dans le cadre du projet Européen STREP n° IST-004303 NeCST (Networked Control Systems Tolerant to fault)<sup>1</sup>. Ce projet a regroupé quatre partenaires universitaires (Université Henri Poincaré, Université de Duisburg-Essen, Université de Technologie d'Helsinki, Université de Hull) et trois partenaires industriels (PREDICT, SAE AUTOMATION, NESTE JACOBS OY).

L'objectif du projet NeCST a été d'explorer les possibilités de recherche dans le domaine des SCR et cela afin d'améliorer les performances des systèmes de contrôle en termes de diagnostic et de tolérance aux fautes. Concernant les systèmes embarqués, la contribution principale du projet NeCST dans ce domaine est la proposition de différents moyens pour améliorer la fiabilité du composant embarqué (Zhang et Ding, 2007) (Vatanski *et al.*, 2009) (Patton *et al.*, 2007) (Kettunen, 2006). La figure 1.8 illustre clairement les différents points d'intérêt du projet NeCST.

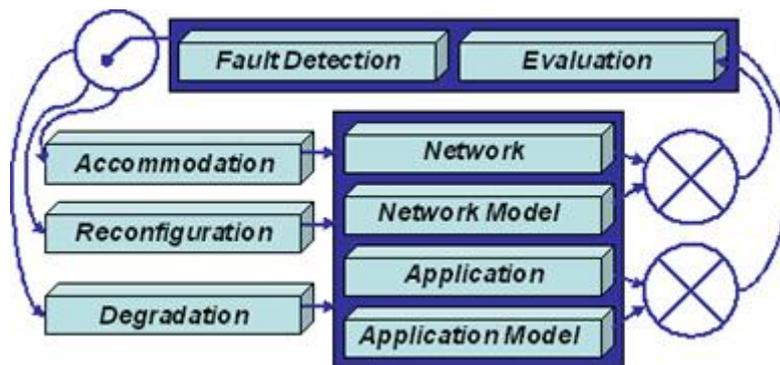


Figure 1.8. Principes de base du projet NeCST (<http://www.strep-necst.org>)

Durant l'année où j'ai participé au projet NeCST, je me suis intéressé à la modélisation du réseau « *Network Model* » et cela en utilisant la théorie du calcul réseau et aussi à la partie accommodation du réseau (Diouri *et al.*, 2007a).

Durant ma thèse, j'ai également participé au projet ANR Safe-NECS<sup>2</sup> (Safe-Networked Control Systems). Ce projet a regroupé 5 laboratoires français ayant des domaines d'expertises complémentaires pour l'étude des SCR (Automatique, Réseaux de communication, Diagnostic et Supervision, Systèmes temps-réel) :

- Centre de Recherche en Automatique de Nancy (CRAN)
- GIPSA-lab (Grenoble Images Parole Signal Automatique)

<sup>1</sup> <http://www.strep-necst.org/>

<sup>2</sup> <http://safe-necs.cran.uhp-nancy.fr/>

- LAAS (Laboratoire d'Analyse et d'Architecture des Systèmes)
- LORIA (Laboratoire Lorrain de Recherche en Informatique et ses Applications)
- INRIA Rhône-Alpes (Institut National de Recherche en Informatique et en Automatique)

L'objectif du projet Safe-NeCS a été de développer la co-conception qui prend en compte de manière coordonnée les différents paramètres (en termes de commande du système et en termes de performances du (ou des) réseau(x) de communications). L'intérêt de cette vision intégrée est d'assurer une meilleure conception (ou co-conception) du SCR et cela pour minimiser les ressources nécessaires pour satisfaire les contraintes temporelles du SCR. Cette minimisation est notamment importante pour les systèmes embarqués autonomes. Le projet s'est aussi intéressé au problème du diagnostic et plus précisément à la distribution de celui-ci entre différents « observateurs » et sur la façon de les faire communiquer en minimisant l'occurrence d'un diagnostic global erroné.

Le projet Safe-NeCS a été mené en se basant sur les co-simulations et l'implémentation (plateforme ORCCAD) (Berbra *et al.*, 2010) de la commande, de la supervision et de l'ordonnancement et cela en utilisant des modèles réalistes et détaillés pour décrire les tâches et l'ordonnancement des messages dans les CPU ainsi que dans les réseaux. Le défi du projet Safe-NeCS a été donc de maîtriser trois niveaux de régulation : la boucle fermée de commande, la boucle fermée de supervision et la boucle fermée d'ordonnancement. L'application qui a été choisie pour illustrer le travail du projet Safe-NeCS est un drone qui sera présenté dans la suite (chapitre 2).

Mon travail dans le cadre du projet Safe-NeCS a donné lieu à une publication qui a été faite en collaboration avec le GIPSA-lab dans la conférence MED'08 (Diouri *et al.*, 2008). Ce travail a concerné l'utilisation du réseau Ethernet commuté implémentant le protocole d'ordonnancement WRR et le but a été de montrer qu'une bonne configuration des poids de l'ordonnanceur WRR peut donner un bon comportement du système tout en maximisant la bande passante du trafic non contraint temporellement. La détermination de la configuration des poids a été possible grâce au résultat du calcul du délai maximal de traversée du réseau obtenu en utilisant le calcul réseau et qui a été établi dans (Diouri *et al.*, 2007a).

### 1.3.2. Positionnement

Dans ce chapitre, on a vu que les SCR peuvent être abordés avec plusieurs approches. Auparavant, il y a eu d'autres études au CRAN autour de cette problématique. Les travaux de (Michaut et Lepage, 2003) ont concerné l'adaptation de l'application à la qualité de service du réseau. Les travaux de (Georges, 2005) ont quant à eux concerné le calcul des délais de bout en bout dans le cas d'architectures Ethernet commutées supportant ou non la Classification de Service (Cds). Les études menées par (Brahimi *et al.*, 2008) ont mis l'accent sur l'importance des outils de co-simulation pour l'étude des SCR. Les travaux de (Boughnami *et al.*, 2009)

ont concerné l'IEEE 802.15.4/ZigBee et l'intérêt s'est porté sur deux mécanismes de gestion de la QoS utilisant la priorité probabiliste d'une part et déterministe d'autre part. (Habib *et al.*, 2009) proposent quant à eux une approche permettant de fournir des informations sur les performances temporelles d'un Système Distribué Commandé en Réseau (SDCR) sans-fil.

Dans cette thèse, l'intérêt se porte particulièrement sur l'approche commande de réseau. A l'échelle nationale, il y a plusieurs travaux autour de cette problématique des SCR. (Juanole et Mouney, 2007) ont proposé le concept de priorité hybride (hiérarchique) en utilisant le réseau CAN. (Koubâa, 2004) a proposé le (m,k)-WFQ qui assure que m sur k paquets consécutifs doivent respecter leurs contraintes de délais. A l'échelle internationale, certains auteurs ont basé leur étude sur le contrôle de la congestion (Floyd et Fall, 1999) (Mascolo, 1999) (Mascolo, 2006). (Floyd et Fall, 1999) ont justifié le besoin d'un contrôle de congestion de bout en bout et ils ont aussi montré l'importance de mécanismes dans le réseau pour détecter et restreindre les flux *best effort* durant les périodes de congestion. Concernant le réseau Ethernet, (Kweon *et al.*, 1999) ont proposé une technique de lissage qui permet de fournir des garanties de délai statistiques. Dans (Kweon *et al.*, 1999), chaque nœud utilise une méthode de lissage de trafic pour contrôler le trafic. (Lo Bello *et al.*, 2005) ont proposé une méthode de lissage de trafic floue permettant au réseau Ethernet de fournir une borne de délai statistique. (Lian *et al.*, 2004) ont proposé une commande de commutateur Ethernet dans le but de maintenir la QoS globale tout en garantissant les performances du système de commande. (Lian *et al.*, 2004) ont adopté la politique d'ordonnancement EDF (Liu et Layland, 1973) au niveau du commutateur Ethernet pour dispatcher l'ensemble des paquets qui viennent d'arriver dans le cas où l'ordonnancement est faisable. (Pedreiras *et al.*, 2003) se sont intéressés au trafic temps-réel dans le cas d'un réseau Ethernet commuté à priorités.

Dans cette thèse, l'étude concerne le réseau Ethernet commuté (norme IEEE 802.1D) (IEEE, 1998) et l'intérêt se porte sur l'ordonnancement WRR qui est déjà implémenté dans certains commutateurs Ethernet contrairement à EDF. Dans cette thèse, on ne considère pas un réseau Ethernet commuté à priorités contrairement à (Pedreiras *et al.*, 2003) car notre but est l'étude d'un protocole d'ordonnancement équitable qui permette certes de satisfaire les contraintes temporelles de l'application temps-réel mais aussi de garantir un service aux flux non contraints temporellement.

Les systèmes en réseaux sont des systèmes qui doivent respecter les contraintes temps-réel. Dans ce contexte, il est possible de distinguer deux catégories de SCR :

- ceux ayant des contraintes temps-réel strictes : pour cette catégorie, il faut garantir que le délai soit inférieur à un certain seuil (Liu et Layland, 1973). Si le délai est supérieur à ce seuil, le système pourrait être irréversiblement endommagé.
- et ceux ayant des contraintes temps-réel souples : pour cette catégorie, une distribution statistique de dépassements de délais est acceptée (Liu et Layland, 1973). Ainsi, celles-ci peuvent tolérer quelques dépassements du seuil fixé par l'application temps-

réel souple et cela selon par exemple la nature de la distribution temporelle des délais hors limite.

Dans ma thèse, je me focalise sur l'étude des réseaux Ethernet et plus particulièrement des réseaux Ethernet commutés car ils présentent plusieurs avantages comme l'élimination des collisions et l'implantation de mécanismes de classification de services. L'intérêt s'est porté particulièrement sur le protocole d'ordonnancement WRR car celui-ci permet de satisfaire les contraintes temporelles tout en évitant les effets de famine. De plus, une reconfiguration en ligne des paramètres WRR n'entraînent pas de rupture de communication.

Dans ma thèse, je me suis intéressé aux deux types d'applications : celles qui ont des contraintes temps-réel strictes (chapitre 3) et celles ayant des contraintes temps-réel souples (chapitre 4). Pour ce qui est des systèmes ayant des contraintes temps-réel strictes, la théorie du calcul réseau (Le Boudec et Thiran, 2004) est utilisée afin de garantir de manière déterministe les contraintes temps-réel tout en maximisant la bande passante du trafic de fond. Pour ce qui est des applications temps-réel souple, la théorie de la logique floue qui a été introduite par Zadeh en 1965 (Zadeh, 1965) est utilisée afin de caractériser les délais. Ensuite, cette identification sert à concevoir une commande floue permettant de satisfaire les contraintes temps-réel souples du SCR tout en maximisant les ressources du flux de fond. Dans cette thèse nous allons considérer un réseau dont le trafic de fond n'est pas précisément connu (seules des contraintes d'arrivées seront supposées). On se place ainsi dans le cadre d'un réseau partagé pour lequel la connaissance des délais est à priori inconnue.

Mais, avant d'étudier les applications SCR à contraintes temps-réel strictes et celles à contraintes temps-réel souples, nous allons dans le chapitre 2 expliquer les technologies de l'Ethernet commuté avec classification de services WRR et montrer comment celles-ci peuvent être utilisées pour commander les ressources de communication dans le contexte des SCR.

# Chapitre 2 :

## Les systèmes contrôlés en réseaux utilisant le réseau Ethernet

### 2.1. Ethernet

#### 2.1.1. Présentation de l'Ethernet

La partie 2.1.1 reprend partiellement la contribution de (Divoux *et al.*, 2008).

##### 2.1.1.1. Histoire

« Aloha ! » En 1969, c'est avec cette exclamation exotique que Norman Abramson est accueilli à Hawaï. Ce Professeur de la prestigieuse université de Stanford, passionné de surf tombe sous les charmes de l'archipel, et réussit l'année suivante à intégrer l'université d'Hawaï. Il travaille sur un projet de coordination des réservations hôtelières sur les différentes îles. Il appelle « Aloha » le réseau radio qu'il met en place. L'originalité réside dans la simplicité du mécanisme de partage de la fréquence unique utilisée par les abonnés au réseau : le protocole d'accès au médium se réduit en effet à :

- Vous voulez émettre ? Alors, émettez !
- Si vous entrez en collision avec une autre transmission, alors essayez « plus tard »...

On supposait une collision si aucun acquittement n'était reçu au bout d'un « certain » temps.

Abramson publie ses travaux qui retiennent l'attention de Robert Metcalfe, jeune ingénieur du Xerox PARC (Palo Alto Research Center). Cet ancien étudiant d'Harvard venait d'être embauché par Xerox, bien qu'il n'ait pu soutenir sa thèse sur la commutation de paquets jugée insuffisamment théorique. Le PARC est à l'époque richement doté de serveurs, imprimantes, et de nombreuses machines Alto (une des premières stations de travail personnelles développée par Xerox) dont l'interconnexion en réseau est confiée à Metcalfe. Il procède alors à une modélisation mathématique d'Aloha dont il identifie et quantifie les faiblesses. Il propose deux améliorations majeures au protocole d'origine :

- écouter avant de parler (vérifier que le médium n'est pas déjà occupé avant de transmettre) : méthode CSMA/CD (Carrier Sense Multiple Access with Collision Detection) ;
- un algorithme de calcul de la date de retransmission (plus formel que « plus tard ») : algorithme du BEB (Binary Exponential Backoff).

### 2.1.1.2. Principes

Ethernet est une spécification des couches 1 et 2 du modèle OSI (*Open Systems Interconnection*). En fait, c'est une spécification d'un profil du protocole CSMA/CD. Mais c'est aussi bien plus. En effet, CSMA/CD ne définit pas l'algorithme de résolution d'une collision alors qu'Ethernet définit l'algorithme BEB ainsi que la longueur des données, le médium physique (bus), ...

Ethernet implémente une version *1-persistent* de CSMA/CD puisqu'une station prête à émettre des données les transmet dès que le canal est libre avec une probabilité de 1. La procédure d'émission suit les étapes ci-dessous :

- écoute de la voie de transmission (détection de porteuse) ;
- si elle est occupée : attente de libération du canal ;
- si elle est libre : attente pour la neutralisation électrique du médium (trou inter trame ou Inter Frame Gap (IFG) = 9,6  $\mu$ s dans le protocole natif) puis émission. Pendant toute la transmission de la trame, la station vérifie qu'elle n'entre pas en collision. Ceci arrive si au moins deux stations émettent simultanément (synchronisées par exemple par la libération de la voie de transmission d'une autre station). Comme les stations ne détectent pas nécessairement la collision en même temps, elles transmettent encore quelques octets pour garantir la détection par toutes (séquence dite de « bourrage »). Ensuite, pour se désynchroniser, les stations impliquées dans la collision tirent un temps d'attente aléatoire (backoff) dans un intervalle qui croît exponentiellement avec le nombre de tentatives de retransmission (limité à 16). Celle qui a tiré le temps le plus court reprend la première la procédure d'émission : écoute puis transmission si le canal est libre...

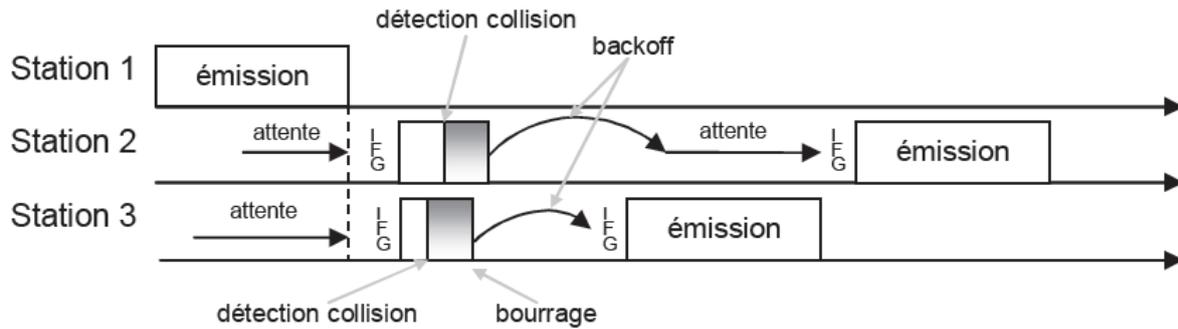


Figure 2.1. Résolution d'une collision (Divoux *et al.*, 2008)

### 2.1.1.3. Evolutions

#### Introduction

Outre sa simplicité et son faible coût, c'est également l'évolutivité d'Ethernet qui en a fait le succès que l'on connaît aujourd'hui. Si la méthode d'accès CSMA/CD est restée inchangée, il y a eu de nombreuses évolutions en termes de débit, de support de transmission et de topologie.

#### Débits

Les progrès de l'électronique ont bien sûr permis de dépasser rapidement les débits originaux. FastEthernet spécifié en 1995 (IEEE 802.3u), pousse à 100 Mbps. GigaEthernet depuis 1998 (IEEE 802.3z) permet d'atteindre 1000 Mbps. Le « Téra-bit Ethernet » est actuellement en phase de recherche. Le débit est sûrement l'un des critères majeurs qui a conduit les industriels à considérer Ethernet comme support de transmission dans des applications temps-réel. En effet, les réseaux de terrain classique proposent le plus souvent des débits modestes autour du Mbps.

#### Topologie

La méthode d'accès, très efficace lorsque le réseau est peu chargé, s'avère catastrophique (écroulement du réseau) lorsque le nombre de collisions augmente. On évalue à 40% de charge le seuil à ne pas dépasser. Ce qui arriva quand le nombre d'abonnés au réseau augmenta et que les services offerts les conduisirent à l'utiliser de plus en plus. La première idée fut de segmenter le bus en utilisant un « pont filtrant ». Le principe est simple : sur analyse des adresses, le pont laisse passer les trames lorsqu'émetteur et récepteur sont de part et d'autre du pont. Il bloque les trames s'ils sont du même côté, rendant donc possible dans ce cas deux transmissions simultanées sur chaque segment.

Parallèlement, on commença à construire des bâtiments pré câblés. Afin de garantir l'évolutivité de l'usage des prises (pour y brancher un téléphone ou une machine informatique), on passa sur paire téléphonique, mais on opta aussi pour une topologie en étoile autorisant un « rebrassage » aisé des deux réseaux. L'introduction de multi répéteurs ou *hubs* émulant le fonctionnement du bus fut une étape importante de l'évolution d'Ethernet. Le « mariage » du hub et du pont donnera très vite naissance au commutateur ou *switch*, équipement central de toute architecture Ethernet moderne.

Les commutateurs sont structurés relativement aux mécanismes de mémorisation, aux processus décisionnels de commutation et à l'architecture de commutation.

Les mécanismes de mémorisation s'appuient sur l'un des trois types de retransmission suivants :

- *Store & forward* : dans ce mode, le commutateur doit attendre la réception complète de la trame avant d'effectuer la commutation ;
- *Cut-through* : la commutation est activée dès la réception de l'adresse de destination, soit à la réception du 14<sup>ème</sup> octet ;
- *Fragment-free* : ici, le commutateur attend le 64<sup>ème</sup>, c'est-à-dire la réception complète d'une trame de longueur minimale.

L'avantage du mode *store & forward* est que le commutateur ne retransmet les trames qu'après avoir vérifié qu'elles ne sont pas erronées (détection d'erreurs par calcul de clé de contrôle). Néanmoins, ce mode ajoute un retard de traitement d'autant plus important que la trame est longue. Le mode *cut-through* réduit ce délai à sa valeur minimale (indépendante de la longueur de la trame). Sous l'hypothèse d'un faible taux d'apparition d'erreurs, le mode *cut-through* est donc préférable.

La mise en attente dans les buffers dépend aussi de leur emplacement qui peut se faire à l'entrée des ports, à la sortie ou encore entre les deux (mémoire partagée). Les deux dernières évitent les problèmes de blocage en tête de ligne propres à une mémorisation à l'entrée du commutateur. A noter aussi que la mémoire partagée offre plus de bande passante que dans un schéma de mémorisation par port. Finalement, la décision de commutation peut se faire de manière centralisée ou décentralisée et le transfert entre les ports d'entrées et de sorties repose sur des architectures bus, matricielles ou à mémoire partagée.

Ces différents mécanismes introduisent des retards dans l'acheminement des messages. C'est la notion de temps de latence inhérent au commutateur. Cela signifie que si le réseau est peu chargé, ce qui est souvent le cas dans des environnements industriels véhiculant des petits paquets de manière périodique, un réseau sur hub est plus performant en terme de délai qu'un réseau commuté. De plus, la diffusion d'une trame à plusieurs destinataires (multicast) souvent exploitée dans les applications de contrôle/commande est intrinsèque au bus et est

« moins naturelle » dans l'Ethernet commuté. Cependant, les hubs ne supportent pas la plupart des évolutions récentes autour d'Ethernet.

Les topologies autour de l'Ethernet commuté ont associé dans un premier temps des hubs et des commutateurs. Les hubs permettaient de raccorder les ordinateurs et les commutateurs jouaient le rôle de ponts entre les hubs. Puis, les commutateurs ont vu leurs prix baisser, et ont remplacé progressivement tous les hubs. On parle alors d'architectures micro-segmentées. L'implémentation systématique de liens full-duplex (IEEE 802.3x) sur des architectures micro-segmentées conduit à une topologie dite full-segmentée. La conséquence de la full-segmentation est d'éliminer complètement l'apparition de collisions. La nature du problème est transformée en gestion de la congestion des buffers des commutateurs.

### Synchronisation d'horloges

Dans les applications distribuées temps-réel, l'existence d'une référence temporelle est primordiale. Le protocole PTP (Precision Time Protocol) spécifié dans le standard IEEE 1588 permet de synchroniser avec une grande précision (de l'ordre de la  $\mu\text{s}$ ) les horloges de chaque nœud du réseau. Une fois les horloges synchronisées, il suffit d'estampiller chaque message avec sa date d'envoi. Une simple différence entre la date de réception et la date d'envoi portée par le message permet de connaître le délai d'acheminement d'une trame à travers le réseau. L'annexe D du standard 1588 présente une implémentation de PTP pour Ethernet. Le protocole PTP est utile pour fournir en temps réel aux applications de contrôle/commande, les informations sur les retards du réseau. Celles-ci pourront alors s'adapter dynamiquement aux variations des performances du réseau.

### **2.1.2. Méthodes permettant de rendre Ethernet adapté à interconnecter des systèmes temps-réel**

Au début, de nombreux travaux de recherche ont été entrepris pour résoudre le problème du non déterminisme du protocole d'accès CSMA/CD.

(Tobagi, 1982) a proposé le Prioritized-CSMA (P-CSMA) qui alloue des périodes d'émission aux stations en y ajoutant des priorités. Le P-CSMA permet de réduire les collisions car celles-ci n'ont lieu que dans le cas où deux stations de même priorité émettent en même temps. (Kopetz *et al.*, 1989) ont proposé d'utiliser le Time Division Multiple Access (TDMA) au dessus d'Ethernet de façon à fournir à tour de rôle pour chaque station un temps de parole exclusif sur le réseau. L'inconvénient du TDMA est que les intervalles de temps sont utilisés même si les stations n'ont rien à transmettre. (Le Lann et Riviere, 1993) ont proposé le CSMA-DCR (CSMA with Deterministic Collision Resolution) qui est un algorithme de résolution de collisions en attribuant un ordre d'accès aux différents transmetteurs après l'occurrence d'une collision. L'inconvénient du CSMA-DCR est qu'il est relativement lourd à mettre en œuvre et incompatible avec le matériel standard que l'on trouve sur étagère.

Parallèlement à ces travaux de recherche, des solutions pragmatiques ont été proposées par les différents constructeurs d'équipements Ethernet industriels pour s'affranchir du caractère aléatoire d'accès au médium CSMA/CD. Ces solutions ont été récemment comparées par (Robert *et al.*, 2010). On pourra citer notamment :

- EtherCAT (Ethernet for Control Automation Technology) qui est développé par Beckhoff (type 12 dans le standard IEC 61158) (IEC, 2006a) (IEC, 2006b). Dans l'EtherCAT, un protocole Maître/esclaves est ajouté au dessus d'Ethernet. Dans l'EtherCAT, une latence constante (temps d'écriture et de lecture) est introduite par les équipements. Dans l'EtherCAT, un anneau logique est défini entre les différents esclaves et lorsque la trame atteint le dernier esclave de l'anneau, celle-ci est renvoyée au maître. (Robert *et al.*, 2010) ont souligné que l'EtherCAT n'utilise pas des cartes Ethernet standards mais plutôt un FPGA (Field-Programmable Gate Circuit) et cela pour introduire des facultés de lecture/écriture à la volée. EtherCAT est donc une solution propriétaire non compatible avec d'autres.
- Profinet IRT (Isochronous Real-Time) qui est une des versions du Profinet lequel est développé par Siemens (type 10 dans le standard IEC 61158) (IEC, 2006a) (IEC, 2006b). Le protocole Profinet se base sur la division temporelle du cycle de communication en deux modes : asynchrone (pour des trafics non contraints temporellement) et isochrones (pour des trafics temps-réel). (Robert *et al.*, 2010) ont noté également le fait que dans Profinet, les cartes Ethernet standard ne peuvent pas être utilisées.

D'autres réseaux Ethernet industriels sont commercialisés, mais n'introduisent pas d'ajout protocolaire au dessus de CSMA/CD pour garantir des performances de délais. C'est à l'utilisateur de prévoir des mécanismes au niveau applicatif (type TDMA) pour maîtriser les délais de réseau. On peut citer le réseau Modbus/TCP (type 15 dans le standard IEC 61158) (IEC, 2006a) (IEC, 2006b) où le choix de TCP n'apparaît pas vraiment judicieux pour le contrôle/commande de systèmes industriels ou encore Ethernet/IP (type 2 dans le standard IEC 61158) (IEC, 2006a) (IEC, 2006b). Ici IP signifie Industrial Protocol. Ethernet/IP est développé par Rockwell Automation. Contrairement à EtherCAT et Profinet IRT, Ethernet/IP s'appuie sur des composants sur étagères (COTS : Components Off The Shelves). Ethernet/IP est de type commuté et les résultats des travaux de notre thèse pourraient très bien s'appliquer sur ce type de réseau.

Le problème de toutes ces méthodes pour rendre Ethernet déterministe est qu'elles tendent à s'éloigner de la norme de base d'Ethernet et qu'elles ne sont pas compatibles entre elles. Notre objectif est de travailler uniquement sur de l'Ethernet largement diffusé comme l'Ethernet commuté pour rendre pérennes nos propositions.

### 2.1.3. Classification de service

Un des avantages des commutateurs Ethernet est qu'ils donnent la possibilité d'implémenter la Classification de Service (CdS) permettant d'ordonner les messages en sortie des ports des commutateurs et cela selon leur priorité. Les algorithmes d'ordonnement habituellement implémentés dans les commutateurs Ethernet sont la priorité stricte et le Weighted Round Robin (WRR) (Demers *et al.*, 1989). Le choix de la politique d'ordonnement adéquat est fondamental pour l'optimisation des ressources du réseau et pour la garantie des contraintes temps-réel.

Comme le montre la figure 2.2, une classe de service est un groupe de flux pour lequel le commutateur fournit un niveau particulier de performance. Chaque classe de service est associée à une file en sortie de chaque port du commutateur. Les ports de sorties sont gérés par une politique d'ordonnement.

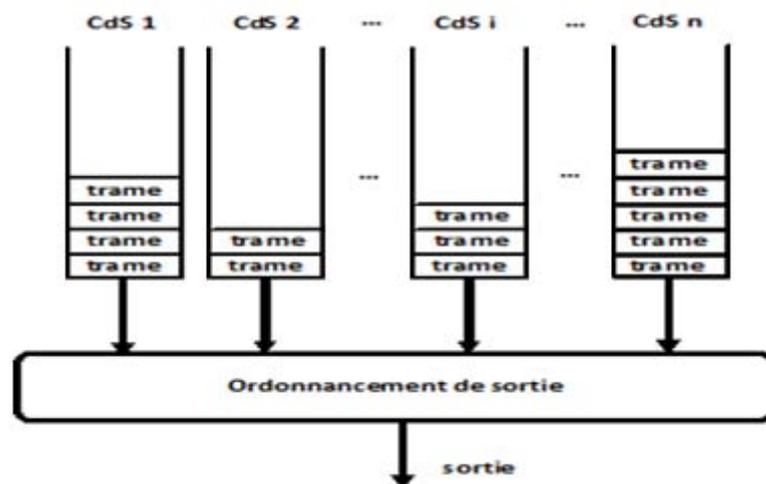


Figure 2.2. Files d'attente pour chaque classe de service au niveau d'un port de sortie d'un commutateur

L'utilisation des réseaux Ethernet commutés est très intéressante dans le contexte des systèmes ayant des contraintes temps-réel strictes. La politique d'ordonnement qui est retenue dans la thèse est le Weighted Round Robin (WRR) (Demers *et al.*, 1989) (Parekh, 1992). Le WRR correspond à une implémentation pratique qui approxime le Generalized Processor Sharing (GPS) (Parekh, 1992). Certains commutateurs supportent le WRR. Par exemple, le commutateur Cisco Catalyst 2950 permet de faire de la classification de service et permet l'implémentation du protocole WRR. Le commutateur Cisco Catalyst 2950 permet de distinguer entre quatre classes de service et cela pour chaque port de sortie (la concurrence a lieu au niveau des ports de sortie du commutateur). Sur le commutateur Cisco Catalyst 2950, il est aussi possible de configurer la priorité stricte.

## 2.2. Variantes des ordonnancements équitables basés sur GPS

Pour comprendre l'origine du WRR, il est important de savoir qu'à la base, c'est une approximation pratique du Generalized Processor Sharing (GPS) (Parekh, 1992). Le GPS sert les files non vides de l'ordonnanceur tout en différenciant les services offerts à chaque file. D'un autre côté, il y a le Processor Sharing (PS) qui sert toutes les files non vides avec le même débit ou le Round Robin (RR) qui traite un paquet par cycle pour chaque file. Le PS et le GPS utilise un modèle idéal qui ne peut pas être implémenté dans la pratique car il suppose que les paquets sont infiniment divisibles. Ainsi, il y a plusieurs approximations qui ont été proposées ; le Fair Queuing (FQ) (Demers *et al.*, 1989) donne une approximation par paquet du PS. Le Packet GPS (PGPS) puis le Weighted Fair Queuing (WFQ) proposé par (Parekh, 1992) donne une approximation du GPS. Dans un système WFQ, le paquet choisi pour la transmission est celui qui serait transmis en premier dans le système GPS correspondant. WFQ donne une approximation du GPS en termes de délais et de bits servis pour chaque session.

Le Worst-case Fair weighted Fair Queuing (WF<sup>2</sup>Q) (Bennett et Zhang, 1996) est aussi un algorithme qui donne une approximation du GPS en termes de délai et de bits servis pour chaque session. Dans un système WF<sup>2</sup>Q, l'ordonnanceur considère seulement l'ensemble des trames qui ont commencé (si possible fini) à être transmises dans le système GPS correspondant. A partir de cet ensemble, l'ordonnanceur choisit la trame qui sera la première transmise dans le système GPS correspondant.

Dans l'exemple suivant (figure 2.3), (Bennett et Zhang, 1996) ont considéré que les paquets ont la même taille. Ils ont montré à travers cet exemple qu'un paquet peut être transmis beaucoup plus tôt dans un système WFQ si on le compare avec le système GPS correspondant. En effet, on voit bien d'une part que le paquet  $p_1^{10}$  est transmis à  $t=20$  dans le système GPS (figure 2.3.b) et on voit bien d'autre part que ce même paquet est transmis à  $t=10$  dans le système WFQ (correspondant au GPS) (figure 2.3.c).

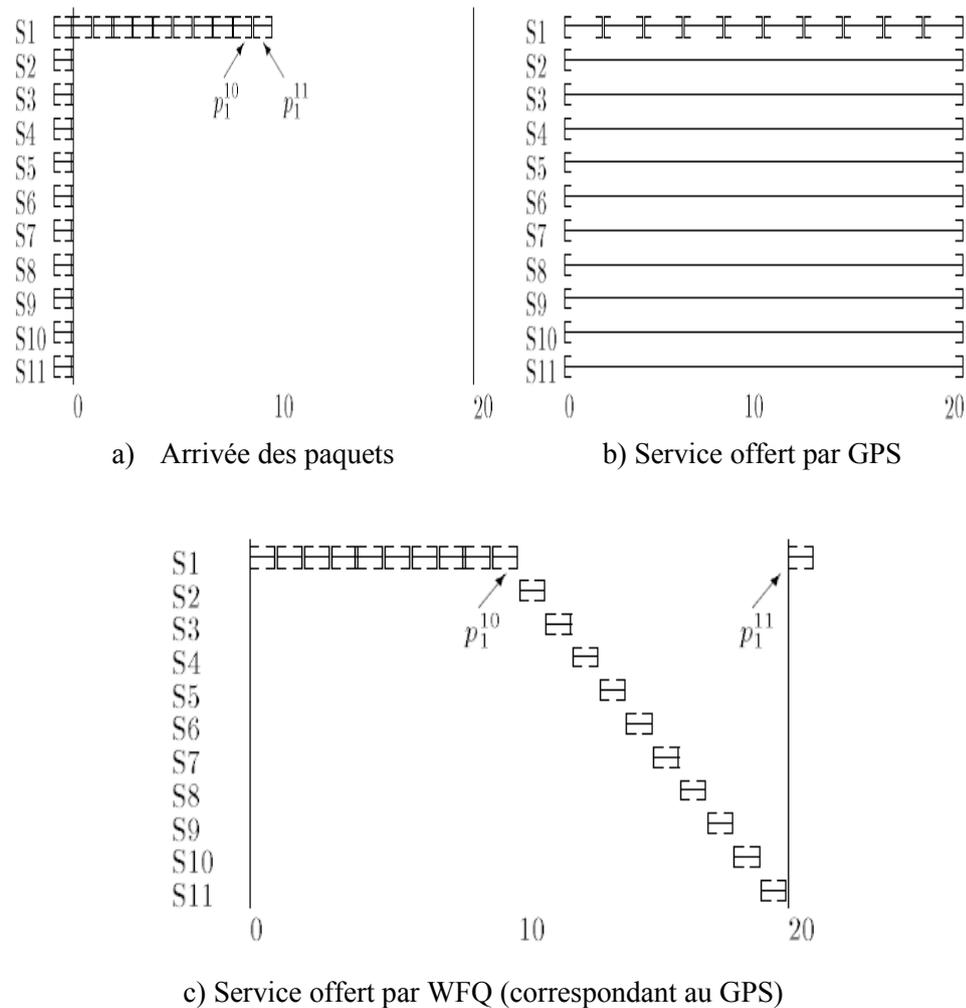


Figure 2.3. Les services offerts par GPS et par le WFQ (Bennett et Zhang, 1996)

Les solutions précédentes apportent intrinsèquement une augmentation de la complexité d'implantation. Actuellement, ces solutions ne sont généralement pas implantées. Dans la suite, nous allons présenter quelques implantations pratiques qui permettent d'approximer certaines de ces solutions.

Dans le Round Robin (RR), les paquets sont placés dans des files correspondant à leur niveau de priorité. L'ordonnanceur traite les files de manière cyclique (suivant un ordre défini par les priorités des files) en envoyant un paquet pour chaque file non vide. Ce mécanisme assure une égalité mais il ne tient pas compte de la flexibilité. Plus encore, l'équité pourrait être remise en cause par les longueurs variables des paquets. Le service devient alors variable sans moyen de le maîtriser.

Dans le but de contrôler l'équité par rapport à une simple politique RR, le WRR (Demers *et al.*, 1989) associe un poids  $w_i$  à chaque flux  $i$ . L'ordonnanceur WRR sert un flux  $i$  avec le taux  $\frac{w_i}{\sum_j w_j}$  avant de traiter la file suivante. En comparant à PGPS, les délais peuvent être plus

importants, particulièrement pour un système très chargé dans lequel une trame vient de rater sa transmission. Dans ce cas, elle doit attendre le cycle suivant.

Comme le montre la figure 2.4, le WRR affecte une priorité pour chaque file  $i$ . Il sert les files de manière cyclique, en commençant par la file ayant la plus haute priorité jusqu'à celle qui a la plus faible priorité. Le nombre de trames d'une file  $i$  qui est transmis est borné par  $w_i$ . Lorsqu'une file est vide, le protocole d'ordonnancement peut passer immédiatement à la prochaine file.

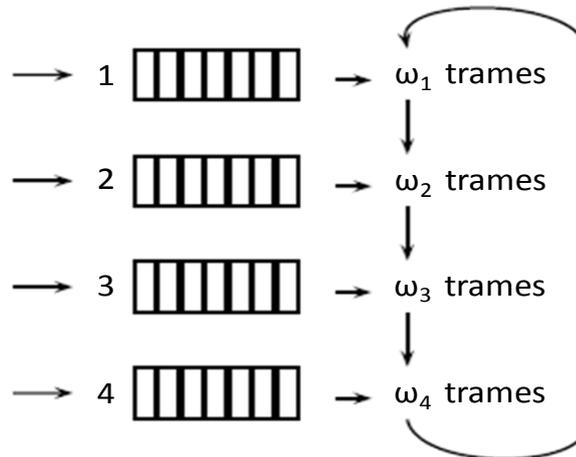


Figure 2.4. Le comportement du WRR (Diouri *et al.*, 2007a)

### Conclusion

Dans cette thèse, il s'agit de garantir les contraintes temporelles des applications temps-réel tout en évitant les situations de famine. Pour permettre une certaine équité dans l'attribution de la bande passante du réseau, nous avons opté, pour les raisons précédemment citées dans ce chapitre, pour le protocole d'ordonnancement WRR. Aussi, comme il sera montré dans la suite, des architectures Ethernet commutées implémentant le protocole d'ordonnancement WRR peuvent très bien être utilisées dans le contexte d'applications temps-réel (strict et souple) tout en garantissant une bande passante aux flux non contraints temporellement. En effet, l'ajustement des poids de l'ordonnanceur WRR permet de favoriser les échanges de trafics fortement contraints temporellement par rapport aux trafics moins prioritaires tout en assurant un certain niveau de service pour ce dernier. L'objectif de cette thèse est d'être capable en fonction du réglage de ces poids d'estimer les délais de bout en bout de chaque trafic selon son niveau de priorité.

De façon empirique, nous allons montrer par simulation et de façon expérimentale l'intérêt de WRR pour les SCR. Ce travail présenté dans la section suivante, correspond en réalité à une pré-étude menée avant la partie modélisation mathématique des délais sur Ethernet commuté de façon à bien comprendre le comportement et les intérêts de WRR dans un cadre

d'applications temps-réel. Ces études ont été réalisées sur plusieurs simulateurs et ont nécessité des développements personnalisés pour pouvoir simuler des SCR sous Ethernet commuté implantant le WRR. Pour illustrer cette analyse à travers un cas réel, nous avons aussi élaboré une plateforme expérimentale basée sur des commutateurs Cisco.

### **2.3. Exemples illustrant l'impact des délais sur le comportement d'un SCR utilisant le réseau Ethernet**

Dans ce paragraphe, trois exemples sont présentés.

Le premier exemple, introduit une plateforme expérimentale qui comporte un contrôleur, un process et deux stations qui génèrent du trafic de fond (Diouri *et al.*, 2007b). Le but est de montrer l'existence de relations entre commande de réseau et commande du système.

Le deuxième exemple, illustre l'impact du délai sur un drone qui a été développé dans le cadre du projet ANR Safe-NeCS. Les résultats qui sont présentés dans cette partie ont été obtenus en collaboration avec le GIPSA-LAB et ont été publiés dans la conférence MED'08 (Diouri *et al.*, 2008). Dans cet exemple, les résultats du réseau Ethernet utilisant une architecture commutée ont été comparés avec ceux obtenus par le réseau CAN et cela sur la commande du drone. Le comportement du drone a été simulé en utilisant le logiciel Matlab/Simulink et les différents réseaux ont été simulés à l'aide du logiciel TrueTime (Andersson *et al.*, 2005).

Le troisième exemple, représente un SCR utilisant une architecture Ethernet commutée. Les simulations pour cet exemple sont effectuées en utilisant le logiciel de simulation OPNET (OPNET, 2005). Cependant, le logiciel OPNET (OPNET, 2005) ne permet pas de simuler le comportement du système de contrôle/commande. Ainsi, il a fallu utiliser un nouveau module développé au sein de notre équipe de recherche au CRAN pour modéliser un SCR.

#### **2.3.1. Evaluation expérimentale (Diouri *et al.*, 2007b)**

Dans (Diouri *et al.*, 2007b), l'architecture qui a été étudiée est celle qui est représentée dans la figure 2.5 : celle-ci se compose d'un contrôleur, d'un process et de deux stations qui génèrent des flux non contraints temporellement lesquels rentrent en concurrence avec le trafic temps-réel (entre le contrôleur et le process). Tous les ports du commutateur ont été configurés à 10 Mb/s en mode *full-duplex*. Dans (Diouri *et al.*, 2007b), l'analyse d'une adaptation en ligne des poids du WRR a été réalisée et cela afin de garantir un bon comportement du système tout en maximisant la bande passante offerte au trafic non contraint temporellement.

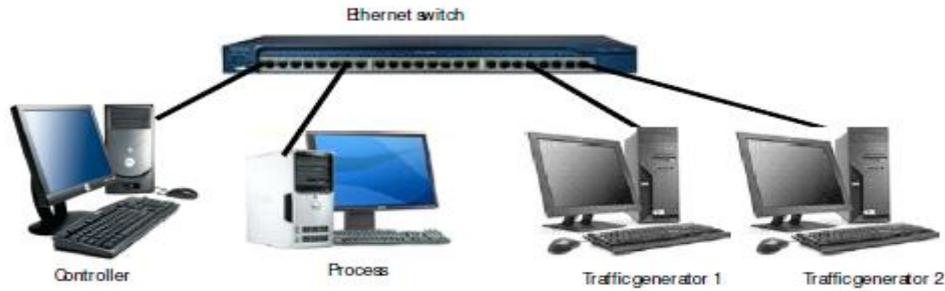


Figure 2.5. Plateforme expérimentale (Diouri *et al.*, 2007b)

Dans (Diouri *et al.*, 2007b), le process considéré a pour équation :  $P(s) = \frac{2}{(s+5)(s+0.2)}$  et le contrôleur considéré a pour équation :  $C(s) = \frac{0.5508s+0.4529}{s}$ .

Les équations du process et du contrôleur ont été programmées en langage C sur leur machine respective. Le contrôleur est échantillonné avec une période de 2 ms.

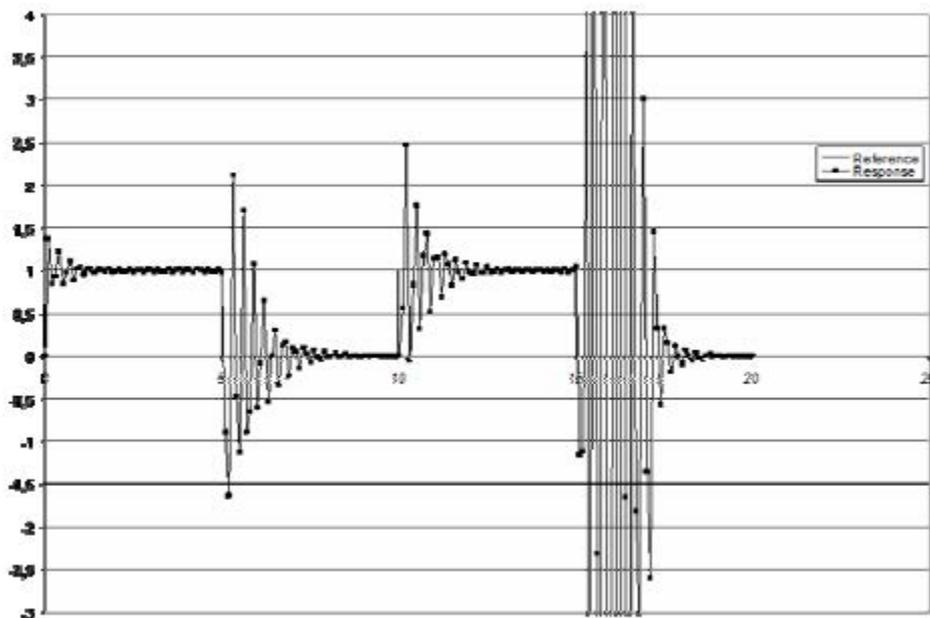


Figure 2.6. Comportement du système (Diouri *et al.*, 2007b)

Ici, le flux de fond correspond à une charge de 8 Mb/s qui est envoyée depuis les deux générateurs de trafic (Traffic Generator 1 et Traffic Generator 2) (figure 2.5) vers le contrôleur et le process.

La figure 2.6 montre le comportement du système en présence de la charge dans le cas où il n'y a pas de protocole d'ordonnancement. On voit bien d'après la figure 2.6 que la présence

de la charge peut introduire des oscillations supplémentaires qui pourront remettre en cause la stabilité de la commande du process.

Ainsi, (Diouri *et al.*, 2007b) ont proposé différentes stratégies de reconfiguration dynamique de poids comme la reconfiguration dynamique basée sur la valeur du délai du trafic temps-réel, ou encore la reconfiguration dynamique basée sur l'erreur du système, etc.

Nous montrons dans la suite l'étude sur l'adaptation du réseau en fonction de l'évolution de l'erreur du système. Cette erreur évolue car le trafic de fond engendre des retards variables entre le process et le contrôleur. Sur la figure 2.8, initialement les poids du WRR affectés au trafic temps-réel (ligne pleine) et au trafic de fond (ligne en pointillés) sont de même ordre. Puis le poids affecté au trafic temps-réel est continuellement décrémente (de manière globale) tandis que celui du trafic de fond est incrémenté (de manière globale). Le but est d'offrir le plus de bande passante possible au trafic de fond. Lorsque l'erreur dépasse 5%, on réalise l'opération inverse pour donner plus de bande passante au trafic temps-réel. Nous pouvons noter que cette phase peut être plus ou moins marquée selon la charge du trafic de fond. La figure 2.7 montre que cette adaptation en ligne du réseau maintient le bon fonctionnement de l'application.

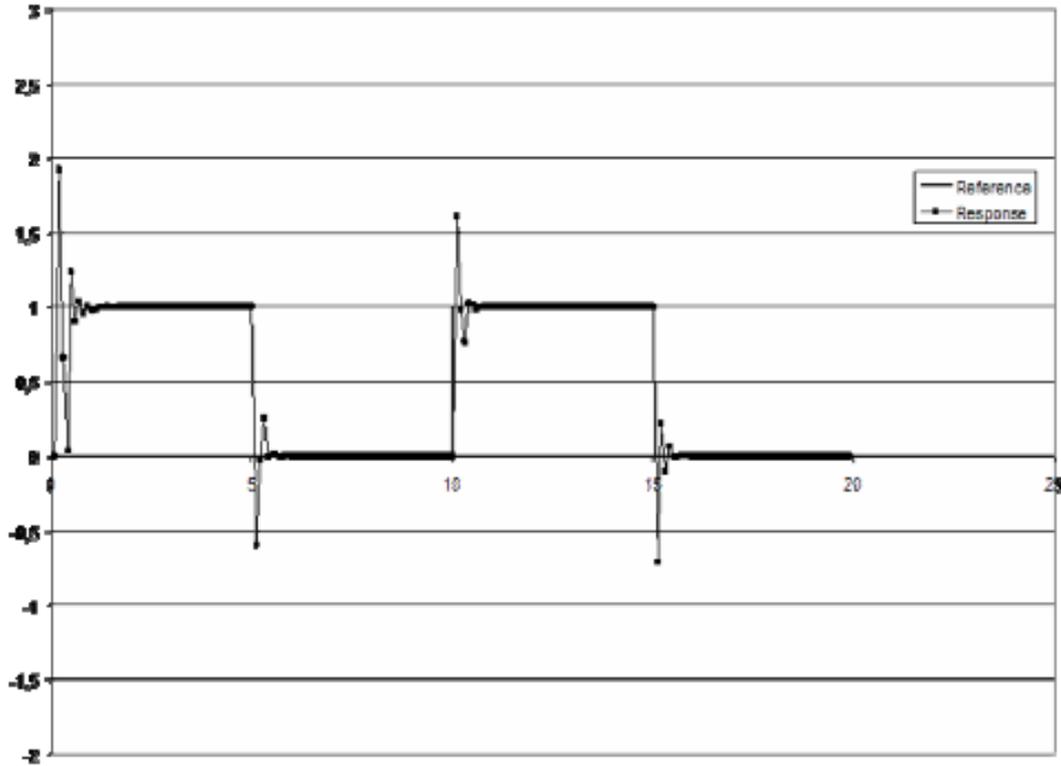


Figure 2.7. Comportement du système (cas de la compensation de l'erreur) (Diouri *et al.*, 2007b)

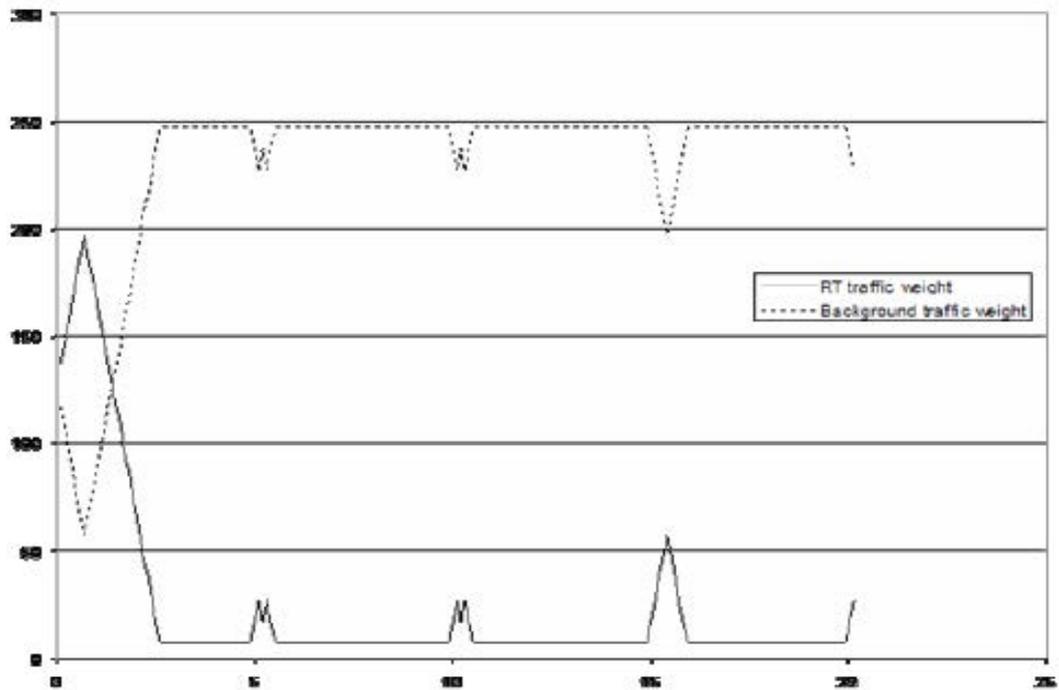


Figure 2.8. Réglage des poids (cas de la compensation de l'erreur) (Diouri *et al.*, 2007b)

Cette expérience est intéressante car elle décrit bien la relation qui existe entre commande de réseau en ligne et commande du système. Cependant, elle ne donne pas les clés pour le réglage optimal des poids du WRR et n'étudie pas les problèmes liés à la stabilité de la commande de réseau puisque les poids du WRR évoluent continuellement.

## 2.3.2. Apport d'un ordonnancement WRR dans le cas de la commande d'un drone

### 2.3.2.1. Présentation du drone

Le drone est un véhicule aérien contrôlé par les vitesses de rotation de quatre pales, pour cela le drone dispose de quatre moteurs électriques (figure 2.9). Sur ce drone, il est possible d'embarquer une caméra afin d'envoyer des images *via* le réseau sans fil sur une station terrestre.

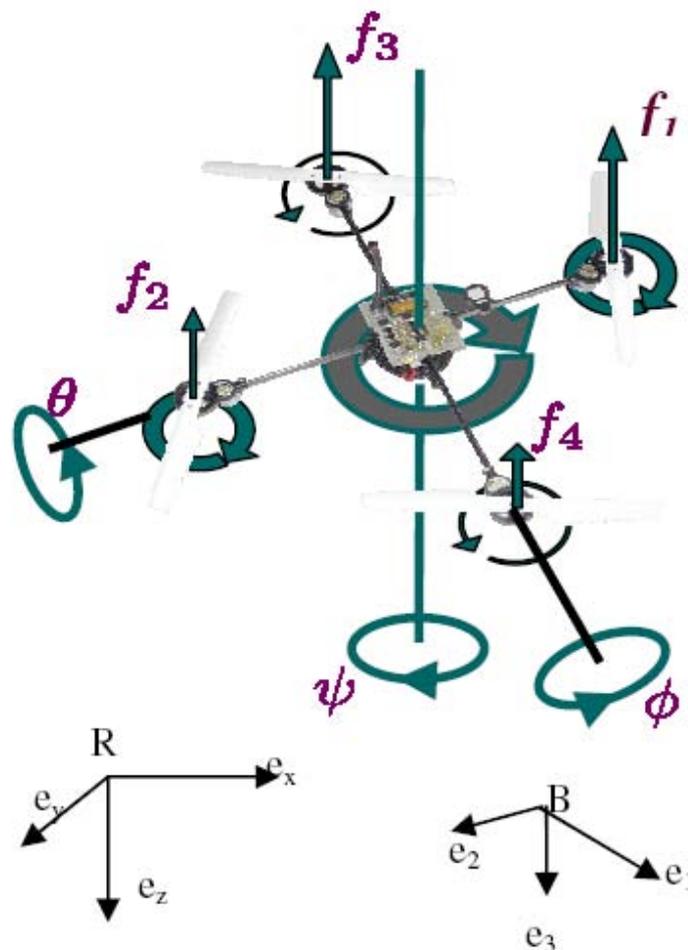


Figure 2.9. Benchmark du drone (Diouri *et al.*, 2008)

Le développement des équations a été fait en utilisant deux repères : le repère inertiel  $R(e_x, e_y, e_z)$  et le repère propre au drone  $B(e_1, e_2, e_3)$  ayant pour origine le centre de gravité de celui-ci.

L'attitude du drone est décrite par les 3 angles de rotation roulis ( $\varphi$ ), tangage ( $\theta$ ) et lacet ( $\psi$ ) qui sont représentés dans le repère B (figure 2.9).

Le drone est contrôlé en faisant varier indépendamment les vitesses de rotation  $\omega_{mi}$  de chaque moteur électrique. La description complète du modèle mécanique a été donnée dans (Tanwani *et al.*, 2007). La figure 2.10 représente l'architecture du drone après avoir embarqué un réseau. Les réseaux qui ont été étudiés dans (Diouri *et al.*, 2008) sont CAN et Ethernet commuté. Le but de l'étude suivante va alors être de montrer qu'un ordonnancement WRR peut permettre à Ethernet de remplacer un réseau de terrain déterministe et conventionnel comme le réseau CAN.

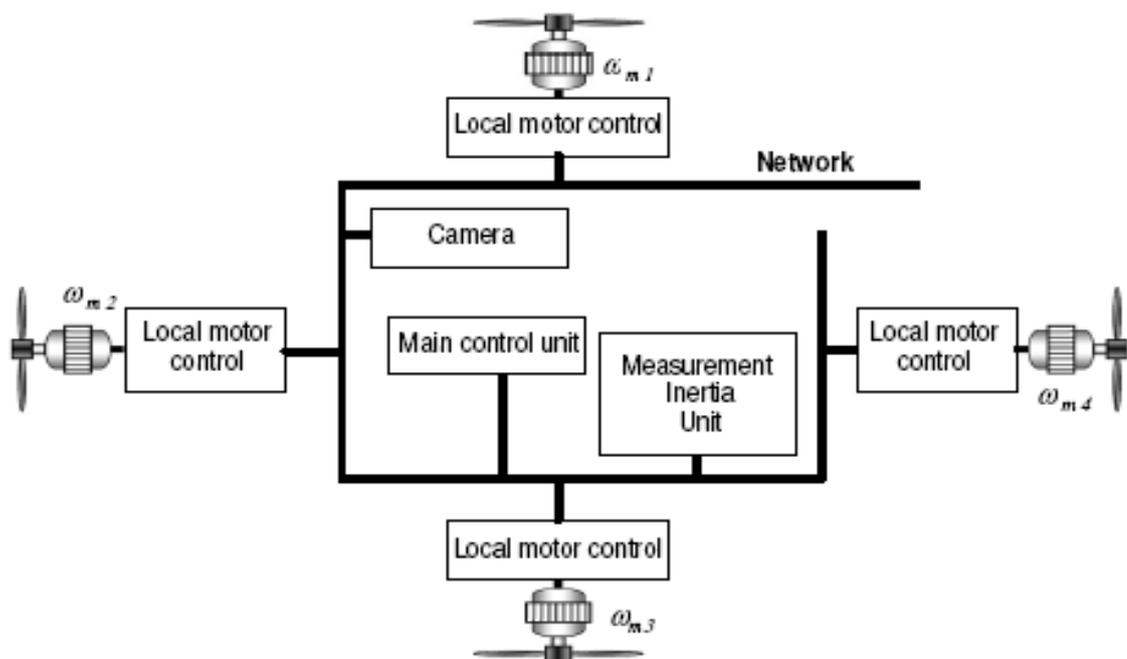


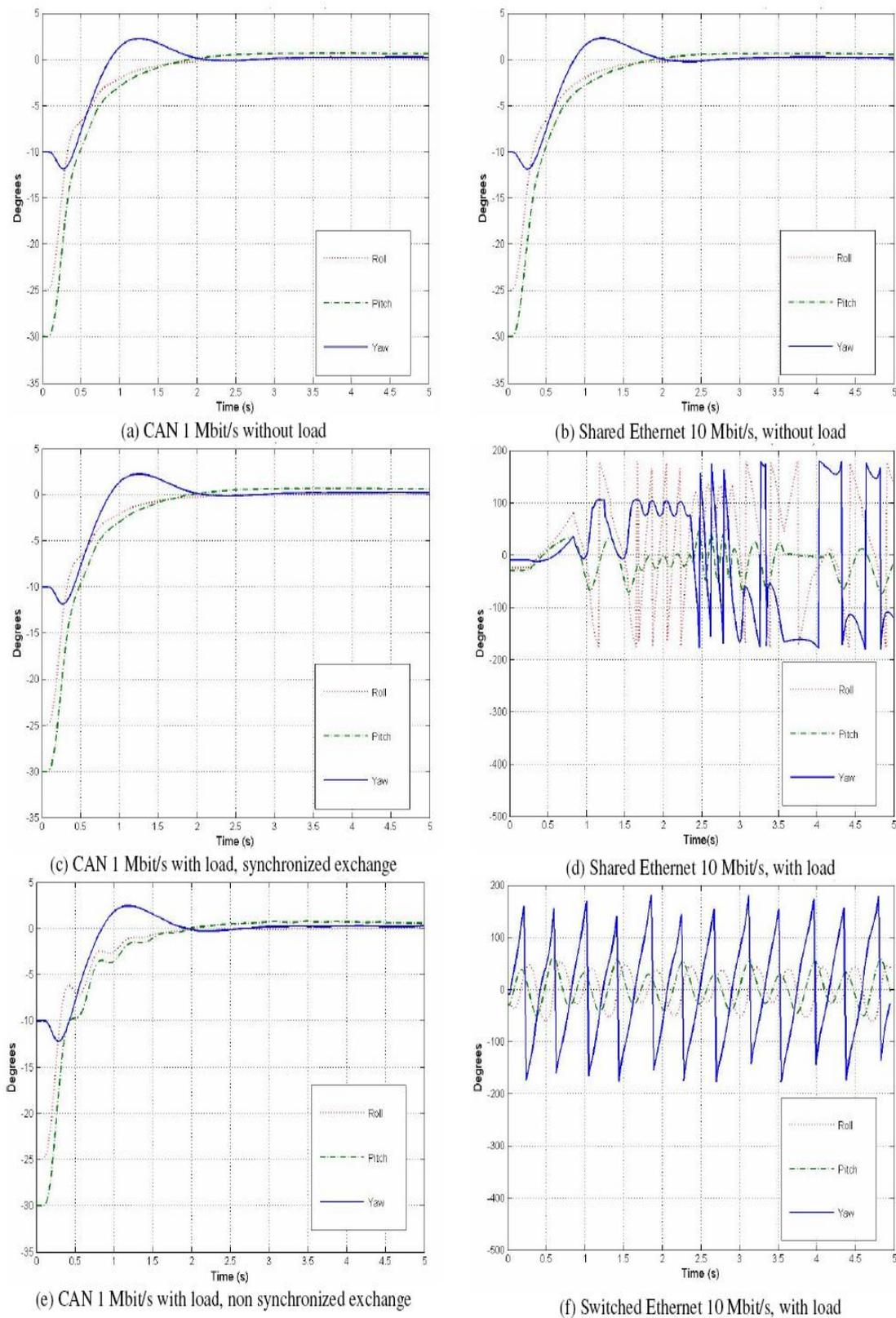
Figure 2.10. Architecture du drone incluant le réseau embarqué (Diouri *et al.*, 2008)

Les capteurs envoient 9 mesures de manière périodique avec une période d'échantillonnage ( $T_s = 10$  ms). Par contre, l'envoi des commandes est événementiel et le contrôleur attend de recevoir toutes les mesures avant de procéder au calcul de la commande (Berbra *et al.*, 2007). Le contrôleur envoie quatre signaux de commande (une pour chaque moteur local) aussitôt qu'il achève le calcul de celle-ci. Le débit du réseau CAN est de 1 Mb/s et celui du réseau Ethernet commuté est de 10 Mb/s. Pour le réseau CAN, la taille des paquets qui rentre dans la commande du drone est de 59 bits tandis que pour le réseau Ethernet la taille des paquets du même type est de 72 octets (taille minimale d'une trame Ethernet). Ici, le réseau Ethernet commuté s'apparente à une architecture centralisée où tous les équipements sont interconnectés sur un même commutateur.

Dans les deux cas, le réseau devra supporter le transport d'un trafic assimilable à une vidéo qui transmettrait 256 niveaux de gris à l'unité de commande principale « Main control unit ». Dans le cas du réseau CAN, le flux relatif à la transmission vidéo est le moins prioritaire. Tandis que pour le réseau Ethernet commuté il existe deux classes de services : la première concerne tous les paquets de commande et de mesure et la seconde concerne tous les paquets du trafic de fond.

### **2.3.2.2. Impact des délais sur la commande du drone**

Dans les expérimentations qui ont été menées, la consigne a été fixée à (0, 0, 0) pour les angles ( $\varphi$ ,  $\theta$ ,  $\psi$ ) et l'état initial de ces derniers est (-25, -30, -10). Les figures 2.11.a, 2.11.c et 2.11.e montrent le comportement du réseau CAN. Dans le cas d'un trafic non-synchronisé (quand le flux vidéo est envoyé juste avant les autres flux), on remarque qu'il y a une dégradation au niveau de la phase transitoire. Ceci peut s'expliquer par le fait qu'un paquet du flux de commande doit attendre que le paquet du flux vidéo ait été transmis avant d'être transmis à son tour. D'un autre côté, les figures 2.11.b, 2.11.d et 2.11.f montrent que dans le cas de l'Ethernet standard et dans le cas de l'Ethernet commuté sans classification de service, l'introduction du flux vidéo dégrade dramatiquement la commande du système. Ces résultats montrent clairement que le débit n'apporte pas toujours une réponse satisfaisante pour contrôler un SCR puisque le réseau CAN à 1Mb/s donne de meilleurs résultats qu'un réseau Ethernet à 10Mb/s.

Figure 2.11. Impact sur la sortie du système (Diouri *et al.*, 2008)

Ainsi, dans le cas de l'Ethernet commuté, il a fallu implémenter un protocole d'ordonnancement équitable adéquat et cela afin de stabiliser le système et de maximiser la bande passante allouée au flux vidéo (chapitre 3).

Comme signalé, le comportement du drone a été simulé en utilisant le logiciel Matlab/Simulink et les différents réseaux ont été simulés à l'aide du logiciel TrueTime (Andersson et *al.*, 2005). Cependant, dans la boîte à outils TrueTime, il est impossible d'évaluer les performances d'un réseau Ethernet commuté implémentant la CdS et le protocole d'ordonnancement WRR. De ce fait, il a fallu développer un module WRR sur le logiciel TrueTime.

### **2.3.2.3. Extension du bloc réseau TrueTime pour la simulation d'un ordonnancement WRR**

Dans un premier temps, le comportement du WRR sur le réseau Ethernet commuté est apporté à TrueTime. Les parties entourées sur la figure 2.12 montrent les paramètres du bloc réseau de TrueTime que nous avons ajoutés ainsi que le paramétrage du réseau Ethernet commuté implémentant le protocole d'ordonnancement WRR « Switched Ethernet (WRR) ». Le champ « Switch initial weights priorities » permet d'affecter les poids initiaux pour chaque classe de trafic en commençant par celle qui a la plus haute priorité. Dans la figure 2.12, on distingue deux classes de trafic : celle ayant la plus haute priorité peut émettre une trame par cycle et la deuxième peut émettre 255 trames par cycle.

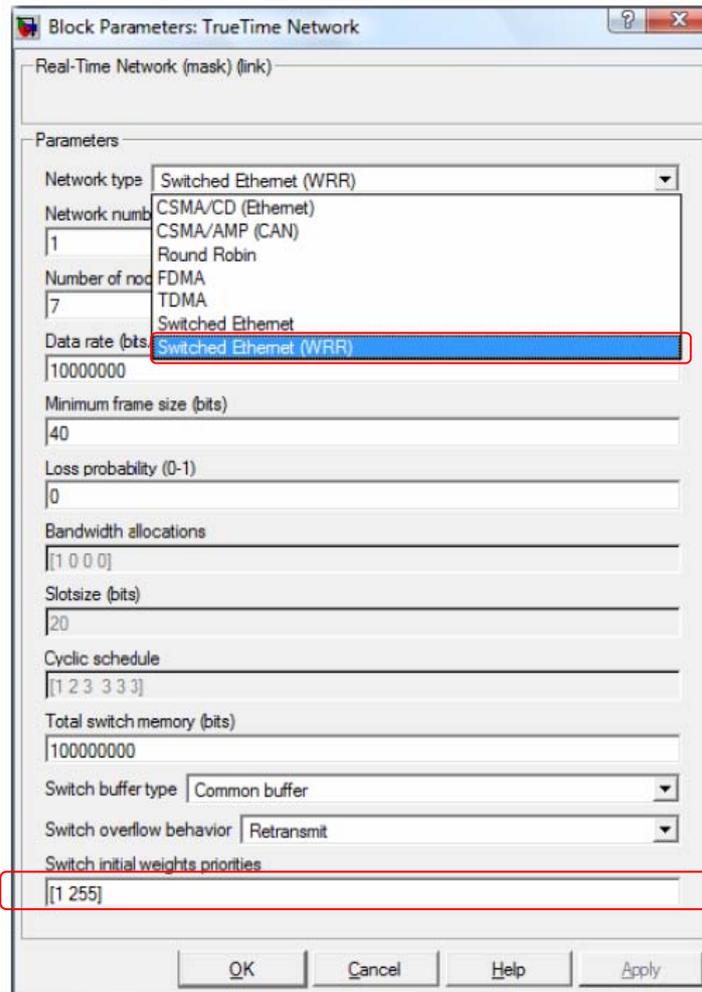


Figure 2.12. Les paramètres du bloc réseau étendu de TrueTime (Diouri *et al.*, 2008)

#### 2.3.2.4. Apport du WRR dans le cas du drone (Diouri *et al.*, 2008)

La figure 2.13 montre le comportement du drone (qui a été présenté dans le paragraphe 2.3.2.1) dans le cas d'une architecture Ethernet commutée à 10 Mb/s implémentant le protocole d'ordonnancement WRR et soumise à un trafic de fond qui perturbe la commande. Pour la figure 2.13, le poids du trafic temps-réel a été fixé à 9 et celui du trafic de fond a été fixé à 3.

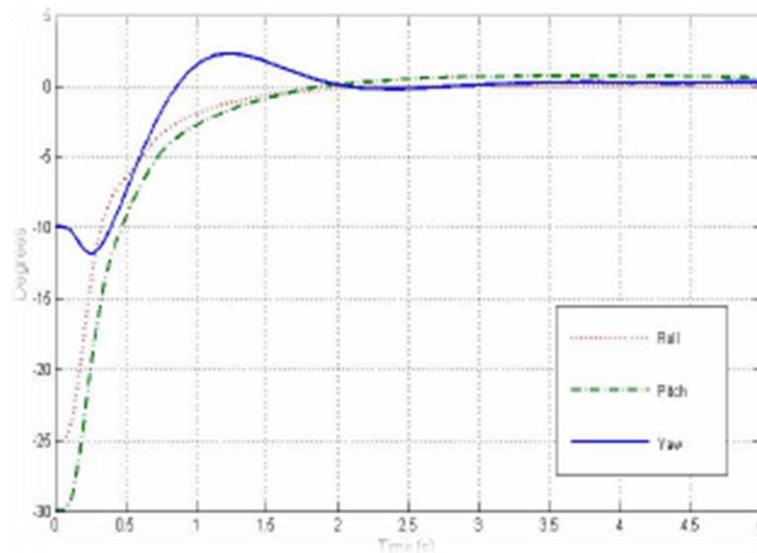


Figure 2.13. Ethernet commuté à 10 Mb/s, avec une CdS WRR et avec un trafic de fond non synchronisé (Diouri *et al.*, 2008)

Une comparaison entre les figures 2.11.f et 2.13 montre que grâce au WRR, on arrive à stabiliser la commande du drone même en présence d'un trafic de fond (figure 2.13). En effet, les angles ( $\varphi$ ,  $\theta$ ,  $\psi$ ) atteignent bien la consigne qui a été fixée à (0, 0, 0) (à noter que sur la figure 2.13 : « *Roll* » correspond à  $\varphi$ , « *Pitch* » correspond à  $\theta$  et « *Yaw* » correspond à  $\psi$ ). D'autre part, on remarque d'après la figure 2.13 que le réseau Ethernet commuté implémentant le WRR nous a permis de réduire l'impact de la charge sur la commande du système (figures 2.11.d et 2.11.f) et nous a permis d'avoir un comportement du système similaire à celui obtenu en utilisant le réseau CAN (figure 2.11.e). Les valeurs des poids appliquées pour cette simulation seront discutées dans la suite de ce mémoire (paragraphe 3.7.1)

### 2.3.3. Simulation d'architecture Ethernet commutée sous OPNET (OPNET, 2005)

#### 2.3.3.1. Description de l'architecture considérée dans la thèse comme fil conducteur

Nous allons nous baser sur l'architecture Ethernet commutée représentée à la figure 2.14 : dans celle-ci le flux du contrôleur vers le process (et inversement du process vers le contrôleur) traverse les deux commutateurs. Cette architecture se compose de deux stations qui envoient du trafic temps-réel (contrôleur et process), de quatre autres stations qui génèrent du trafic de fond (Générateurs de trafic n° 1, n° 2, n° 3 et n° 4) et de deux commutateurs Ethernet Cisco Catalyst 2950 (figure 2.14). Dans la suite de la thèse, le comportement du contrôleur et le comportement du process seront identiques à chaque fois que l'on référencera la figure 2.14.

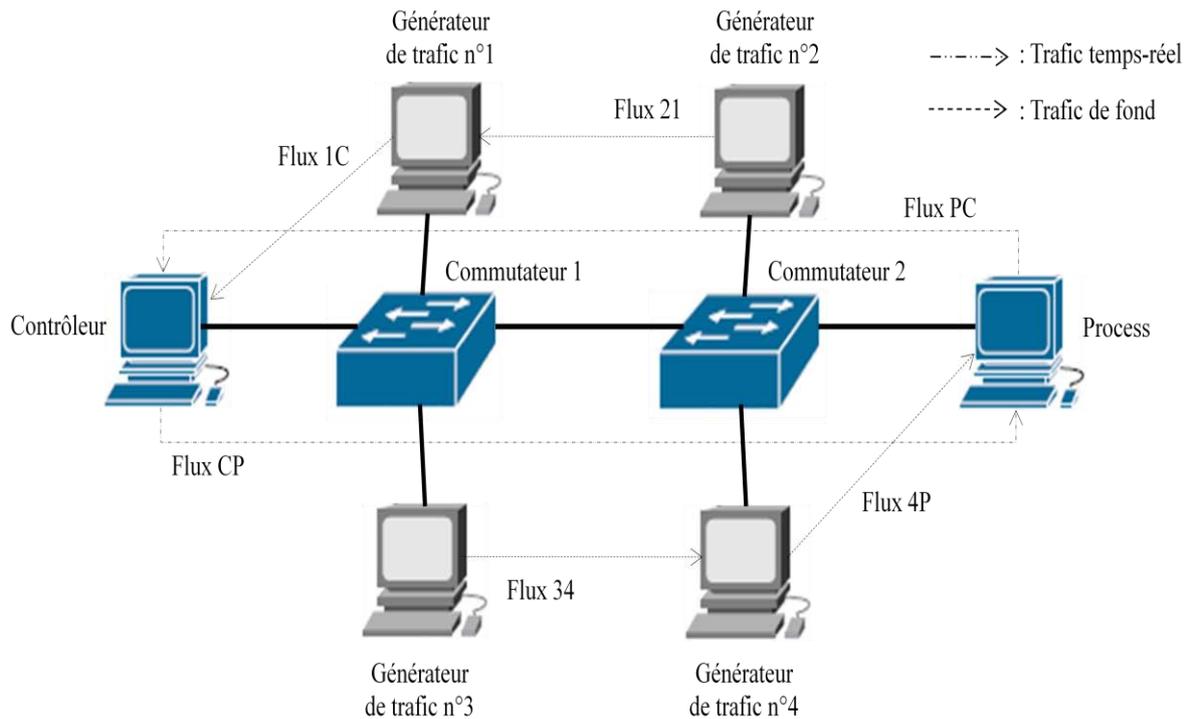


Figure 2.14. Architecture Ethernet commutée étudiée dans la thèse

Les deux commutateurs choisis vont permettre de mettre en place les protocoles d'ordonnancement WRR. Dans cette architecture, tous les liens sont à 10 Mb/s en mode *full-duplex*. On dispose de deux classes de trafic. La première correspond au trafic temps-réel (du contrôleur vers le process et du process vers le contrôleur) et la deuxième correspond au trafic de fond (en pointillés sur la figure 2.14). La taille des paquets temps-réel est de 125 octets et un paquet est envoyé toutes les 10 ms. Cette période correspond au seuil du délai à ne pas dépasser fixé par le système temps-réel. La taille des paquets des flux de fond varie autour de 1500 octets selon une loi exponentielle. L'inter-arrivée des paquets du trafic de fond varie également suivant une loi exponentielle de telle sorte que la moyenne de la charge soit égale à 7 Mb/s.

L'équation du contrôleur échantillonné avec une période de 10 ms est :

$$U[k] = U[k - 1] + 0.23 \times E[k] - 0.1435 \times E[k - 1]$$

L'équation du process échantillonné avec une période de 10 ms est :

$$Y[k] = 1.32866888032377 \times Y[k - 1] - 0.3534546819578 \times Y[k - 2] \\ + 0.0290119551965 \times U[k] + 0.02055964807308 \times U[k - 1]$$

### 2.3.3.2. Simulation du comportement de l'architecture Ethernet commutée sur le logiciel OPNET

La figure 2.15 représente l'architecture Ethernet commutée (présentée dans le paragraphe 2.3.3.1). Dans la suite, cette architecture est étudiée en utilisant le logiciel OPNET. L'utilisation d'un tel logiciel s'avère importante car il permet d'une part de simuler parfaitement les réseaux et d'autre part de faciliter les études statistiques que nous avons été amenés à faire. En effet, c'est un logiciel qui est rapide quand il s'agit de faire plusieurs longues simulations avec des tirages différents de variables aléatoires et il dispose également d'outils statistiques qui permettent de faire des calculs rapidement (intervalle de confiance, valeur maximale du délai de chaque simulation, etc.). Ces études statistiques ont servi notamment à l'identification du délai en utilisant la logique floue (chapitre 4).

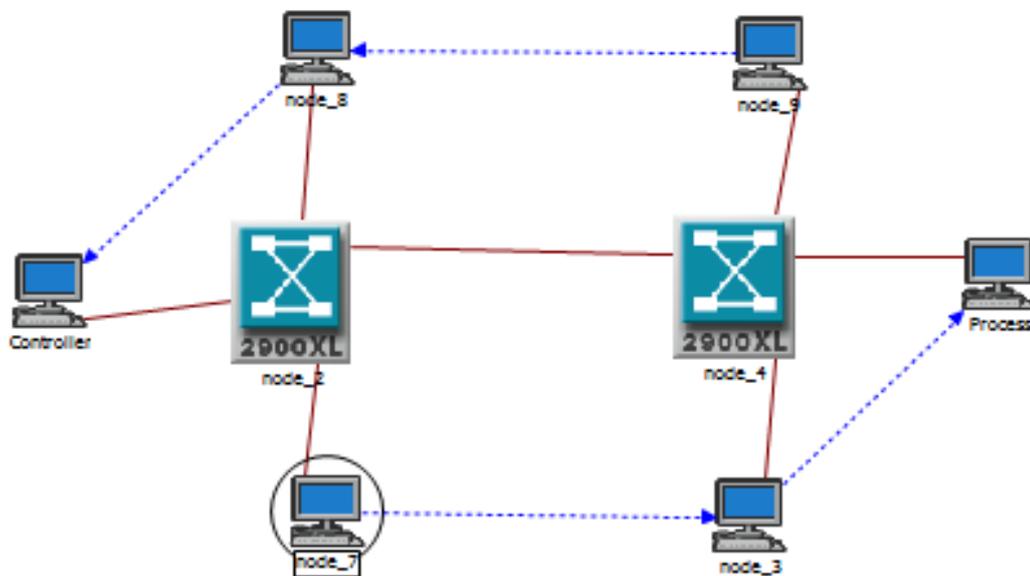


Figure 2.15. Architecture Ethernet commutée étudiée en utilisant le logiciel OPNET (OPNET, 2005)

La durée de chaque simulation est de 10 minutes et la consigne varie de manière périodique entre -1 et 1. Dans la suite, l'étude du SCR est réalisée sans classification de service puis avec classification de service. Mais au préalable, il a fallu implémenter un modèle du SCR sous le logiciel de simulation OPNET.

### 2.3.3.3. Ajout d'un modèle de comportement du SCR sous OPNET

Dans la version standard d'OPNET, il est impossible de simuler le comportement des SCR. Ainsi, au sein de notre équipe de recherche (Systèmes en Réseaux) au CRAN, nous avons développé des nœuds « automate » pour implanter des algorithmes de contrôle/commande, de cartes déportées de capteurs et d'actionneurs (figure 2.16).

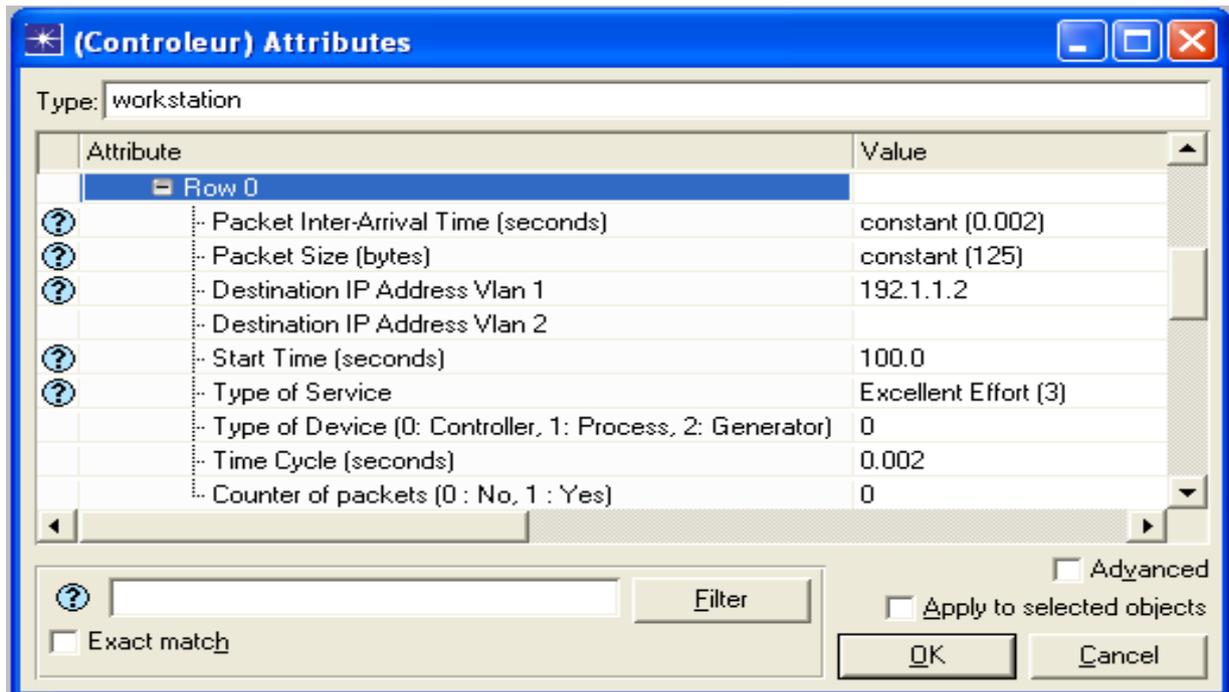
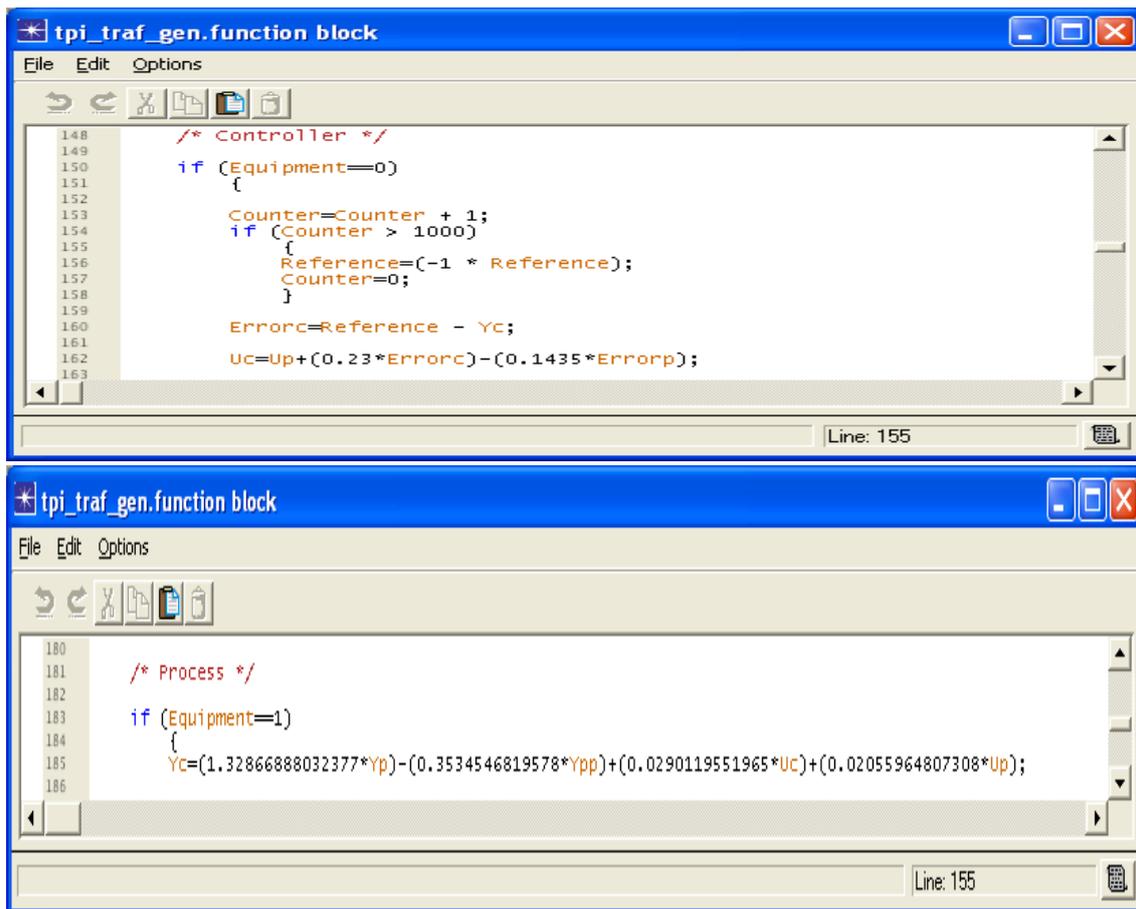


Figure 2.16. Nœuds pour les SCR sur OPNET

La figure 2.16 présente l'interface qui a été programmée pour modéliser des contrôleurs, des capteurs, des actionneurs ou encore des applications externes chargeant le réseau. On distingue dans cette interface deux types de cycle à configurer : le cycle automate exécutant périodiquement les tâches de niveau applicatif et le cycle de communication spécifiant l'inter-arrivée des messages sur le réseau. La figure 2.17 montre la programmation des lois de commande ou de l'équation simulant le process qu'il est possible de changer pour étudier n'importe quel type d'applications.



```
148 /* Controller */
149
150 if (Equipment==0)
151 {
152
153     Counter=Counter + 1;
154     if (Counter > 1000)
155     {
156         Reference=(-1 * Reference);
157         Counter=0;
158     }
159
160     Errorc=Reference - Yc;
161
162     Uc=Up+(0.23*Errorc)-(0.1435*Errorp);
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181 /* Process */
182
183 if (Equipment==1)
184 {
185     Yc=(1.32866888032377*Yp)-(0.3534546819578*Ypp)+(0.0290119551965*Uc)+(0.02055964807308*Up);
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Figure 2.17. Parties du code permettant de modifier les équations du contrôleur et du process

#### 2.3.3.4. Impact des délais sur la commande du process

La figure 2.18 montre que les délais causés par la présence de la charge sont conséquents et la figure 2.19 montre que le trafic de fond rend instable le système de contrôle/commande. Les saturations observables à la figure 2.19 correspondent à des niveaux où la commande tend vers l'infini (instabilité). Une valeur maximale est introduite sur la commande pour éviter des problèmes de dépassement mémoire du logiciel OPNET.

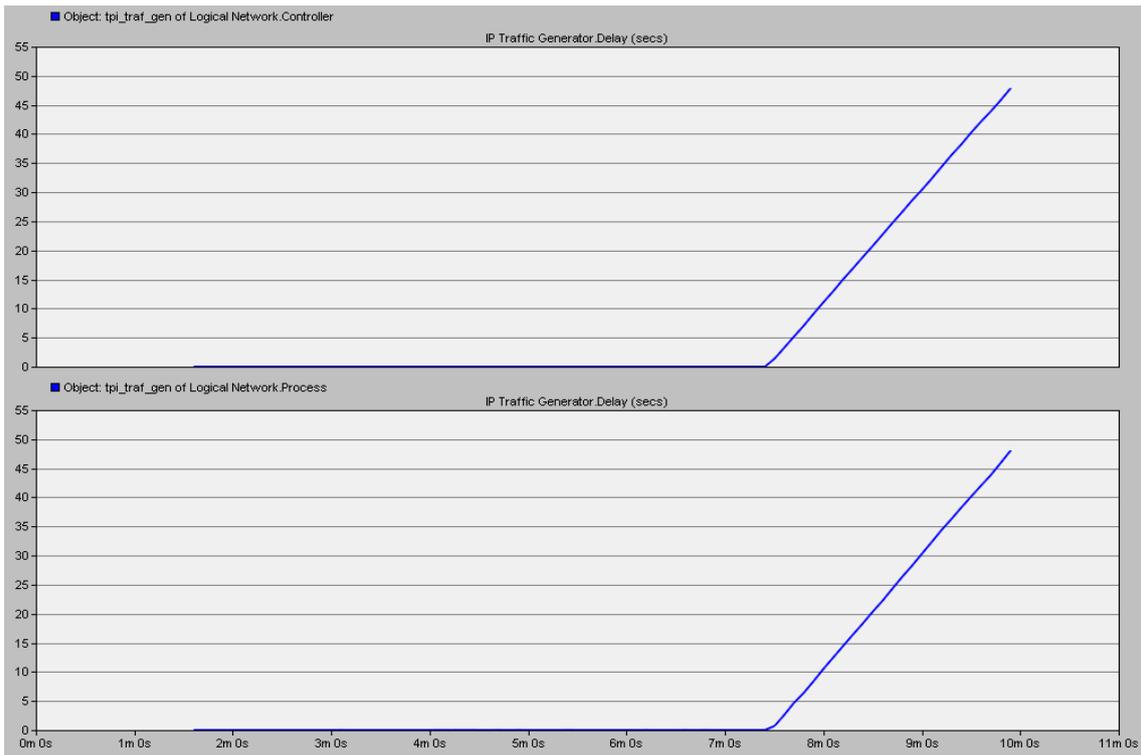


Figure 2.18. Evolution des délais aux niveaux du contrôleur et du process en présence de la charge

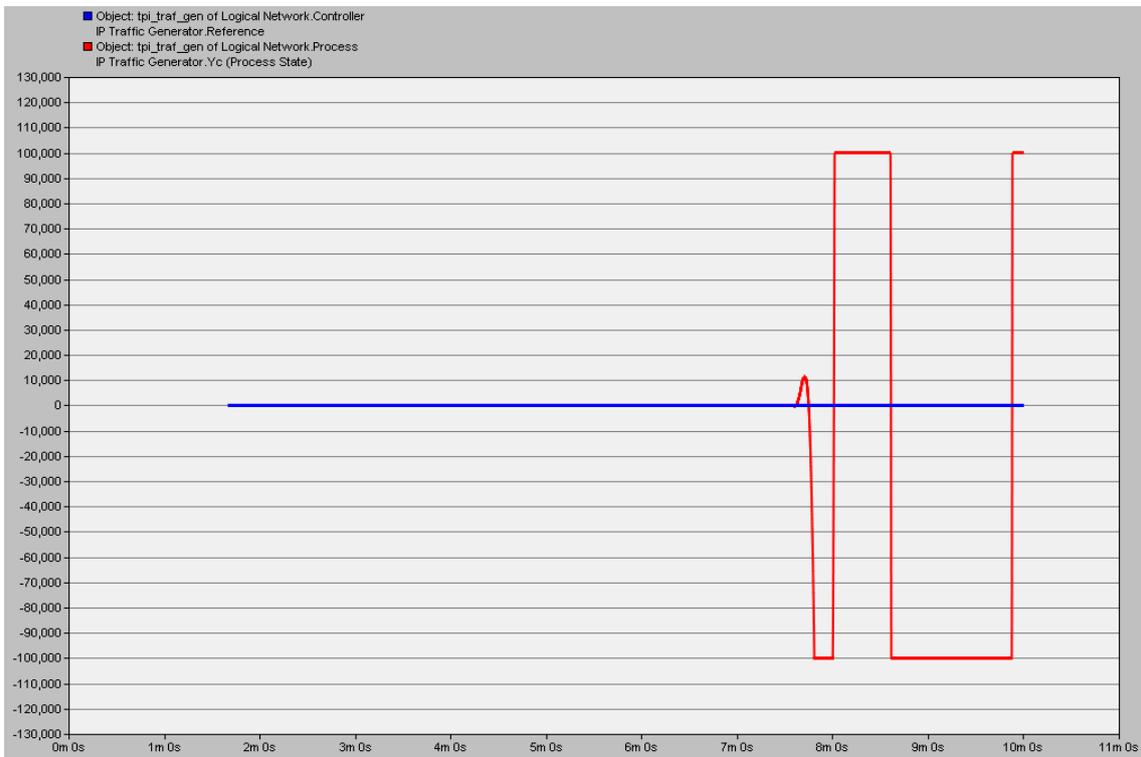


Figure 2.19. Comportement du SCR étudié dans la thèse en présence de la charge

### 2.3.3.5. Performance du WRR

On s'intéresse ici à l'architecture présentée dans les paragraphes 2.3.3.1. Comme il a été signalé dans le paragraphe 2.3.3.2, cette architecture a été étudiée en utilisant le logiciel OPNET (OPNET, 2005). Le logiciel OPNET permet de simuler les réseaux Ethernet commutés implémentant le protocole d'ordonnancement WRR statique. La figure 2.20 représente le comportement des délais subis par le trafic temps-réel et la figure 2.21 représente le comportement de la sortie du système et cela lorsque le poids du WRR affecté au trafic temps-réel est fixé à 30 ( $w_{Traffic\_TR} = 30$ ) et celui du trafic de fond est fixé à 1 ( $w_{Traffic\_TF} = 1$ ).

La mise en place de ces poids offre donc plus de bande passante au trafic temps-réel qui a pour conséquence de réduire ses délais et donc de rendre stable le système (figure 2.21).

Le logiciel OPNET permet de simuler les réseaux Ethernet commutés implémentant le protocole d'ordonnancement WRR statique. Cependant, il est impossible d'évaluer les performances d'un réseau Ethernet commuté implémentant une adaptation en ligne des poids du WRR. Ainsi, il a fallu ajouter cette possibilité et cela afin de pouvoir introduire une stratégie de placement de poids en ligne permettant de satisfaire les contraintes temps-réel souples de l'application tout en évitant au mieux les situations de famine (chapitre 4).

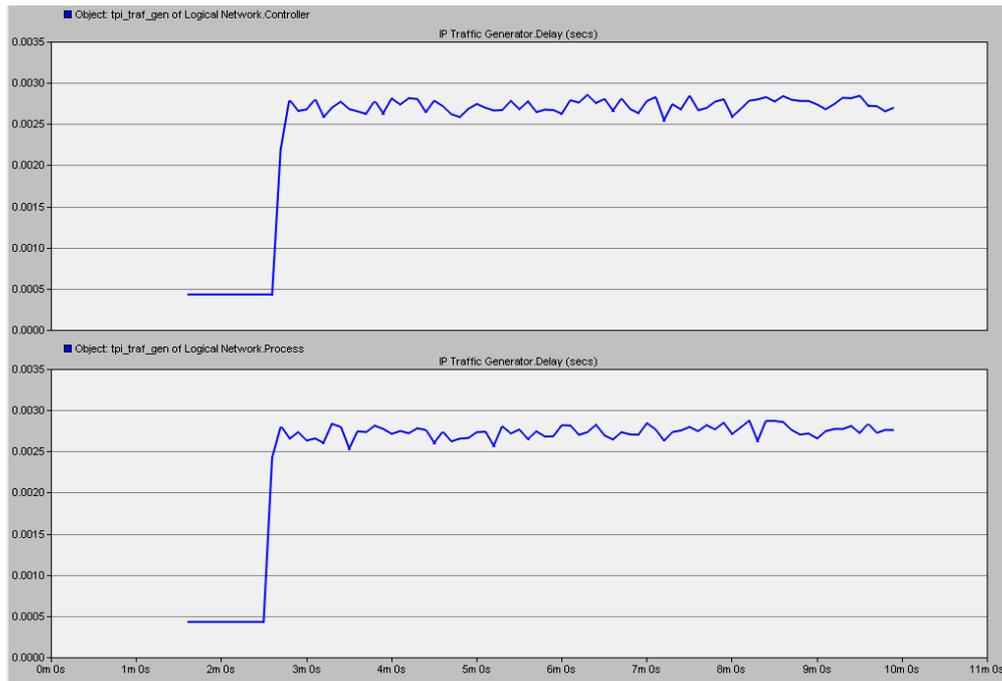


Figure 2.20. Evolution des délais aux niveaux du contrôleur et du process en présence de la charge  
(cas du WRR avec  $w_{Trafic_{TR}} = 30$  et  $w_{Trafic_{TF}} = 1$ )

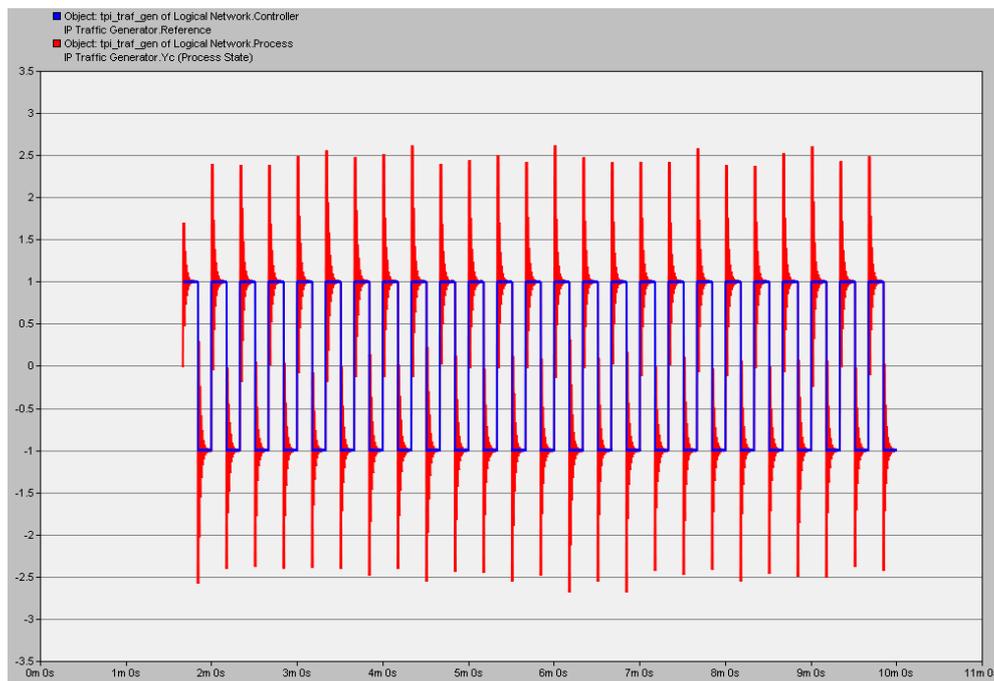


Figure 2.21. Comportement du SCR considéré dans la thèse avec WRR  
( $w_{Trafic_{TR}} = 30$  et  $w_{Trafic_{TF}} = 1$ )

## 2.4. Conclusion

On conclut à partir de ce chapitre, que l'ordonnanceur WRR peut protéger le trafic temps-réel face à l'apparition d'autres trafics perturbateurs. Cependant, les réglages du WRR présentés dans ce chapitre ont été positionnés de façon empirique et donc n'apportent pas forcément des garanties en termes de délai maximal ou de bande passante optimale offerte aux flux non contraints temporellement. Aussi l'objectif de cette thèse est de proposer des modèles mathématiques du délai en fonction des poids du WRR de façon à pouvoir régler très finement l'ordonnanceur pour garantir une qualité de service pour le trafic temps-réel tout en donnant le plus possible de bande passante aux autres trafics.

A noter, que l'analyse du délai pour les réseaux Ethernet commutés a été précédemment étudiée par (Jasperneite et al., 2002) (Loeser et Hartig, 2004) (Georges, 2005). Précédemment, (Loeser et Hartig, 2004) ont utilisé le Dresden Real-time Operating System (DROPS) (Hartig et al., 1999). (Loeser et Hartig, 2004) ont montré l'intérêt d'utiliser un régulateur de trafic dans le réseau Ethernet commuté pour borner les délais. Pour réguler le trafic, ils ont implémenté un régulateur de trafic qui utilise le principe du seau à de jetons (*token bucket*) (Turner, 1986) au niveau de chaque nœud. Ce régulateur de trafic fournit une courbe d'arrivée ayant la forme T-SPEC. La forme T-SPEC de la courbe d'arrivée est donnée par l'équation suivante :  $\alpha(t) = \min(Ct + M, rt + b)$  (avec  $C$  est le débit de transmission maximal,  $r$  est le débit moyen à long terme,  $M$  est la taille maximale de paquet et  $b$  représente la rafale maximale du trafic (*burstiness*)). (Loeser et Hartig, 2004) ont considéré la forme T-SPEC de la courbe d'arrivée pour calculer le délai maximale. En utilisant la théorie du calcul réseau (Le Boudec et Thiran, 2004) (Chang, 2000) (Cruz, 1991a) (Cruz, 1991b), (Loeser et Hartig, 2004) ont établi que le délai maximal d'un commutateur est égal à la durée de transmission des rafales des T-SPECs avec la bande passante maximale des ports plus le délai imposé par la structure du commutateur. Une méthode de calcul des délais maxima de bout en bout, basée sur la théorie du calcul réseau, pour les réseaux Ethernet commutés utilisant la priorité stricte a été proposée dans (Georges, 2005).

Par rapport à ces travaux, notre apport est de proposer des modèles de délai pour les réseaux Ethernet commutés à classification de services basée sur l'ordonnanceur WRR. Deux approches sont envisagées. Dans le chapitre 3, le but est de garantir qu'aucun dépassement du seuil du retard fixé par les systèmes à contraintes temps-réel strictes n'ait lieu. On se base alors sur la théorie du calcul réseau et une configuration hors-ligne du réseau. Le problème est que le calcul réseau est une théorie qui pourra donner plus de bande passante que nécessaire au trafic temps-réel au détriment des autres trafics car cette théorie repose sur le calcul du délai maximal. Aussi, dans le chapitre 4, nous considérons un autre type d'application dit temps-réel souple qui peut accepter des dépassements sur les contraintes de délais. Nous proposons alors un ajustement dynamique (en ligne) des poids du WRR en fonction de l'évolution des trafics en nous appuyant sur la logique floue pour la modélisation du délai.



**Partie II :**

**Contributions**



# Chapitre 3 :

## Approche commande de réseau pour les SCR à contraintes temps-réel strictes

### 3.1. Introduction

Dans notre étude, nous avons choisi d'utiliser le réseau Ethernet commuté implémentant le protocole d'ordonnancement WRR. Le choix de WRR doit permettre de satisfaire les contraintes applicatives des systèmes à contraintes temps-réel strictes tout en maximisant les ressources allouées au trafic non contraint temporellement. Le but consiste donc à s'appuyer sur une méthodologie déterministe, en se basant sur la théorie du calcul réseau principalement développée dans (Cruz, 1991a) (Cruz, 1991b) (Chang, 2000) (Le Boudec et Thiran, 2004). Cette théorie nous permettra de déterminer une configuration des poids du WRR robuste au trafic concurrent. La particularité de cette recherche consistera à minimiser les situations de famine pour le trafic de fond. Ainsi, le service offert par un ordonnanceur WRR est d'abord modélisé. Ensuite, un majorant du délai de bout en bout est établi grâce à cette modélisation du service. Enfin, un algorithme permettant d'établir les poids qui maximisent le service offert au trafic concurrent tout en garantissant les contraintes temporelles strictes de l'application temps-réel est proposé.

La démarche méthodologique générale préconisée est dans un premier temps d'identifier les limites de stabilité du système à contrôler, correspondant à la Qualité de Contrôle (QdC) imposée par l'application. La limite temporelle doit être traduite en contrainte temporelle. Dans une deuxième étape, il faut identifier le trafic supporté par le réseau (chemin, volume, fréquence, etc.) et l'architecture de communication (interconnexion des équipements, capacité des équipements réseau, débit des lignes, etc.). A partir de ces données, le calcul réseau est appliqué pour estimer des majorants de délais en fonction des poids alloués dans les ordonnanceurs WRR. Il s'agit alors de trouver un réglage des poids robuste au trafic concurrent tout en minimisant les situations de famine. Dans cette démarche, la commande de réseau se fait hors-ligne et n'est pas modifiée en cours de fonctionnement du SCR.

## 3.2. Présentation du calcul réseau

Le calcul réseau repose sur la théorie mathématique des dioïdes et en particulier les dioïdes  $(\min, +)$  et  $(\max, +)$  (on parle généralement d'algèbre  $(\min, +)$  et d'algèbre  $(\max, +)$ ). La différence entre  $(\min, +)$  et la théorie des systèmes traditionnels est que  $(\min, +)$  considère une autre algèbre où les opérations changent de la manière suivante : l'addition  $(+)$  revient à définir l'infimum (ou le minimum s'il existe) et la multiplication  $(\times)$  devient l'addition  $(+)$ .

Depuis quelques temps, il existe quelques logiciels implémentant des fonctionnalités relatives au calcul réseau. On pourra citer notamment :

- Le CyNC (Schioler *et al.*, 2007) qui est une *Toolbox* sur le logiciel Matlab/Simulink.
- Le logiciel COINC (Bouillard et Thierry, 2008) qui a été réalisé en C++ et qui a été fait dans le cadre du projet COINC (COINC Project, INRIA 2006).
- Le « Disco network calculator » (Schmitt et Zdarsky, 2006) qui a été réalisé en JAVA.

Dans le contexte des applications ayant des contraintes temps-réel strictes, le calcul réseau s'avère être une théorie adéquate dans la mesure où il permet d'identifier (entre autres) des bornes de délai. Le calcul réseau a été utilisé pour les processeurs réseaux (Thiele *et al.*, 2001), les réseaux de capteurs (Schmitt et Roedig, 2005) et sur les architectures Ethernet commutées (Georges *et al.*, 2005) (Jasperneite *et al.*, 2002).

Dans notre cas, on considère qu'il n'y a pas de pertes de paquets sur le réseau : cette hypothèse est concevable pour le réseau Ethernet pour lequel il est généralement accepté que les pertes de paquets sont négligeables, contrairement par exemple aux réseaux sans-fil. La théorie du calcul réseau se base sur des concepts tels que les courbes d'arrivées et les courbes de services (Le Boudec et Thiran, 2004).

### 3.2.1. Quelques définitions du calcul réseau

Le calcul réseau (Cruz, 1991a) (Cruz, 1991b) (Le Boudec et Thiran, 2004) (Chang, 2000) est une théorie permettant l'appréhension de phénomènes rencontrés par les flux durant leur traversée des équipements d'interconnexion. Elle permet, entre autres, de comprendre des propriétés fondamentales de l'ordonnancement et de dimensionnement du buffer et du délai. Le calcul réseau se distingue par une représentation déterministe du trafic. Ainsi, pour modéliser un trafic, on se base sur le fait que l'arrivée des messages est soumise à certaines contraintes. Pour cela, chaque flux de données est décrit par des fonctions cumulatives qui représentent la quantité cumulée de données qui transite par l'entrée et/ou par la sortie du système à partir d'un instant donné ( $t = 0$ ). Dans la mesure où plusieurs notations ont pu être définies dans la littérature pour l'expression de termes identiques, nous adopterons par la suite les notations et définitions données par (Le Boudec et Thiran, 2004) :

- $R(t)$  définit le nombre de bits qui ont transité par l'entrée d'un système durant l'intervalle  $[0, t]$ .  $R(t)$  est appelée fonction d'entrée.
- $R^*(t)$  représente le nombre de bits cumulés qui ont transité par la sortie d'un système durant l'intervalle  $[0, t]$ .  $R^*(t)$  est appelée fonction de sortie.

Le calcul réseau fait aussi appel à des définitions telles que l'arriéré de traitement, la courbe d'arrivée et la courbe de service (Parekh et Gallager, 1993) (Le Boudec et Thiran, 2004) :

L'arriéré de traitement  $x(t)$  (Le Boudec et Thiran, 2004) : Pour un système sans pertes, l'arriéré de traitement à un instant  $t$  donné est :  $x(t) = R(t) - R^*(t)$ .

Courbe d'arrivée  $\alpha(t)$  (Le Boudec et Thiran, 2004) : Soit  $\alpha$  une fonction croissante au sens large définie pour  $t \geq 0$ , on dit qu'un flux  $R$  est contraint par  $\alpha$  si et seulement si : Pour tout  $s \leq t$  :  $R(t) - R(s) \leq \alpha(t - s)$ . On dit que  $R$  a  $\alpha$  comme courbe d'arrivée.

Courbe de service  $\beta(t)$  (Le Boudec et Thiran, 2004) : On considère un système  $S$  et un flux à travers  $S$  avec des fonctions d'entrée et de sortie  $R$  et  $R^*$ . On dit que  $S$  offre au flux une courbe de service  $\beta$  si et seulement si  $\beta$  est croissante au sens large,  $\beta(0) = 0$  et pour tout  $t \geq 0$  :  $R^*(t) \geq \inf_{s \leq t} (R(s) + \beta(t - s))$ .

Courbe de service stricte (Le Boudec et Thiran, 2004) : On dit qu'un système  $S$  offre une courbe de service stricte  $\beta$  à un flux si, durant n'importe quelle période de durée  $u$  ayant un arriéré de traitement, la sortie du flux est au moins égale à  $\beta(u)$ .

Il a été montré qu'il est possible d'exprimer les définitions précédentes dans l'algèbre  $(\mathbb{R} \cup \{+\infty\}, \min, +)$  et cela en utilisant les notations suivantes :

- $\wedge$  représente l'infimum (ou le minimum quand il existe).
- $\vee$  représente le supremum (ou le maximum quand il existe).
- $\otimes$  représente la convolution min-plus :  $(f_1 \otimes f_2)(t) = \inf_{0 \leq s \leq t} (f_1(s) + f_2(t - s))$
- $\oslash$  représente la déconvolution min-plus :  $(f_1 \oslash f_2)(t) = \sup_{u \geq 0} (f_1(t + u) - f_2(u))$

Ainsi, les nouvelles définitions obtenues sont les suivantes :

Courbe d'arrivée (Le Boudec et Thiran, 2004) : On dit qu'un flux  $R$  est contraint par une courbe d'arrivée  $\alpha$  si et seulement si  $R \leq R \otimes \alpha$ .

Courbe de service (Le Boudec et Thiran, 2004) : On considère un système  $S$  et un flux à travers  $S$  avec des fonctions d'entrée et de sortie  $R$  et  $R^*$ . On dit que  $S$  offre au flux une courbe de service  $\beta$  si et seulement si  $\beta$  est croissante au sens large,  $\beta(0) = 0$  et  $R^* \geq R \otimes \beta$ .

A partir des trois théorèmes présentés dans (Le Boudec et Thiran, 2004) relatifs à la borne de l'arrière de traitement (théorème 1.4.1 de (Le Boudec et Thiran, 2004)), à la borne du délai (théorème 1.4.2 de (Le Boudec et Thiran, 2004)) et du flux de sortie (théorème 1.4.3 de (Le Boudec et Thiran, 2004)), on peut établir simplement que le délai maximal correspond à la distance horizontale maximale entre la courbe d'arrivée et de service et que l'arrière de traitement maximal est la distance verticale maximale entre la courbe d'arrivée et de service (figure 3.1).

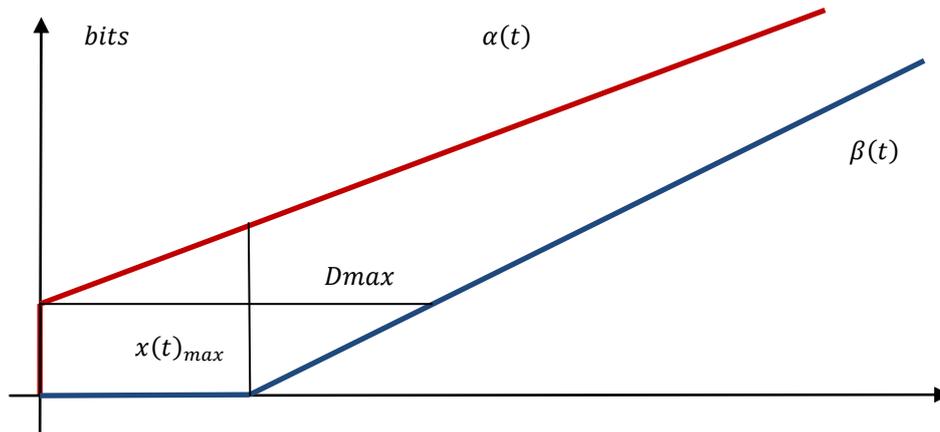


Figure 3.1. Majorant du délai  $D_{max}$  et de l'arrière de traitement  $x(t)_{max}$

Il est aussi intéressant de présenter le théorème 1.4.3 de (Le Boudec et Thiran, 2004) car il sera utilisé par la suite. Dans ce théorème (Le Boudec et Thiran, 2004) ont établi la courbe d'arrivée du flux de sortie dans le cas d'un flux ayant une courbe d'arrivée  $\alpha$  traversant un système qui offre une courbe de service  $\beta$  :

Théorème « flux de sortie » (Le Boudec et Thiran, 2004) : On considère un flux contraint par la courbe d'arrivée  $\alpha$  traversant un système qui offre une courbe de service  $\beta$ . Le flux de sortie est contraint par la courbe d'arrivée  $\alpha^* = \alpha \oslash \beta$ .

(Le Boudec et Thiran, 2004) ont également établi la courbe de service offerte à un flux traversant deux systèmes correspondant à la composition des courbes de service offertes par chacun des systèmes :

Théorème « Concaténation des nœuds » (Le Boudec et Thiran, 2004) : Soit un flux traversant les systèmes  $S_1$  et  $S_2$  en séquence. On suppose que  $S_i$  offre une courbe de service  $\beta_i, i = 1, 2$  au flux. Alors le réseau correspondant à la traversée des deux systèmes offre une courbe de service  $\beta_1 \otimes \beta_2$  au flux.

Par ce théorème, (Le Boudec et Thiran, 2004) ont pu mettre l'accent sur le phénomène du *Pay Bursts Only Once*. Ainsi, (Le Boudec et Thiran, 2004) ont considéré la concaténation de deux nœuds offrant chacun une courbe de service de type *rate-latency*  $\beta_{R_i, T_i} = R_i(t - T_i)^+$  avec

$i = 1$  ou  $2$  et ils ont considéré une courbe d'arrivée ayant comme expression  $rt + b$  pour  $t > 0$  (tel que  $r < R_1$  et  $r < R_2$ ) et cela pour comparer les bornes de délais obtenues en :

- Appliquant la courbe de service du réseau ce qui a donné une borne de délai  $D_0 = \frac{b}{R} + T_0$  avec  $R = \min(R_1, R_2)$  et  $T_0 = T_1 + T_2$ .
- Calculant de manière itérative les bornes individuelles sur chaque nœud ce qui a donné une borne de délai égale à  $D_1 + D_2 = \frac{b}{R_1} + \frac{(b+rT_1)}{R_2} + T_0$ .

(Le Boudec et Thiran, 2004) ont constaté que  $D_0 < D_1 + D_2$  et que  $D_1 + D_2$  contient deux fois l'élément  $\frac{b}{R_i}$  alors que  $D_0$  le contient qu'une seule fois : c'est le phénomène du *Pay Bursts Only Once*.

On considère le cas représenté par la figure 3.2 où un paquet appartenant à une classe de trafic donnée traverse  $m$  commutateurs offrant chacun une courbe de service  $\beta_j$  à la classe de trafic considérée (avec  $j$  correspond au  $j^{\text{ème}}$  commutateur traversé et  $j \in \llbracket 1, m \rrbracket$ ).

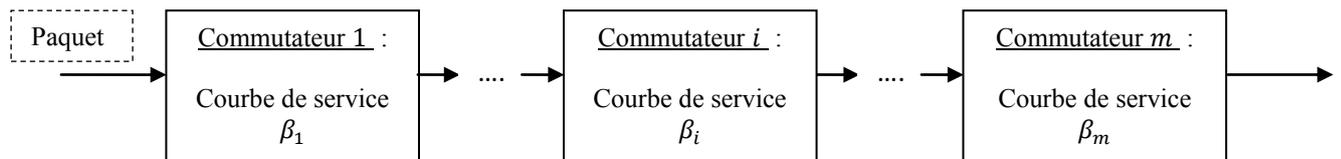


Figure 3.2. Cas d'un paquet traversant  $m$  commutateurs

Pour calculer le délai maximal dans le cas représenté dans la figure 3.2, il suffit ainsi de calculer le délai dans le cas suivant ( $\beta_i$  représente la courbe de service du  $i^{\text{ème}}$  commutateur traversé) (figure 3.3) :

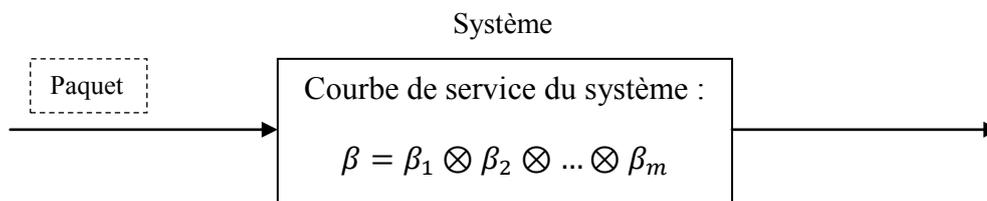


Figure 3.3. Cas d'un paquet traversant  $m$  commutateurs en utilisant le théorème de la concaténation des nœuds (Le Boudec et Thiran, 2004)

Remarques :

- Il a été montré dans le paragraphe 1.4.3 de (Le Boudec et Thiran, 2004) que les bornes obtenues en considérant la courbe de service globale sont meilleures que celles obtenues en considérant chaque buffer de manière isolée.
- De même dans le paragraphe 1.4.3 de (Le Boudec et Thiran, 2004), il a été énoncé que si l'on a deux nœuds offrant chacun une courbe de service de type *rate-latency*  $\beta_{R_i, T_i}$  ( $i = 1, 2$ ) alors un simple calcul donne ce qui suit :

$$\beta_{R_1, T_1} \otimes \beta_{R_2, T_2} = \beta_{\min(R_1, R_2), T_1 + T_2}$$

A partir de cette expression, (Le Boudec et Thiran, 2004) déduisent que la concaténation de nœuds de type *rate-latency* consiste à accumuler les latences et à prendre le minimum des débits.

**3.2.2. Remarques**

Comme il a été montré dans ce paragraphe, il est possible de définir le majorant du délai et de l'arrière de traitement en se basant uniquement sur les courbes d'arrivée et de service. Dans ce paragraphe, on voit aussi qu'il est possible de calculer le majorant du délai de bout en bout dans le cas d'un flux qui traverse plusieurs systèmes. Néanmoins, les résultats précédents ne sont établis que dans le cas où le flux est le seul à traverser les différents équipements. On notera par la suite (paragraphe 3.3.6) que l'analyse pourra se complexifier pour des réseaux à plusieurs flux. Pour résoudre ce problème, nous proposerons une modélisation du service spécifique à un type de flux couplée avec une prise en compte de l'agrégation. L'agrégation de trafic reste encore un point d'étude dans la littérature concernant le calcul réseau.

En conclusion, la difficulté principale du calcul réseau réside dans la modélisation adéquate (non pessimiste) des courbes d'arrivée et de service de chaque système. Dans le paragraphe suivant, l'objectif sera de modéliser la courbe de service offerte par un commutateur à chacune des classes de services (ou files) du protocole d'ordonnancement WRR.

**3.3. Modèle du commutateur et modélisation de sa courbe de service****3.3.1. De l'architecture en bus vers une architecture commutée**

Pour améliorer les performances d'Ethernet, il y a de nombreuses possibilités dont l'une consiste à augmenter le débit et la seconde consiste à introduire des commutateurs tels qu'ils ont été définis dans le standard 802.1D (IEEE, 1998).

Concernant la première solution, il s'avère qu'elle ne permet pas de résoudre le problème de saturation du réseau à cause de la gestion des collisions du CSMA/CD (voir l'analyse du chapitre précédent sur le drone (Diouri *et al.*, 2008)).

La deuxième solution quant à elle est plus intéressante dans la mesure où le domaine de collisions est réduit aux ports de sortie du commutateur. Il faut cependant modérer ces propos en soulignant que chaque commutateur traversé ajoute une latence au trafic qui n'existait pas sur les architectures Ethernet initiales (en bus). C'est-à-dire si le trafic sur le réseau est faible, il est préférable d'utiliser de l'Ethernet en bus. Mais notre étude ne s'inscrit pas dans cette situation.

Dans la norme IEEE 802.1D (IEEE, 1998), les trois étapes assurées par les commutateurs sont les suivantes :

- Dans un premier temps, le commutateur reçoit la trame.
- Ensuite, le commutateur commence un processus de retransmission en tenant compte des informations sur le port de sortie.
- Puis le commutateur transmet la trame.

L'objectif est de se baser sur un modèle de commutateur qui respecte la norme IEEE 802.1D (IEEE, 1998). Cela permet par la suite d'appliquer facilement les résultats obtenus à des commutateurs qui ont été fabriqués en respectant cette norme.

### 3.3.2. Modèle de commutateur considéré

Dans notre cas, nous allons reprendre le modèle du commutateur proposé dans (Georges, 2005) (figure 3.4), repris aussi par (Brahimi, 2007).

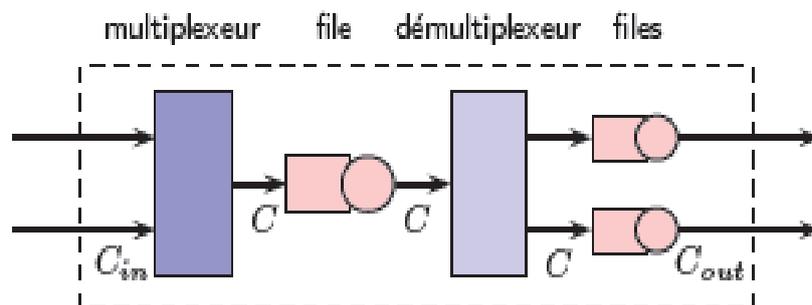


Figure 3.4. Vers un modèle de commutateur 802.1D (Georges, 2005)

La figure 3.5 illustre le modèle de commutateur considéré où à chaque sortie du commutateur on affecte autant de buffers (files FIFO) que de classes différenciées. En l'occurrence, ici on a quatre classes de services, ce qui correspond au nombre de classes implémentées dans les

commutateurs du commerce (par exemple Cisco Catalyst 2950). Le commutateur va donc dispatcher le trafic sur ces files FIFO en fonction de leur niveau de priorité. L'ordonnanceur WRR traite ensuite ces files de la façon suivante : un poids  $w_i$  est affecté à chaque file  $i$  signifiant qu'à chaque cycle un nombre  $w_i$  de paquets est traité pour chaque file  $i$ . L'ordre du cycle est défini de telle sorte que les files sont traitées séquentiellement de la plus haute à la plus basse priorité.

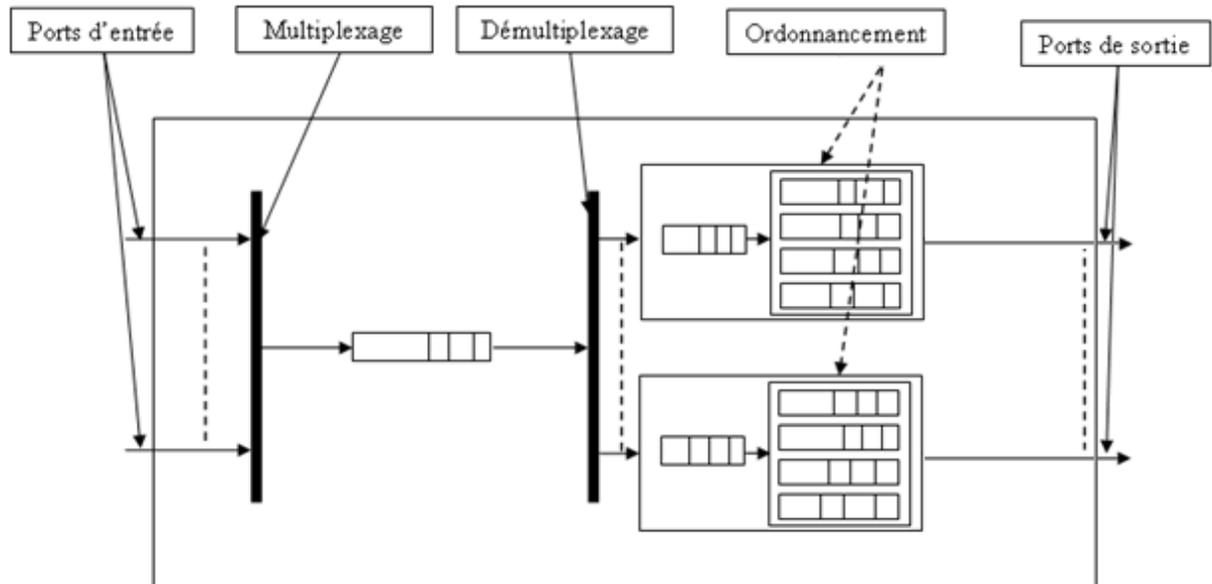


Figure 3.5. Modèle du commutateur incluant le protocole d'ordonnancement WRR

### 3.3.3. Modèle de l'arrivée des données du flux temps-réel

Dans cette étude, on suppose que l'arrivée des trames de contrôle/commande est bornée par une courbe d'arrivée affine. En considérant que la longueur d'une trame temps-réel est constante et que les trames arrivent périodiquement, alors la courbe d'arrivée  $\alpha(t)$  est égale à  $\sigma + \rho t$  (avec  $\sigma$  la quantité maximale du trafic qui peut être contenue dans une rafale et  $\rho$  la borne maximale de l'arrivée moyenne du trafic). Cette courbe d'arrivée est représentée à la figure 3.6. Elle s'applique à la courbe d'arrivée dans le cas d'un lissage de trafic du type seuil percé.

## Lissage par seau percé

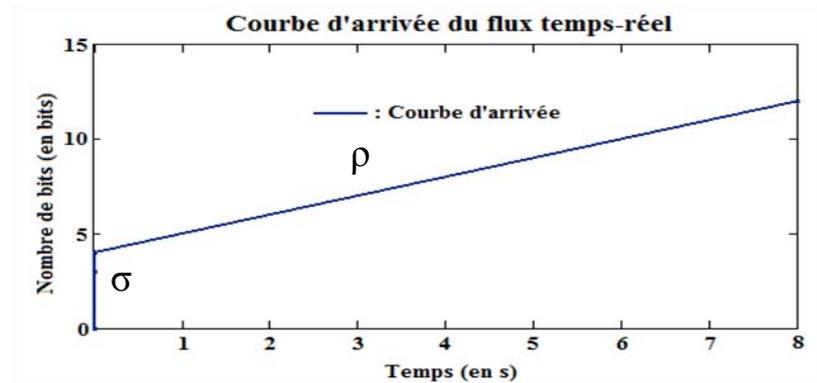
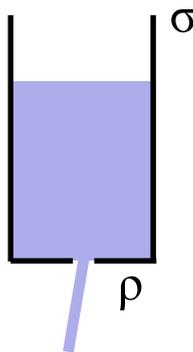


Figure 3.6. Courbe d'arrivée du trafic temps-réel dans le cas d'un lissage par seau percé

La courbe représentée dans la figure 3.6 représente la courbe d'arrivée obtenue dans le cas d'un lissage par seau percé (avec  $\sigma$  est le volume maximal du seau et  $\rho$  représente le débit avec lequel le seau se vide).

### 3.3.4. Courbe de service

Dans cette partie, l'objectif est de modéliser le service offert par une série d'ordonnanceurs WRR.

#### 3.3.4.1. Service offert à une classe de trafic traversant un ordonnanceur WRR

On suppose que chaque ordonnanceur WRR comporte  $n$  classes de trafic et qu'il n'existe qu'un seul flux par classe de trafic étudiée.

Dans la suite, on se base sur les notations suivantes représentées dans le tableau 3.1.

Notations	Significations
$m$	Nombre d'ordonnanceurs WRR que le flux étudié traverse de bout en bout.
$n$	Nombre de classes de trafic.
$w_{i,j}$	Poids offert à la classe de trafic $i$ par le $j^{\text{ème}}$ ordonnanceur WRR traversé par un paquet de celle-ci, avec $1 \leq i \leq n$ et $1 \leq j \leq m$ .
$L_i$	Taille maximale du paquet de la $i^{\text{ème}}$ classe de trafic, avec $1 \leq i \leq n$ .
$C_j$	Capacité du $j^{\text{ème}}$ ordonnanceur WRR traversé par le flux étudié, avec $1 \leq j \leq m$ .
$\beta_{1,j}$	Courbe de service offerte au flux étudié quand un paquet de celui-ci traverse le $j^{\text{ème}}$ ordonnanceur WRR.
$\beta_1$	Courbe de service offerte au flux étudié de bout en bout.

Tableau 3.1. Notations utilisées

Remarques :

Dans la suite, on choisit que le flux temps-réel correspond au flux de la classe de trafic 1. Il est à noter que ce choix ne change absolument rien au raisonnement et que celui-ci reste valable si on avait choisi une autre valeur de  $i$  comprise entre 2 et  $n$ .

Il est aussi à noter qu'ici le flux étudié (temps-réel) n'est dans cette 1<sup>ère</sup> partie uniquement concurrent qu'avec des flux appartenant à des classes de trafic différentes de la sienne.

Dans un premier temps, nous allons nous concentrer sur la courbe de service offerte à la classe de trafic 1 quand un paquet appartenant à celle-ci traverse un seul ordonnanceur WRR pour arriver à destination. Donc, dans ce cas, on considère que  $m = 1$ .

Dans ce cas, les 3 expressions (eq. 3.1) (Diouri *et al.*, 2007a), (eq. 3.2) et (eq. 3.3) qui expriment la courbe de service offerte à la classe de trafic 1 (classe de trafic sur laquelle se porte l'étude du délai de bout en bout) quand un paquet traverse un seul ordonnanceur WRR (commutateur) sont équivalentes. La courbe  $\beta_1$  représentée à la figure 3.7 montre qu'il y a des périodes d'attente dans lesquelles aucun paquet appartenant à la classe de service 1 n'est traité, ainsi que des périodes d'envoi qui dépendent du poids affecté à cette classe de service.

$$\beta_1(t) = \beta_{1,1}(t) = \text{Max} \left\{ \begin{array}{l} C_1 \left( t - \left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) \left[ \frac{t}{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1} \right] \right)^+ ; \\ (C_1 w_{1,1} L_1) \left[ \frac{t}{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1} \right] \end{array} \right\} \quad (\text{eq. 3.1})$$

$$\beta_1(t) = \beta_{1,1}(t) = \text{Sup}_{n_1 \in \mathbb{N}} \left\{ n_1 w_{1,1} L_1 - C_1 \left[ n_1 \frac{\left( w_{1,1} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} - t \right]^+ \right\} \quad (\text{eq. 3.2})$$

$$\beta_1(t) = \beta_{1,1}(t) = \text{Inf}_{n_1 \in \mathbb{N}} \left\{ \begin{array}{l} n_1 w_{1,1} L_1 \\ + C_1 \left[ t - n_1 \frac{\left( w_{1,1} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} - \frac{\sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k}{C_1} \right]^+ \end{array} \right\} \quad (\text{eq. 3.3})$$

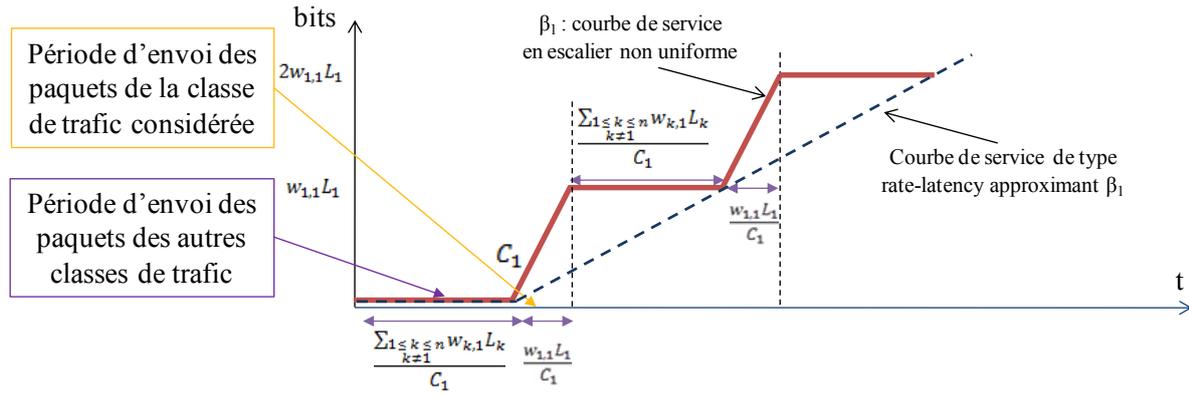


Figure 3.7. Courbe de service offerte par un ordonnanceur WRR

La courbe de service de type *rate-latency* représentée sur la figure 3.7 est celle qui est généralement utilisée. Les trois expressions de  $\beta_1$  ((eq. 3.1), (eq. 3.2) et (eq. 3.3)) correspondent à la proposition de cette thèse en termes de modélisation de la courbe de service offerte à un paquet appartenant à une classe de trafic donnée lorsque celui-ci traverse un ordonnanceur WRR. La courbe de service  $\beta_1$  exprimée sur la figure 3.7 est une fonction périodique de période  $\frac{\sum_{k=1}^{n-1} w_{k,1}L_k}{C_1}$  qui représente la période d'un cycle du WRR. Chacun de ces cycles se compose de deux périodes :

- Période d'envoi allouée aux classes de trafics autres que celle étudiée et qui est égale à  $\frac{\sum_{k=1}^{n-1} w_{k,1}L_k}{C_1}$  (la partie horizontale de la courbe de  $\beta_1$ ).
- Période d'envoi allouée à la classe de trafic étudiée et qui est égale à  $\frac{w_{1,1}L_1}{C_1}$  : la partie de la courbe de  $\beta_1$  ayant une pente de  $C_1$ .

On constate que l'équation (eq. 3.3) peut aussi s'écrire de la manière suivante :

$$\beta_1(t) = \beta_{1,1}(t) = \text{Inf}_{n_1 \in \mathbb{N}} \{K_1 + C_1[t - T_1]^+\}$$

$$\text{Avec : } \begin{cases} K_1 = n_1 w_{1,1} L_1 \\ T_1 = n_1 \left( \frac{w_{1,1} L_1 + \sum_{k=1}^{n-1} w_{k,1} L_k}{C_1} \right) + \frac{\sum_{k=1}^{n-1} w_{k,1} L_k}{C_1} \end{cases}$$

Dans la suite, on utilise cette formalisation de la courbe de service car celle-ci permet d'obtenir plusieurs propriétés intéressantes. Cette courbe de service est nommée « **courbe en escalier non uniforme** ».

Remarque :

La courbe de service donnée par l'équation (eq. 3.1) (Diouri *et al.*, 2007a) facilite la résolution des problèmes de type :

- Détermination de l'intervalle de temps qui permet d'avoir  $\beta_1(t) =$  valeur donnée ?
- Détermination simplifiée de la valeur de  $\beta_1(t)$  à un instant  $t$  donné.

D'un autre côté, la courbe de service donnée par l'équation (eq. 3.3) est plus adéquate en termes de généralité pour des opérations de produit de convolution.

Pour la courbe de service offerte à la classe de trafic étudiée quand un paquet de celle-ci traverse plusieurs ordonnanceurs WRR (commutateurs), on a le principe du Pay Bursts Only Once (PBOO) qui repose sur l'utilisation du produit de convolution min-plus (paragraphe 3.2.1).

### 3.3.4.2. Service offert au flux considéré lorsqu'il traverse $m$ ordonnanceurs WRR

Dans cette partie, nous allons démontrer par récurrence que le service offert à la classe de trafic 1 quand un paquet de celle-ci traverse  $m$  ordonnanceur(s) WRR est donné par l'expression suivante :

$$\begin{aligned} \beta_1(t) &= \otimes_{j \in \llbracket 1, m \rrbracket} \beta_{1,j}(t) = \\ &= \text{Inf}_{n_1, \dots, n_m \in \mathbb{N}} \{ \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) [t - \sum_{1 \leq l \leq m} T_l]^+ \} \quad (\text{eq. 3.4}) \\ &\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases} \end{aligned}$$

L'égalité précédente est vérifiée pour  $m = 1$ .

On suppose que l'égalité représentée par l'équation (eq. 3.4) est vérifiée jusqu'à l'ordre  $m$  et on démontre qu'elle est valable aussi pour  $m + 1$ .

La courbe de service offerte à la classe de trafic 1 quand un paquet de celle-ci traverse  $m + 1$  ordonnanceur(s) WRR est donnée par la formule suivante :

$$\begin{aligned} \beta_1(t) &= \otimes_{j \in \llbracket 1, m+1 \rrbracket} \beta_{1,j}(t) = \left( \otimes_{j \in \llbracket 1, m \rrbracket} \beta_{1,j}(t) \right) \otimes \beta_{1,m+1}(t) \\ &\text{Avec : } \beta_{1,m+1}(t) = \text{Inf}_{n_{m+1} \in \mathbb{N}} \{ K_{m+1} + C_{m+1} [t - T_{m+1}]^+ \} \\ \text{Tel que : } &\begin{cases} K_{m+1} = n_{m+1} w_{1,m+1} L_1 \\ T_{m+1} = \frac{n_{m+1} \left( w_{1,m+1} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,m+1} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,m+1} L_k}{c_{m+1}} \end{cases} \end{aligned}$$

a. Produit de convolution de  $m + 1$  courbes en escalier non uniforme :

$$\begin{aligned}
\otimes_{j \in \llbracket 1, m+1 \rrbracket} \beta_{1,j}(t) &= \left( \otimes_{j \in \llbracket 1, m \rrbracket} \beta_{1,j}(t) \right) \otimes \beta_{1,m+1}(t) = \\
&= \text{Inf}_{0 \leq s \leq t} \left\{ \text{Inf}_{n_1, \dots, n_m \in \mathbb{N}} \left\{ \left( \sum_{1 \leq l \leq m} K_l \right) + \text{Min}(C_1, \dots, C_m) \left[ t - s - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right. \\
&\quad \left. + \text{Inf}_{n_{m+1} \in \mathbb{N}} \{ K_{m+1} + C_{m+1}(s - T_{m+1})^+ \} \right\} \\
&= \text{Inf}_{0 \leq s \leq t} \left\{ \text{Inf}_{n_1, \dots, n_{m+1} \in \mathbb{N}} \left\{ \left( \sum_{1 \leq l \leq m+1} K_l \right) + \text{Min}(C_1, \dots, C_m) \left( t - s - \sum_{1 \leq l \leq m} T_l \right)^+ \right. \right. \\
&\quad \left. \left. + C_{m+1}(s - T_{m+1})^+ \right\} \right\} \\
&= \text{Inf}_{n_1, \dots, n_{m+1} \in \mathbb{N}} \left\{ \left( \sum_{1 \leq l \leq m+1} K_l \right) \right. \\
&\quad \left. + \text{Inf}_{0 \leq s \leq t} \left\{ \text{Min}(C_1, \dots, C_m) \left( t - s - \sum_{1 \leq l \leq m} T_l \right)^+ + C_{m+1}(s - T_{m+1})^+ \right\} \right\}
\end{aligned}$$

Selon (Le Boudec et Thiran, 2004), on a le résultat suivant :

$$\begin{aligned}
&\text{Inf}_{0 \leq s \leq t} \left\{ \text{Min}(C_1, \dots, C_m) \left( t - s - \sum_{1 \leq l \leq m} T_l \right)^+ + C_{m+1}(s - T_{m+1})^+ \right\} \\
&= \text{Min}(\text{Min}(C_1, \dots, C_m), C_{m+1}) \left( t - \sum_{1 \leq l \leq m+1} T_l \right)^+ \\
&= \text{Min}(C_1, \dots, C_{m+1}) \left( t - \sum_{1 \leq l \leq m+1} T_l \right)^+
\end{aligned}$$

D'où :

$$\begin{aligned}
\otimes_{j \in \llbracket 1, m+1 \rrbracket} \beta_{1,j}(t) &= \text{Inf}_{n_1, \dots, n_{m+1} \in \mathbb{N}} \left\{ \left( \sum_{1 \leq l \leq m+1} K_l \right) + \text{Min}(C_1, \dots, C_{m+1}) \left( t - \sum_{1 \leq l \leq m+1} T_l \right)^+ \right\} \\
\text{Avec : } &\begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m+1 \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq l}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq l}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m+1 \rrbracket \end{cases}
\end{aligned}$$

L'égalité représentée par l'équation (eq.3.4) est alors vérifiée pour  $m + 1$  et donc on a démontré par récurrence que celle-ci est vraie pour tout  $m \in \mathbb{N}$ .

### b. Conclusion

Le service offert à la classe de trafic 1 quand le chemin de bout en bout traversé par un paquet de celle-ci comporte  $m$  ordonnanceurs WRR est alors donné par la formule suivante :

$$\beta_1(t) = \otimes_{j \in \llbracket 1, m \rrbracket} \beta_{1,j}(t) = \text{Inf}_{n_1, \dots, n_m \in \mathbb{N}} \left\{ \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ t - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \quad (\text{eq. 3.4})$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq m \\ k \neq l}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq m \\ k \neq l}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

### 3.3.5. Calcul du délai maximal (cas de $m$ ordonnanceurs)

Dans le cas d'un paquet de la classe de trafic 1 traversant  $m$  ordonnanceurs WRR, la courbe d'arrivée de la classe de trafic 1 est toujours donnée par l'équation :  $\alpha(t) = \sigma + \rho t$  (paragraphe 3.3.3).

On a : Pour tout  $t$ ,  $d(t) \leq \sup_{s \geq 0} \text{Inf} \{ \Delta \geq 0 / \alpha(s) \leq \beta_1(s + \Delta) \}$

Dans le cas où le paquet de la classe de trafic 1 traverse  $m$  ordonnanceurs WRR, la courbe de service est donnée par la formule (eq. 3.4) du paragraphe 3.3.4.2. Celle-ci peut s'écrire de la manière suivante :

$$\beta_1(t) = \text{Inf}_{(n_1, \dots, n_m) \in \mathbb{N}^m} \{ \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(t) \} \quad (\text{eq. 3.5})$$

$$\text{Avec : } \begin{cases} \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(t) = \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) [t - \sum_{1 \leq l \leq m} T_l]^+ \\ K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq m \\ k \neq l}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq m \\ k \neq l}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

On déduit alors que :

$$d(t) \leq \sup_{s \geq 0} \text{Inf} \left\{ \Delta \geq 0 / \alpha(s) \leq \text{Inf}_{(n_1, \dots, n_m) \in \mathbb{N}^m} \{ \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(s + \Delta) \} \right\}$$

$$\text{Ainsi : } d(t) \leq \sup_{s \geq 0} \left\{ \text{Inf} \{ \Delta \geq 0 / \forall (n_1, \dots, n_m) \in \mathbb{N}^m, \sigma + \rho s \leq \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(s + \Delta) \} \right\}$$

Soient les ensembles ordonnés  $S^{(n_1, \dots, n_m) \in \mathbb{N}^m} = S_{n \in \mathbb{N}}^{(n_1, \dots, n_m) \in \mathbb{N}^m}$ , avec :

$$\forall i \in \mathbb{N} \text{ et } \forall (n_1, \dots, n_m) \in \mathbb{N}^m, s_i^{(n_1, \dots, n_m)} < s_{i+1}^{(n_1, \dots, n_m)} \text{ et } S^{(n_1, \dots, n_m)} \in \mathbb{R}^+$$

Et tel que pour tout  $(n_1, \dots, n_m) \in \mathbb{N}^m$ ,  $\forall i \in \mathbb{N}$   $\alpha(s) \leq \beta_{(n_1, \dots, n_m)} \left( s + s_i^{(n_1, \dots, n_m)} \right)$ .

Pour tout  $(n_1, \dots, n_m) \in \mathbb{N}^m$ , soit  $s_0^{(n_1, \dots, n_m)} = \text{Inf}\{S^{(n_1, \dots, n_m)}\}$ .

$$\forall (n_1, \dots, n_m) \in \mathbb{N}^m, s_0^{(n_1, \dots, n_m)} = \text{Inf}\left\{s_i^{(n_1, \dots, n_m)} / \alpha(s) \leq \beta_{(n_1, \dots, n_m)}(s + s_i^{(n_1, \dots, n_m)})\right\}$$

$$D'où : d(t) \leq \sup_{s \geq 0} \left\{ \text{Inf}\left\{ \Delta \in \bigcap_{(n_1, \dots, n_m) \in \mathbb{N}^m} S^{(n_1, \dots, n_m) \in \mathbb{N}^m} \right\} \right\} \quad (\text{eq. 3.6})$$

Etant donné que  $S^{(n_1, \dots, n_m) \in \mathbb{N}^m}$  sont des ensembles ordonnés, on pose  $S = \bigcap_{(n_1, \dots, n_m) \in \mathbb{N}^m} S^{(n_1, \dots, n_m) \in \mathbb{N}^m}$ . On a  $S$  est un ensemble ordonné.

Puisque  $\beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}$  sont des fonctions croissantes au sens large, alors pour un  $x$  et  $y$  tq  $x < y$ , on a :  $\beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(z + x) \leq \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(z + y)$ .

Et donc, si :  $\alpha(z) \leq \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(z + x)$  (Compte tenu de l'hypothèse de stabilité des réseaux :  $\lim_{x \rightarrow +\infty} \beta(x) \geq \alpha(x)$ )

Alors :  $\alpha(z) \leq \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(z + y)$

Ainsi,  $\lim_{n \rightarrow +\infty} s_n^{(n_1, \dots, n_m)} \rightarrow +\infty$ .

Et donc,  $\forall (n_1, \dots, n_m) \in \mathbb{N}^m, S^{(n_1, \dots, n_m)} \in [s_0^{(n_1, \dots, n_m)}, +\infty[$

D'où :  $S = \bigcap_{(n_1, \dots, n_m) \in \mathbb{N}^m} S^{(n_1, \dots, n_m) \in \mathbb{N}^m} \neq \emptyset$ .

De plus, l'intersection implique que :  $S = [s_0, +\infty[$  avec  $s_0 = \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \{s_0^{(n_1, \dots, n_m) \in \mathbb{N}^m}\}$

On déduit alors d'après l'équation (eq. 3.6) que :

$$d(t) \leq \sup_{s \geq 0} \{ \text{Inf}\{S\} \} = \sup_{s \geq 0} \{s_0\} = \sup_{s \geq 0} \left\{ \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \{s_0^{(n_1, \dots, n_m) \in \mathbb{N}^m}\} \right\}$$

D'où :

$$d(t) \leq \sup_{s \geq 0} \left\{ \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \left\{ \text{Inf}\left\{ s_i^{(n_1, \dots, n_m)} \geq 0 / \alpha(s) \leq \beta_{(n_1, \dots, n_m)}(s + s_i^{(n_1, \dots, n_m)}) \right\} \right\} \right\}$$

Ainsi :  $d(t) \leq \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \left\{ \sup_{s \geq 0} \left\{ \text{Inf}\left\{ \Delta \geq 0 / \alpha(s) \leq \beta_{(n_1, \dots, n_m)}(s + \Delta) \right\} \right\} \right\}$

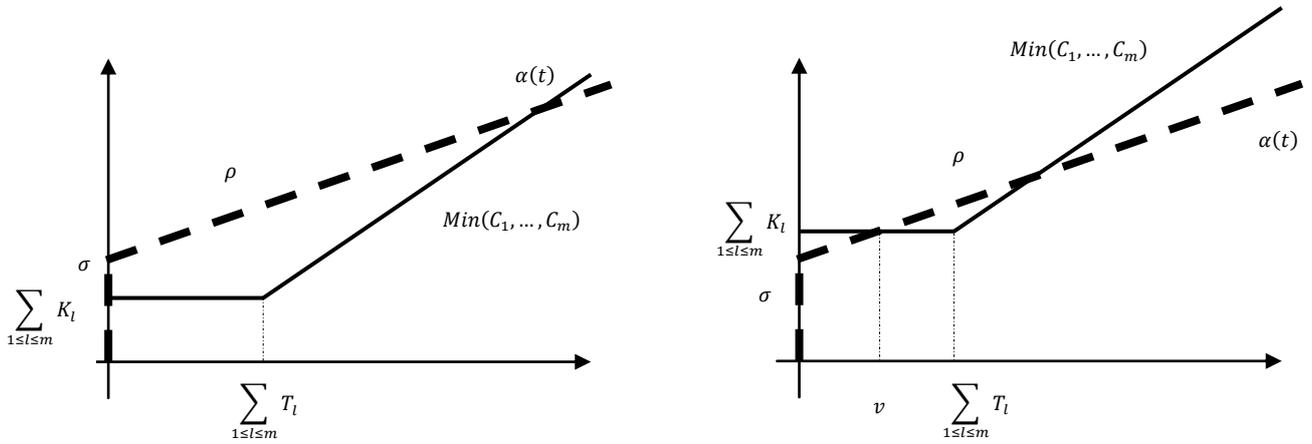
Donc :  $d(t) \leq \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \left\{ \sup_{s \geq 0} \left\{ \text{Inf}\left\{ \Delta \geq 0 / \sigma + \rho s \leq \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(s + \Delta) \right\} \right\} \right\} \quad (\text{eq. 3.7})$

$$\text{Avec : } \begin{cases} \beta_{(n_1, \dots, n_m) \in \mathbb{N}^m}(s + \Delta) = \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m)[s + \Delta - \sum_{1 \leq l \leq m} T_l]^+ \\ K_l = n_l w_{1,l} L_1 \quad \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k}{c_l} \quad \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Soit un  $m$ -uplet  $(n_1, \dots, n_m) \in \mathbb{N}^m$ , calculons :

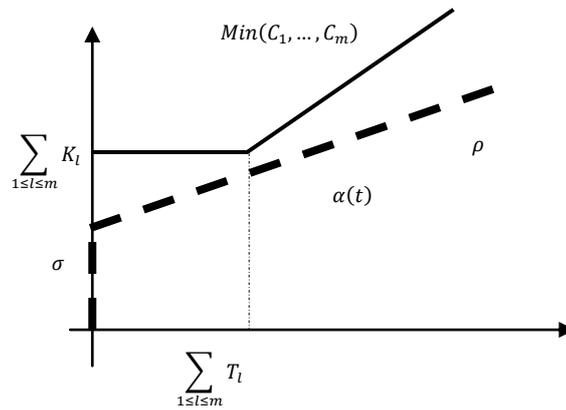
$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf}\left\{ \Delta \geq 0 / \sigma + \rho s \leq \beta_{(n_1, \dots, n_m)}(s + \Delta) \right\} \right\}$$

Pour le calcul du  $D_{(n_1, \dots, n_m)}$  pour un  $(n_1, \dots, n_m) \in \mathbb{N}^m$ , on distingue entre 3 cas :



a. Cas 1 :  $\sum_{1 \leq l \leq m} K_l \leq \sigma$

b. Cas 2 :  $\sigma < \sum_{1 \leq l \leq m} K_l \leq \sigma + \rho \sum_{1 \leq l \leq m} T_l$



c. Cas 3 :  $\sum_{1 \leq l \leq m} K_l > \sigma + \rho \sum_{1 \leq l \leq m} T_l$

Figure 3.8. Les 3 cas à distinguer dans le calcul du délai

Cas 1 : Si :  $\sum_{1 \leq l \leq m} K_l \leq \sigma$

Donc dans ce cas :

$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right\}$$

$$= \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s - \sum_{1 \leq l \leq m} K_l \leq \text{Min}(C_1, \dots, C_m) [s + \Delta - \sum_{1 \leq l \leq m} T_l]^+ \right\} \right\}$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l (w_{1,l} L_1 + \sum_{\substack{k=1 \\ k \neq l}}^{m} w_{k,l} L_k) + \sum_{\substack{k=1 \\ k \neq l}}^{m} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Selon (Le Boudec et Thiran, 2004), le délai maximal subi par un flux dont la courbe d'arrivée est  $\alpha(t) = \sigma + \rho t$  quand celui-ci traverse un système ayant une courbe d'arrivée de type *rate-latency*  $\beta(t) = R(t - T)^+$  est donné par :  $T + \frac{\sigma}{R}$ .

Comme dans ce cas :  $\sum_{1 \leq l \leq m} K_l \leq \sigma$ , donc :  $\sigma - \sum_{1 \leq l \leq m} K_l \geq 0$ .

Donc :

$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s - \sum_{1 \leq l \leq m} K_l \leq \text{Min}(C_1, \dots, C_m) [s + \Delta - \sum_{1 \leq l \leq m} T_l]^+ \right\} \right\}$$

$$= \sum_{1 \leq l \leq m} T_l + \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)}$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Cas 2 : Si  $\sigma < \sum_{1 \leq l \leq m} K_l \leq \sigma + \rho \sum_{1 \leq l \leq m} T_l$

$$\text{Soit : } v = \frac{(\sum_{1 \leq l \leq m} K_l) - \sigma}{\rho}$$

D'où :  $0 \leq v \leq \sum_{1 \leq l \leq m} T_l$

Dans ce cas :

$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right\}$$

$$= \sup_{0 \leq s < v} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) [s + \Delta - \sum_{1 \leq l \leq m} T_l]^+ \right\} \right\}$$

$$\vee \sup_{s \geq v} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) [s + \Delta - \sum_{1 \leq l \leq m} T_l]^+ \right\} \right\}$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

D'après la figure 3.8, quand  $0 \leq s < v$  alors  $\beta(s) \geq \alpha(s)$ , ainsi :

$$D_{(n_1, \dots, n_m)} = 0 \vee \sup_{s \geq v} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right\}$$

$$\text{Avec : } s + \Delta - \sum_{1 \leq l \leq m} T_l \geq 0$$

Car si :  $s + \Delta - \sum_{1 \leq l \leq m} T_l < 0$  alors :

$$\sup_{s \geq v} \left\{ \text{Inf} \left\{ \Delta \geq 0/\sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right\} = 0$$

Et donc :

$$D_{(n_1, \dots, n_m)} = 0 \vee \sup_{s \geq v} \left\{ \text{Inf} \left\{ \Delta \geq 0/\sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right] \right\} \right\}$$

$$\text{Avec : } s + \Delta - \sum_{1 \leq l \leq m} T_l \geq 0$$

$$\text{D'où : } D_{(n_1, \dots, n_m)} = 0 \vee \sup_{s \geq v} \left\{ \text{Inf} \left\{ \Delta \geq 0/\frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)} + \sum_{1 \leq l \leq m} T_l + s \left( \frac{\rho - \text{Min}(C_1, \dots, C_m)}{\text{Min}(C_1, \dots, C_m)} \right) \leq \Delta \right\} \right\}$$

$$\text{Avec : } s + \Delta - \sum_{1 \leq l \leq m} T_l \geq 0$$

$$\text{Or puisque : } \sigma \leq \sum_{1 \leq l \leq m} K_l \leq \sigma + \rho \sum_{1 \leq l \leq m} T_l, \text{ d'où : } \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} + \sum_{1 \leq l \leq m} T_l \geq 0$$

$$\text{Et puisque : } \rho \leq \text{Min}(C_1, \dots, C_m) \text{ et } \sigma - \sum_{1 \leq l \leq m} K_l \leq 0$$

$$\text{D'où : } \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} \leq \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)}$$

$$\text{Ainsi : } \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)} + \sum_{1 \leq l \leq m} T_l \geq 0$$

Ainsi dans ce cas, le supremum de  $D_{(n_1, \dots, n_m)}$  est atteint à  $s = v$ . Et on a :

$$\begin{aligned} D_{(n_1, \dots, n_m)} &= \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)} + \sum_{1 \leq l \leq m} T_l - \frac{(\sum_{1 \leq l \leq m} K_l) - \sigma}{\rho} \left( \frac{\text{Min}(C_1, \dots, C_m) - \rho}{\text{Min}(C_1, \dots, C_m)} \right) \\ &= \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} + \sum_{1 \leq l \leq m} T_l \end{aligned}$$

$$\text{Avec : } s + \Delta - \sum_{1 \leq l \leq m} T_l \geq 0$$

La condition  $s + \Delta - \sum_{1 \leq l \leq m} T_l \geq 0$  est bien vérifiée dans ce cas.

Ainsi, dans ce cas :

$$D_{(n_1, \dots, n_m)} = \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} + \sum_{1 \leq l \leq m} T_l$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{k=1 \\ k \neq l}}^n w_{k,l} L_k \right) + \sum_{\substack{k=1 \\ k \neq l}}^n w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Cas 3 :  $\sum_{1 \leq l \leq m} K_l > \sigma + \rho \sum_{1 \leq l \leq m} T_l$

$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \sum_{1 \leq l \leq m} K_l + \text{Min}(C_1, \dots, C_m) \left[ s + \Delta - \sum_{1 \leq l \leq m} T_l \right]^+ \right\} \right\}$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Dans ce cas, le délai est nul d'après la figure 3.8.

### Conclusion

Soit un  $m$ -uplet  $(n_1, \dots, n_m) \in \mathbb{N}^m$ , on a :

$$D_{(n_1, \dots, n_m)} = \sup_{s \geq 0} \left\{ \text{Inf} \left\{ \Delta \geq 0 / \sigma + \rho s \leq \beta_{(n_1, \dots, n_m)}(s + \Delta) \right\} \right\}$$

$$= \begin{cases} \sum_{1 \leq l \leq m} T_l + \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)} & \text{si } \sum_{1 \leq l \leq m} K_l \leq \sigma \\ \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} + \sum_{1 \leq l \leq m} T_l & \text{si } \sigma < \sum_{1 \leq l \leq m} K_l \leq \sigma + \rho \sum_{1 \leq l \leq m} T_l \\ 0 & \text{si } \sum_{1 \leq l \leq m} K_l > \sigma + \rho \sum_{1 \leq l \leq m} T_l \end{cases}$$

$$\text{Avec : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Et on a aussi :  $d(t) \leq \sup_{(n_1, \dots, n_m) \in \mathbb{N}^m} \{ D_{(n_1, \dots, n_m)} \}$

$$\text{Avec : } D_{(n_1, \dots, n_m)} = \begin{cases} \sum_{1 \leq l \leq m} T_l + \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\text{Min}(C_1, \dots, C_m)} & \text{si } \sum_{1 \leq l \leq m} K_l \leq \sigma \\ \frac{\sigma - \sum_{1 \leq l \leq m} K_l}{\rho} + \sum_{1 \leq l \leq m} T_l & \text{si } \sigma < \sum_{1 \leq l \leq m} K_l \leq \sigma + \rho \sum_{1 \leq l \leq m} T_l \\ 0 & \text{si } \sum_{1 \leq l \leq m} K_l > \sigma + \rho \sum_{1 \leq l \leq m} T_l \end{cases}$$

$$\text{Et : } \begin{cases} K_l = n_l w_{1,l} L_1 & \text{Avec } l \in \llbracket 1, m \rrbracket \\ T_l = \frac{n_l \left( w_{1,l} L_1 + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k \right) + \sum_{\substack{1 \leq k \leq m \\ k \neq 1}} n w_{k,l} L_k}{c_l} & \text{Avec } l \in \llbracket 1, m \rrbracket \end{cases}$$

Donc on a calculé le délai de traversée de  $m$  ordonnanceurs WRR.

### 3.3.6. Problématique de l'agrégation de flux

La figure 3.9 montre qu'il y a une concurrence entre le flux 13 et le flux 23 au niveau du port de sortie du commutateur 1 (ces deux flux appartiennent à la même classe de service). Cette figure met bien l'accent sur la problématique de l'agrégation de flux.

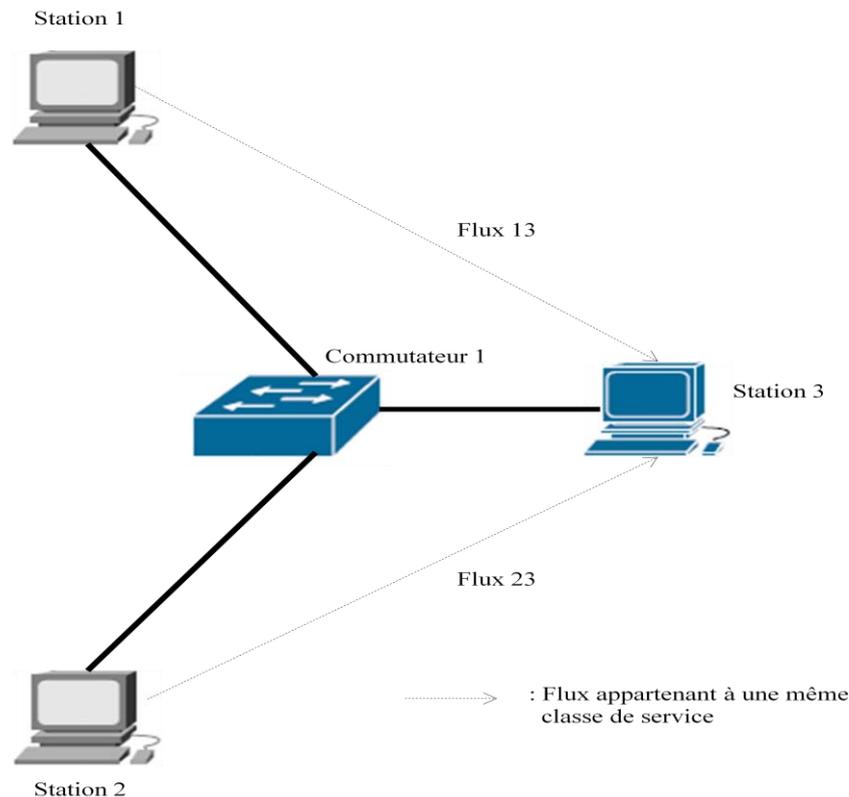


Figure 3.9. Phénomène d'agrégation de flux

Les travaux qui ont été menés jusqu'à ce stade ont considéré qu'un seul flux est présent par classe de service. Néanmoins, les topologies réseau peuvent conduire à ce que plusieurs flux appartenant à la même classe de service entrent en concurrence en ce qui concerne l'accès à un port de sortie d'un commutateur. En effet, il est nécessaire dans ce cas que soit la courbe de service soit exprimée pour un flux donné et non pour l'intégralité des flux (tous les flux appartenant à la classe de service) ou alors que l'on suppose que le délai maximal sera identique pour tous les flux en ramenant le problème à un seul super flux qui correspondrait à la somme de ces flux. Il est facile de constater que cette deuxième solution peut engendrer un pessimisme sur l'expression du majorant du délai de bout en bout.

La courbe de service proposée dans cette thèse s'inscrit dans le cadre de cette première approche. En effet, elle permet d'éliminer les problèmes de concurrence entre des flux n'appartenant pas à la même classe de service puisque elle est établie pour une classe de service. Cette qualité permet donc de limiter l'apparition de ce phénomène. Néanmoins, reste le problème de la concurrence entre les flux associés à la même classe de service (figure 3.9). Pour l'étude de réseaux où se présentera ce type de situations, nous suggérons l'utilisation du Pay Multiplexing Only Once (PMOO) (Schmitt et Zdarsky, 2006) comme défini ci-dessous. On notera également que cette problématique de la prise en considération de la concurrence entre différents flux reste actuellement ouverte dans le cas de réseaux fortement cycliques.

La première idée lorsqu'on parle de trafics concurrents appartenant à la même classe de service consiste à recourir au service résiduel (Schmitt et Zdarsky, 2006).

Théorème (*Blind Multiplexing Nodal Service Curves*) (Schmitt et Zdarsky, 2006): Considérons un nœud servant deux flux, 1 et 2, avec un multiplexage aveugle entre les deux flux. Supposons que le nœud garantit une courbe de service minimale stricte  $\beta$  et une courbe de service maximale  $\bar{\beta}$  pour l'agrégation des deux flux. Supposons que le flux 2 est contraint par la courbe d'arrivée  $\alpha_2$ . Alors :  $\beta_1(t) = [\beta - \alpha_2]^+$  est une courbe de service pour le flux 1 si  $\beta_1$  est croissante au sens large.  $\bar{\beta}$  reste aussi la courbe de service maximale pour le flux 1.

En se basant sur un multiplexage FIFO et en considérant deux flux concurrents, (Le Boudec et Thiran, 2004) ont utilisé les courbes d'arrivée de type seuil percé et une courbe de service de type *rate-latency* et ils ont proposé le corollaire suivant.

Corollaire (Augmentation de la rafale due au FIFO) (Le Boudec et Thiran, 2004) : Considérons un nœud servant deux flux, 1 et 2, selon un ordre FIFO. Supposons que le flux  $i$  est contraint par un seuil percé avec un taux  $\rho_i$  et une avalanche  $\sigma_i$ . Supposons que le nœud garantit à l'agrégation des deux flux une courbe de service de type *rate-latency*  $\beta_{R,T}$ . Si  $\rho_1 + \rho_2 < R$ , alors le flux 1 a une courbe de service égale à la fonction *rate-latency* avec un taux  $R - \rho_2$  et une latence  $T + \frac{\sigma_2}{R}$  et à la sortie, le flux 1 est contraint par un seuil percé avec un taux  $\rho_1$  et une avalanche  $b_1^*$  avec :  $b_1^* = \sigma_1 + \rho_1(T + \frac{\sigma_2}{R})$ .

De la même manière que dans le cas d'une courbe d'arrivée affine et d'une courbe de service de type *rate-latency*, le résultat du service résiduel d'une courbe d'arrivée affine et d'une courbe de service en escalier non uniforme correspondra à une courbe de service en escalier non uniforme. On pourra ainsi utiliser ce résultat en vue de l'appliquer à la courbe de service proposée dans cette thèse dans le cas où plusieurs flux de même classe seraient en concurrence en ce qui concerne l'accès à un même port de sortie.

### Le PMOO dans le cas du service résiduel

Le phénomène du PMOO a été initialement identifié par (Schmitt et Zdarsky, 2006).

Soit un flux 1 contraint par la courbe d'arrivée  $\alpha_1$  et un flux 2 contraint par la courbe d'arrivée  $\alpha_2$ . Ces deux flux traversent un premier nœud offrant une courbe de service stricte  $\beta_1$  puis un second nœud offrant une courbe de service  $\beta_2$ . On peut dans cet exemple, et suivant l'hypothèse du *blind multiplexing*, caractériser de deux manières le service de bout en bout offert au flux 1.

La première manière consiste à utiliser dans un premier temps le théorème du *Blind Multiplexing Nodal Service Curves* (Schmitt et Zdarsky, 2006), ce qui nous donne :

- Le service offert au flux 1 par le premier nœud est :  $(\beta_1 - \alpha_2)^+$ .
- La courbe d'arrivée du flux 2 pour le second serveur est quant à elle donnée par :  $\alpha_2 \oslash (\beta_1 - \alpha_1)^+$ .
- Le service offert par le deuxième nœud au flux 1 est :  $(\beta_2 - \alpha_2 \oslash (\beta_1 - \alpha_1)^+)^+$ .

Puis ensuite, on utilise le théorème de la concaténation des nœuds (Le Boudec et Thiran, 2004), ce qui nous donne :

La courbe de service globale offerte au flux 1 est :

$$\beta = (\beta_1 - \alpha_2)^+ \otimes (\beta_2 - \alpha_2 \oslash (\beta_1 - \alpha_1)^+)^+$$

La deuxième manière consiste à utiliser dans un premier temps le théorème de la concaténation des nœuds (Le Boudec et Thiran, 2004), puis ensuite le théorème du *Blind Multiplexing Nodal Service Curves* (Schmitt et Zdarsky, 2006), ce qui nous donne :

$$\beta = ((\beta_1 \otimes \beta_2) - \alpha_2)^+$$

(Fidler et Sander, 2004) ont appliqué le principe du PMOO dans le cas d'un multiplexage FIFO en se basant sur le service résiduel (Le Boudec et Thiran, 2004) et cela pour des flux agrégés ayant des courbes d'arrivée de type sceau percé et traversant des nœuds offrant des courbes de services de types *rate-latency*. Par ailleurs, dans (Fidler et Sander, 2004), un contrôle d'admission avancé basé sur les « *Bandwidth Brokers* » a été présenté dans le but de fournir des garanties strictes de QoS et cela en se basant sur une architecture à services différenciés. Dans (Fidler et Sander, 2004), le principe du *Pay Bursts Only Once* (Le Boudec et Thiran, 2004) a été étendu à un ordonnancement agrégé « *aggregate based scheduling* » pour établir un ensemble de formules permettant de calculer des bornes de délais strictes raisonnables.

Ensuite (Bouillard *et al.*, 2008) (Schmitt et Zdarsky, 2006) ont développé la problématique du PMOO dans le cas général. (Schmitt et Zdarsky, 2006) ont alors considéré l'exemple suivant (figure 3.10).

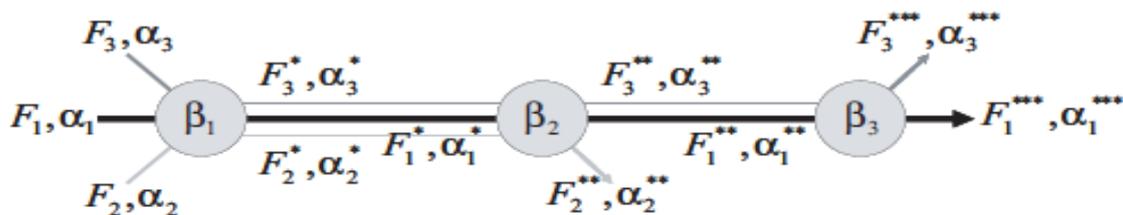


Figure 3.10. Scénario de flux emboîtés interférents (Schmitt et Zdarsky, 2006)

Dans cet exemple (Schmitt et Zdarsky, 2006) présentent une analyse du flux 1. Ils ont noté qu'une première courbe de service de bout en bout pour le flux 1 serait déterminée de la manière suivante :

$$\beta_1^{SF} = [\beta_1 - \alpha_2 - \alpha_3]^+ \otimes [\beta_2 - \alpha_2^* - \alpha_3^*]^+ \otimes [\beta_3 - \alpha_3^{**}]^+$$

(Schmitt et Zdarsky, 2006) ont proposé une autre façon d'analyser le flux 1 représenté sur la figure 3.10 qui consiste à concaténer le nœud 1 et 2, de soustraire le flux 2 et ainsi d'obtenir la courbe de service pour le flux 1 et 3 ensemble, de concaténer cela avec le nœud 3 et de soustraire le flux 3. Ceci leur a permis d'obtenir la courbe de service suivante :

$$\beta_1^{PS} = [[(\beta_1 \otimes \beta_2) - \alpha_2]^+ \otimes \beta_3 - \alpha_3]^+$$

D'un autre côté, (Bouillard *et al.*, 2008) ont montré un exemple où il est impossible de dire si une courbe de service obtenue de manière classique est meilleure que celle obtenue en utilisant le principe du PMOO.

On pourra ainsi utiliser le théorème relatif au *blind multiplexing* (Schmitt et Zdarsky, 2006) en vue de l'appliquer à la courbe de service proposée dans cette thèse dans le cas où plusieurs flux de la même classe seraient en concurrence en ce qui concerne l'accès à un même port de sortie.

Cette stratégie pourra être enrichie par l'utilisation de la méthodologie Least Upper Delay Bound (LUBD) (figure 3.11) proposée par (Lenzini *et al.*, 2008) ainsi que les opérateurs de PMOO définis par (Bouillard *et al.*, 2008) (Schmitt et Zdarsky, 2006).

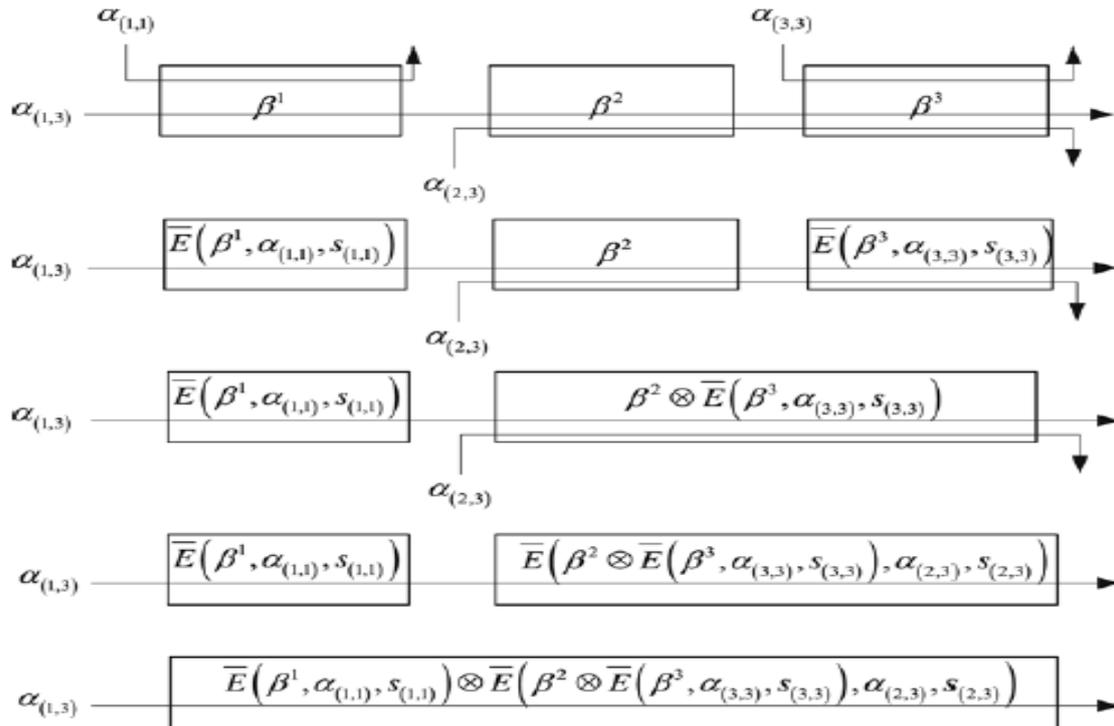


Figure 3.11. Un exemple d'application de la méthodologie LUDB (Lenzini *et al.*, 2008)

Pour la figure 3.11, (Lenzini *et al.*, 2008) se sont basés sur le corollaire 2.7 de (Lenzini *et al.*, 2008) afin de montrer leur approche.

Au final, la problématique soulevée par le PMOO concerne le choix de la situation où il faut mettre en œuvre le service résiduel et le PBOO.

### 3.4. Comparaison des résultats du délai avec ceux obtenus avec une *rate-latency*

#### 3.4.1. Courbe de service de type *rate-latency* offerte à un ordonnanceur WRR

Pour cela, on se base sur une courbe de service de type *rate-latency* qui approche la courbe de service étudiée précédemment. Le but du paragraphe 3.4 est de montrer que la courbe de service étudiée précédemment dans ce chapitre permet de réduire le pessimisme par rapport à une courbe de service de type *rate-latency* (paragraphe 3.4.3).

$$\beta_{1,1}^{rate\_latency}(t) = R_1(t - T_1)^+ \quad (eq. 3.8)$$

$$\text{Avec : } T_1 = \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} \quad \text{et} \quad R_1 = \frac{w_{1,1} L_1}{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1} C_1$$

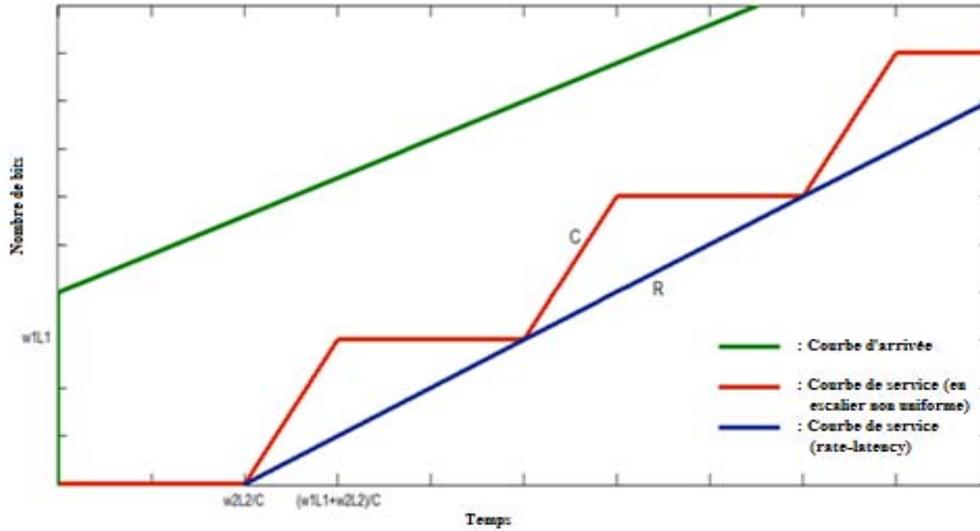


Figure 3.12. Courbes de services de type *rate-latency* et de type en escalier non uniforme (cas où  $n = 2$ )

D'après la figure 3.12, qui exprime les courbes de services de type *rate-latency* et de type escalier non uniforme, l'égalité est obtenue quand :

$$t = \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} + l \left( \left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1 \right) \text{ avec } l \in \mathbb{N}.$$

On en déduit que :

$$\begin{aligned} \beta_{1,1}^{rate\_latency} & \left( \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} + l \left( \left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1 \right) \right) \\ & = \beta_{1,1} \left( \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} + l \left( \left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1 \right) \right) \quad (\forall l \in \mathbb{N}) \end{aligned}$$

### 3.4.2. Courbe de service de type *rate-latency* offerte à une série d'ordonnanceurs WRR

D'après le paragraphe 1.4.3 de (Le Boudec et Thiran, 2004), il a été énoncé que si l'on a deux nœuds offrant chacun une courbe de service de type *rate-latency*  $\beta_{R_i, T_i}$  ( $i = 1, 2$ ) alors un simple calcul donne ce qui suit :

$$\beta_{R_1, T_1} \otimes \beta_{R_2, T_2} = \beta_{\min(R_1, R_2), T_1 + T_2}$$

Ainsi, à partir de cela, on déduit que la courbe de service de type *rate-latency*  $\beta_1^{rate\_latency}$  offerte par une série de  $m$  ordonnanceurs WRR est donnée par la formule suivante :

$$\beta_1^{rate\_latency}(t) = \beta_{1,1}^{rate\_latency} \otimes \dots \otimes \beta_{1,m}^{rate\_latency}$$

Ainsi :

$$\beta_1^{rate\_latency}(t) = \beta_{\min(R_1, \dots, R_m), T_1 + \dots + T_m}$$

$$\text{Avec : } T_i = \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq i}} w_{k,i} L_k \right)}{C_i} \text{ et } R_i = \frac{w_{1,i} L_1}{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq i}} w_{k,i} L_k \right) + w_{1,i} L_1} C_i$$

### 3.4.3. Pessimisme de la courbe de service de type *rate-latency*

Le but ici est d'illustrer *via* un exemple que la courbe de service étudiée précédemment dans ce chapitre permet de réduire le pessimisme par rapport à une courbe de service de type *rate-latency*.

Exemple :

On considère un flux temps-réel (classe de service 1 avec un poids  $w_1$  et dont la taille des paquets est  $L_1$ ) dont la courbe d'arrivée est donnée par l'expression suivante :

$$\alpha(t) = \sigma + \rho t$$

On considère un flux de fond (classe de service 2 avec un poids  $w_2$  et dont la taille des paquets correspond à la taille maximale d'une trame Ethernet soit  $L_2 = L_{max} = 1526 \text{ octets}$ ), celui-ci perturbe le flux 1.

Le cas qui montre le pessimisme maximal de la courbe *rate-latency* est obtenu lorsque :

$$w_1 L_1 - \sigma = 1 \text{ bit}$$

Soit :

$$\sigma = 799 \text{ bits} ; w_1 = 1 ; L_1 = 100 \text{ octets} = 800 \text{ bits},$$

$$w_2 = 1 ; L_2 = L_{max} = 1526 \text{ octets} ; C = 10 \text{ Mbit/s}$$

$$\beta_{1,1}^{rate\_latency}(t) = R_1(t - T_1)^+ \tag{eq. 3.8}$$

$$\text{Avec : } T_1 = \frac{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right)}{C_1} \text{ et } R_1 = \frac{w_{1,1} L_1}{\left( \sum_{\substack{1 \leq k \leq n \\ k \neq 1}} w_{k,1} L_k \right) + w_{1,1} L_1} C_1$$

D'après l'équation (eq. 3.8), la courbe de service de type *rate-latency* offerte au flux temps-réel est donnée par :

$$\beta_1^{\text{rate-latency}}(t) = R_1(t - T_1)^+$$

$$\text{Avec : } T_1 = \frac{w_2 L_2}{C} \text{ et } R_1 = \frac{w_1 L_1}{w_2 L_2 + w_1 L_1} C$$

Selon (Le Boudec et Thiran, 2004), dans ce cas le délai maximal obtenu avec la courbe de service *rate-latency*  $D_{\max}^{\text{rate-latency}}$  est :

$$D_{\max}^{\text{rate-latency}} = \frac{\sigma}{R_1} + \frac{w_2 L_2}{C}$$

$$D_{\max}^{\text{rate-latency}} = \frac{799}{800 * 10^7} \times (1526 \times 8 + 800) + \frac{1526 \times 8}{10^7}$$

$$D_{\max}^{\text{rate-latency}} = 2519,97 \mu s$$

Le délai maximal obtenu avec la courbe de service en escalier non uniforme  $D_{\max}^{\text{escalier non uniforme}}$  est :

$$D_{\max}^{\text{escalier non uniforme}} = 1300,7 \mu s$$

**Le pessimisme engendré par la courbe de service de type *rate-latency* est de  $1219,27 \mu s \approx 1220,8 \mu s = \frac{w_2 L_2}{C}$ . De manière plus significative, dans ce cas :**

$$\frac{D_{\max}^{\text{rate-latency}}}{D_{\max}^{\text{escalier non uniforme}}} = \frac{2519,97 \mu s}{1300,7 \mu s} = 193,74 \%$$

Le résultat de cet exemple montre que la *rate-latency*, qui est largement utilisée jusqu'ici, engendre une sous estimation du service offert. On peut alors mettre au crédit de la courbe proposée dans cette thèse une réduction du majorant de près de la moitié de sa valeur pour cet exemple.

### 3.4.4. Conclusions

Lorsqu'on compare la courbe de service proposée pour le cas d'un ordonnanceur WRR (eq. 3.3) avec la courbe de service de type *rate-latency* vue dans le paragraphe 3.4.1, il est clair que cette dernière est moins précise et que la courbe de service proposée (eq. 3.3)

permet de réduire le pessimisme. Aussi, une simple comparaison entre ces deux courbes de services nous permet de remarquer que les bornes dans le cas des courbes de service de type *rate-latency* ne sont pas toujours atteignables (paragraphe 3.4.3).

Il est aussi à noter qu'il est possible d'obtenir de meilleurs résultats en termes de bornes du délai (pour les deux types de courbes de service) et cela en utilisant la courbe d'arrivée suivante :  $\alpha_1(t) = \min(Ct, \sigma + \rho t)$ . Dans le cas de  $\alpha_1$ , l'arrivée des données est contrainte par la capacité (débit du lien entrant que l'on note ici  $C$  en bit/s). La prise en compte de cette contrainte évite de considérer un surplus de trafic que l'on ne retrouve pas en réalité d'où le gain sur l'éventuel pessimisme du majorant du délai.

Dans le paragraphe suivant (paragraphe 3.5), nous allons comparer les délais maxima obtenus en utilisant le calcul réseau (dans le cas d'une courbe de service en escalier non uniforme et dans le cas d'une courbe de service de type *rate-latency*) avec des résultats de simulations obtenus avec le logiciel OPNET.

### 3.5. Comparaison entre les délais maxima obtenus en utilisant le calcul réseau et ceux obtenus en simulation

Dans cette comparaison, on utilise OPNET (OPNET, 2005) pour simuler nos scénarii. Notre étude est basée sur l'architecture représentée dans la figure 3.13.

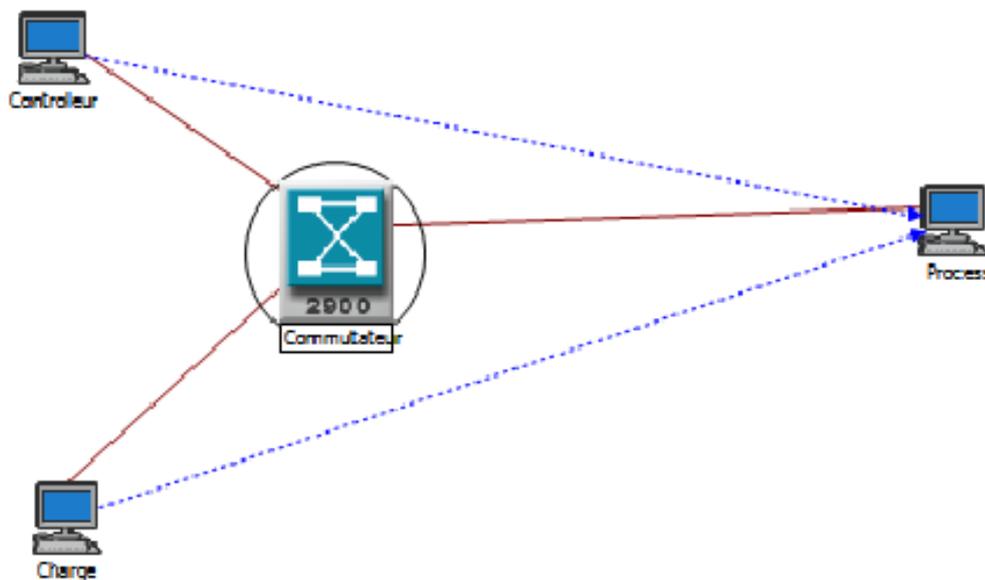


Figure 3.13. Architecture considérée pour la comparaison

Comme le montre la figure 3.13, l'architecture est composée de deux stations à l'origine du trafic temps-réel et une troisième station qui génère le trafic de fond. La capacité des liens est de 10 Mb/s. Le trafic temps-réel se compose des trafics échangés entre la station

« Contrôleur » et la station « Process ». La station « Contrôleur » envoie un paquet de 125 octets toutes les 10 ms à la station « Process ». Le trafic de fond est un flux envoyé depuis la station « Charge » vers la station « Process ». La station « Charge » comme son nom l'indique surcharge le réseau en envoyant des messages avec une période variant suivant une loi exponentielle autour de 1,7 ms et dont la taille varie aussi suivant une loi exponentielle autour de 1500 octets. Ce qui décrit en termes de charge une loi exponentielle autour de 7 Mb/s.

Dans le but d'implémenter la politique d'ordonnancement WRR, nous avons affecté à chaque type de flux un poids  $w_1$  (respectivement  $w_2$ ) qui représente le poids du trafic temps-réel (respectivement le poids du trafic de fond). Pour notre comparaison, nous avons considéré que  $w_1 = 2$  et  $w_2 = 1$ . Nous avons fixé la durée de simulation à 5 minutes et elle a été lancée 30 fois avec des tirages de variables aléatoires (« *seed* ») différents. Le délai maximal obtenu pour les trente simulations effectuées est de : 1230,4  $\mu s$ . Le calcul réseau basé sur la courbe de service en escalier non uniforme donne : 1320,8  $\mu s$ . Le calcul réseau basé sur la courbe de service de type *rate-latency*, quant à lui, donne : 1931,2  $\mu s$ .

Ainsi pour cet exemple, l'écart entre le délai estimé par calcul réseau utilisant une courbe de service de type *rate-latency* et le délai maximum obtenu par simulation est de 56% et n'est plus que de 7% avec une courbe de service en escalier non uniforme.

Ainsi, pour les applications fortement contraintes temporellement, ce qui a été développé en termes de calcul réseau dans ce chapitre (courbe de service en escalier non uniforme) peut être appliqué avec un niveau de pessimisme acceptable.

Il est à noter que les messages dont les délais sont proches des maxima estimés par calcul réseau apparaissent rarement. En effet, pour l'une des trente simulations où le délai de 1230,4  $\mu s$  a été atteint : aucune valeur n'est supérieure à 95 % du délai maximal calculé (1320,8  $\mu s$ ) et seulement 210 sur les 13966 valeurs de délai (soit à peu près 1,5 % des valeurs) obtenues sont supérieures à 90 % du délai maximal calculé. Le pessimisme porte donc plus sur la fréquence de l'apparition de ces maxima qui va avoir pour conséquence une surprotection plus que nécessaire dans la majeure partie du temps des messages temps-réel lorsque l'on va configurer les poids de l'ordonnanceur WRR.

### 3.6. Recherche des poids optimaux

Au chapitre précédent, nous avons montré que si le WRR peut contrôler le retard, il n'existe pas de méthode de détermination des poids pour un contexte SCR.

L'algorithme que nous allons alors proposer permet de déterminer la configuration des poids du WRR qui représentent en fait la bande passante allouée pour chaque type de trafic. Il s'agit

de trouver une configuration optimale des poids pour garantir une contrainte de délai pour le trafic temps-réel strict tout en offrant un maximum de ressources pour les trafics annexes.

L'algorithme qui suit s'applique au cas où on a  $m$  ordonnanceurs et  $n$  classes de trafic traversant ces ordonnanceurs. Pour des raisons de simplicité, nous allons considérer que le trafic temps-réel correspond à la première classe de trafic (classe de trafic n° 1).

### Remarque

On rappelle que dans le cas réel (c'est-à-dire sur les commutateurs existant sur le marché) la valeur maximale de  $n$  est de 4.

L'algorithme de calcul des poids optimaux nous permet d'identifier les poids optimaux  $\left( (w_{1,1_{op}}, \dots, w_{n,1_{op}}), \dots, (w_{1,m_{op}}, \dots, w_{n,m_{op}}) \right)$  tels que :

$$\left( (w_{1,1_{op}}, \dots, w_{n,1_{op}}), \dots, (w_{1,m_{op}}, \dots, w_{n,m_{op}}) \right) = \left\{ \left( (w_{1,1}, \dots, w_{n,1}), \dots, (w_{1,m}, \dots, w_{n,m}) \right) \right. \\ \left. / \left\{ \begin{array}{l} \frac{\sum_{i=1}^m \sum_{k=2}^n w_{k,i_{op}} L_k}{\sum_{i=1}^m (w_{1,i_{op}} L_1 + \sum_{k=2}^n w_{k,i_{op}} L_k)} \geq \frac{\sum_{i=1}^m \sum_{k=2}^n w_{k,i} L_k}{\sum_{i=1}^m (w_{1,i} L_1 + \sum_{k=2}^n w_{k,i} L_k)} \\ \forall \left( (w_{1,1}, \dots, w_{n,1}), \dots, (w_{1,m}, \dots, w_{n,m}) \right) \in \llbracket 1, 255 \rrbracket^{n \times m} \\ \text{et seuil strict de l'application} \geq \bar{D} \left( (w_{1,1}, \dots, w_{n,1}), \dots, (w_{1,m}, \dots, w_{n,m}) \right) \end{array} \right\} \right)$$

Ainsi, l'algorithme permettant d'identifier les poids optimaux est le suivant :

```

/*****Calcul des poids optimaux*****/
/* Procédure pour parcourir toutes les combinaisons des poids pour obtenir les poids optimaux maximisant les ressources pour le trafic de fond */
Procédure calcul_des_poids_optimaux (i,j)
Début
  Pour  $w_{i,j}$  allant de 1 à 255 Faire
     $i \leftarrow i + 1$ ;
    Si ( $i \leq n$ ) Alors
      calcul_des_poids_optimaux (i,j);
    Sinon
       $i \leftarrow 1$ ;
       $j \leftarrow j + 1$ ;
      Si ( $j \leq m$ ) Alors
        calcul_des_poids_optimaux (i,j);
      Sinon
        /* Ici on atteint la nouvelle combinaison des poids à tester */
        /* Ici on calcule le délai maximal ( $D_{max}$ ) pour cette nouvelle combinaison des poids*/
        calcul_  $D_{max}$  ;
        Si ( $D_{max} \leq \text{Seuil}$ ) Alors
          Si ( $\frac{\sum_{b=1}^m \sum_{a=2}^n w_{a,b} L_a}{\sum_{b=1}^m (w_{1,b} L_1 + \sum_{a=2}^n w_{a,b} L_a)} > TB_{pmax}$ )
            Pour  $k$  allant de 1 à  $n$  Faire
              Pour  $l$  allant de 1 à  $m$  Faire
                 $w_{op_{k,l}} \leftarrow w_{k,l}$  ;
              Fin Pour
            Fin Pour
             $TB_{pmax} \leftarrow \frac{\sum_{b=1}^m \sum_{a=2}^n w_{a,b} L_a}{\sum_{b=1}^m (w_{1,b} L_1 + \sum_{a=2}^n w_{a,b} L_a)}$  ;
          Fin Si
        Fin Si
      Fin Si
    Fin Pour
Fin Procédure
*/ Programme principal */
Début
 $L \leftarrow \text{Taille\_du\_paquet\_du\_trafic\_temps\_réel}$  ; /*on considère que celui-ci correspond à  $i = 1$ */
 $\rho \leftarrow \text{Taux\_arrivée\_maximal\_du\_trafic\_temps\_réel}$  ;  $n \leftarrow \text{Nombre\_de\_classes\_de\_trafic\_des\_ordonnances\_WRR}$  ;
 $m \leftarrow \text{Nombre\_ordonnances\_WRR\_traversés\_par\_le\_flux\_étudié}$  ;  $\sigma \leftarrow \text{Rafale\_maximale}$  ;
 $\text{Seuil} \leftarrow \text{Seuil\_de\_délai\_de\_bout\_en\_bout\_à\_ne\_pas\_dépasser\_pour\_le\_trafic\_temps\_réel}$  ;
/* Initialisation des capacités des commutateurs*/
Pour  $i$  allant de 1 à  $n$  Faire
   $C_i \leftarrow \text{Capacité\_du\_port\_de\_sortie\_du\_}i^{\text{ème}} \text{ commutateur traversé par le paquet du trafic étudié}$  ;
Fin Pour
/*Initialisation de la taille maximale des paquets de la classe de service  $i$  (avec  $i \in \llbracket 2, n \rrbracket$ ) */
Pour  $i$  allant de 2 à  $n$  Faire
   $L_i \leftarrow \text{Taille\_maximale\_des\_paquets\_de\_la }i^{\text{ème}} \text{ classe de service}$  ;
Fin Pour
/* Initialisation des poids des différents commutateurs et des poids optimaux*/
Pour  $i$  allant de 1 à  $n$  Faire
  Pour  $j$  allant de 1 à  $m$  Faire
     $w_{i,j} \leftarrow 0$  ;
     $w_{op_{i,j}} \leftarrow 0$  ;
  Fin Pour
Fin Pour
/* Initialisation du délai maximal pour chaque configuration de poids */
 $D_{max} \leftarrow 0$  ;
/* Initialisation du meilleur taux de bande passante offert aux trafics annexes pour un seuil de délai donné */
 $TB_{pmax} \leftarrow 0$  ;
/* En appelant la procédure calcul_des_poids_optimaux avec les paramètres  $i=1$  et  $j=1$ , on parcourt toutes les combinaisons des poids*/
calcul_des_poids_optimaux (1,1) ;
Fin

```

$$TB_p = \frac{\sum_{i=1}^m \sum_{k=2}^n w_{k,i} L_k}{\sum_{i=1}^m (w_{1,i} L_1 + \sum_{k=2}^n w_{k,i} L_k)}$$

### 3.7. Utilisation des résultats obtenus par calcul réseau

Dans ce paragraphe, nous allons montrer l'apport du calcul réseau pour les systèmes temps-réel ayant des contraintes strictes. Pour cela nous allons nous baser sur deux exemples : le premier concerne le drone qui a été présenté dans le paragraphe 2.3.2 et le deuxième concerne une architecture Ethernet commutée qui a été présentée dans le paragraphe 2.3.3.

#### 3.7.1. Résultats du calcul réseau appliqués sur le drone (Diouri *et al.*, 2008)

Cet exemple a été traité entièrement en se basant sur la courbe de service donnée par l'équation (eq. 3.1) (Diouri *et al.*, 2007a). Pour le drone, les conditions de simulation et les outils de simulation ont été présentés dans la partie 2.3.2 ainsi que dans (Diouri *et al.*, 2008). Dans cette étude, il a fallu définir les poids qui permettent de rester en dessous du seuil du délai tout en garantissant le maximum de ressources au flux non contraint temporellement. Pour cela nous nous sommes basés sur les résultats obtenus dans (Diouri *et al.*, 2007a). Ainsi, pour un seuil de délai égal à 5 ms (exigence de commande), il faut configurer le poids de l'ordonnanceur relatif aux mesures à 9 et celui relatif au flux vidéo à 3. Dans cette partie, nous avons considéré que les 9 paquets correspondants aux 9 mesures forment un super-paquet (on peut se permettre cette hypothèse car les 9 mesures sont envoyées simultanément). Lorsque nous avons appliqué l'algorithme de calcul des poids optimaux sur ce super-flux (regroupant les flux des 9 mesures), nous avons trouvé qu'il fallait affecter un poids de 1 à ce super-flux (équivalent à un poids de 9 pour le trafic temps-réel généré par les 9 flux de mesures) et un poids de 3 au flux non contraint temporellement.

En utilisant le nouveau bloc réseau que nous avons développé sous TrueTime (chapitre 2), le résultat représenté dans la figure 3.14 a pu être obtenu.

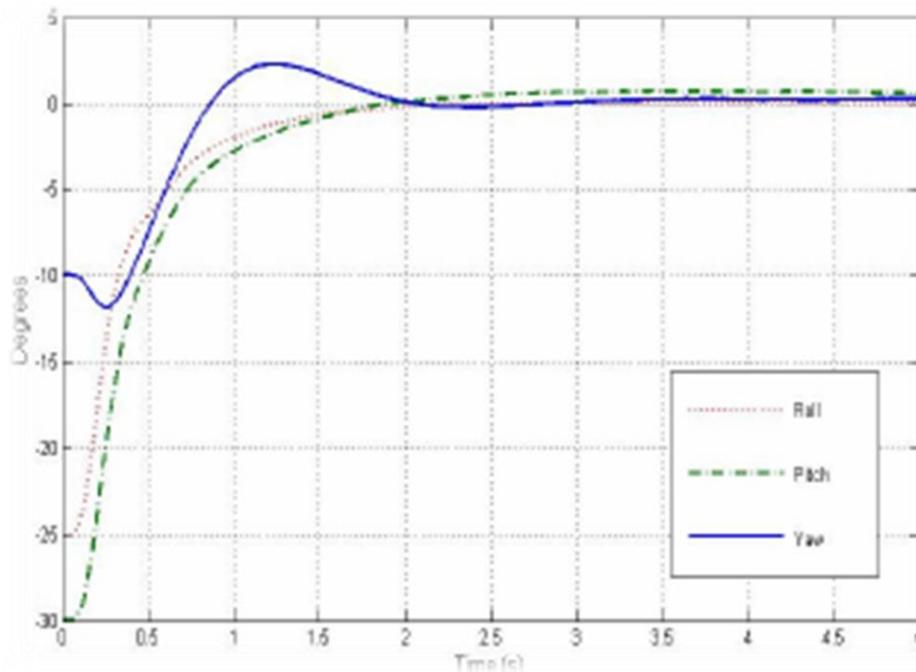


Figure 3.14. Ethernet commuté à 10 Mb/s, avec une CdS WRR et avec un trafic de fond non synchronisé (Diouri *et al.*, 2008)

La figure 3.14 montre que la configuration des poids du WRR basée sur la théorie du calcul réseau permet d'avoir un bon comportement du système. La comparaison entre la figure 3.14 et la figure 2.7 indique qu'en présence d'un trafic de fond non synchronisé, le comportement du système dans le cas d'une architecture Ethernet commutée implémentant le WRR est meilleur pendant la phase transitoire que celui obtenu avec le réseau CAN.

D'un autre côté, le tableau 3.2 présentant les résultats des simulations (Diouri *et al.*, 2008) montre bien l'intérêt d'utiliser un réseau Ethernet commuté implémentant le protocole d'ordonnancement WRR car celui-ci permet de maximiser les ressources du trafic de fond tout en garantissant un bon comportement du système. En effet le tableau 3.2 montre que l'Ethernet commuté implémentant le WRR permet d'avoir des temps de réponse similaires à ceux obtenus en utilisant un réseau CAN. Enfin, le réseau Ethernet commuté offre des débits plus importants allant jusqu'à 10 Gb/s (dans notre exemple 10 Mb/s) comparés à la plupart des réseaux de terrain (tel que CAN dont le débit maximal est de 1 Mb/s). Cela permet d'utiliser ce surplus de bande passante pour transporter les autres flux. Dans cet exemple (tableau 3.2), la bande passante offerte aux flux non contraint temporellement par le réseau Ethernet commuté implémentant le WRR est de 8,76 Mb/s alors qu'elle n'est que de 0,923 Mb/s dans le cas d'un réseau CAN.

## SIMULATION RESULTS

Network type	Response time (s)			Remaining bandwidth (Mbit/s)
	Roll	Pitch	Yaw	
CAN	1.22	1.3	1.84	0.923
WRR	1.23	1.3	1.84	8.76

Tableau 3.2. Résultats des simulations (Diouri *et al.*, 2008)Conclusion

A partir de cet exemple, l'utilisation de l'Ethernet commuté avec WRR est intéressante dans le cas d'un réseau qui inclut des applications contraintes temporellement et d'autres qui ne le sont pas. Dans ce cas, il est possible de déduire à partir de la figure 3.14, de la figure 2.7 et du tableau 3.2 que le fait de combiner le protocole d'ordonnancement WRR avec la théorie du calcul réseau permet :

- d'avoir un bon comportement du système et même légèrement meilleur que celui obtenu avec le réseau CAN (dans la phase transitoire).
- de maximiser les ressources allouées au trafic non contraint temporellement.

**3.7.2. Résultats du calcul réseau appliqués sur un SCR simulé sur OPNET**

Dans cette partie l'équation (eq.3.3) a été utilisée pour identifier les poids optimaux du commutateur. Le SCR étudié a été présenté dans les paragraphes 2.3.3. Comme il a été précédemment dit dans le paragraphe 2.3.3.2, l'étude du SCR a été effectuée en utilisant le logiciel OPNET avec un modèle SCR que nous avons développé. Dans le cas de cet exemple, le seuil du délai à ne pas dépasser est de 10 ms. L'algorithme qui permet de déterminer la configuration de poids qui garantisse le seuil du délai tout en maximisant les ressources du trafic de fond a donné la configuration de poids suivante :

- Poids égaux à 1 pour les trafics temps-réel traversant les commutateurs 1 et 2 (figure 2.14).
- Poids égaux à 3 pour les trafics de fond traversant les commutateurs 1 et 2 (figure 2.14).

Le graphe du bas de la figure 3.15 montre que cette configuration de poids permet d'assurer la stabilité du SCR et les deux graphes du haut de la figure 3.15 montrent que les délais du trafic temps-réel (du contrôleur vers le process et inversement) sont toujours inférieurs à 10 ms.



Figure 3.15. Comportement du SCR en utilisant la configuration des poids basée sur le calcul réseau

Ces 2 courbes du délai illustrent bien le caractère rare du pire cas (le seuil du délai n'est jamais dépassé). Ceci est toutefois à relativiser compte tenu de l'obligation du respect strict des exigences (à savoir aucun dépassement de seuil toléré).

### 3.8. Conclusion

Dans ce chapitre, nous avons présenté une méthodologie basée sur le calcul réseau pour régler les poids du commutateur dans le cas d'applications ayant des contraintes temps-réel strictes.

Une nouvelle courbe de service nommée « courbe de service en escalier non uniforme » a été proposée. De plus, des propriétés de convolutions ont été apportées pour cette courbe de service. Le principal intérêt de cette nouvelle courbe sera de pouvoir réduire le pessimisme de modélisation du service offert par un ordonnancement WRR, et ainsi, de réduire le conservatisme du majorant du délai, et *in fine*, de réduire au plus juste la réservation de bande passante pour le trafic temps-réel.

Cependant, même si l'écart entre les délais maxima obtenus par le calcul réseau et ceux observés par simulation est petit, ces bornes de délais estimées sont statistiquement rarement atteintes (paragraphe 3.5). Ce qui conduit à allouer en général plus de bande passante que nécessaire au trafic temps-réel au détriment des autres trafics. Pour des applications à contraintes temps-réel souples, qui peuvent tolérer des dépassements temporels, il est possible d'envisager de régler dynamiquement les poids du WRR. Le but est alors soit d'offrir plus de bande passante aux trafics annexes ou de la limiter si les délais du trafic temps-réel deviennent trop proches du seuil critique défini par l'application de contrôle/commande.

Ainsi, dans le chapitre suivant, nous considérerons une autre méthode pour commander les poids du commutateur. La méthode utilisée dans le chapitre suivant est basée sur la logique floue (Zadeh, 1965) et nous allons l'appliquer seulement dans le cas des systèmes temps-réel ayant des contraintes temps-réel souples.

# Chapitre 4 :

## Approche commande de réseau pour les SCR à contraintes temps-réel souples

### 4.1. Introduction

Ce chapitre étudie les SCR dont les contraintes temporelles sont considérées comme souples. Pour ce type d'application, des seuils temporels sont également définis pour la surveillance et le contrôle, mais le dépassement de ces seuils n'implique pas forcément un dysfonctionnement du système. Ce non respect des contraintes temporelles est acceptable dans certaines limites au-delà desquelles le système devient inutilisable. Pour un système temps-réel souple, un dépassement exceptionnel est donc toléré. Il pourra ensuite être rattrapé à l'exécution suivante. Cela implique généralement de s'assurer qu'il n'y ait pas de dépassements successifs du seuil pour pouvoir permettre ce rattrapage. C'est dans ce cadre que (Liu et Layland, 1973) mentionnent qu'une distribution statistique du délai est acceptable. Contrairement au chapitre précédent qui traite des SCR à contraintes temps-réel strictes, et où le réseau doit être configuré hors-ligne pour s'assurer que celui-ci garantisse des performances temporelles strictes quelque soit l'évolution instantanée de la charge de fond, l'objectif de ce chapitre est de permettre une configuration en ligne du réseau qui s'adapte en fonction des usages. Les phases de reconfiguration du réseau peuvent entraîner des écarts entre les exigences temporelles applicatives et les performances offertes par le réseau. C'est pourquoi ce chapitre ne s'adresse qu'aux applications temps-réel souple. L'intérêt de ces phases de reconfiguration est de pouvoir optimiser la répartition de la bande passante entre les usagers du réseau dont les besoins peuvent évoluer dans le temps. Une configuration statique telle qu'elle est présentée dans le précédent chapitre ne permet pas cette adaptation. En effet, elle donne une bande passante figée quelque soit les performances instantanées de la commande (QdC) du SCR.

Le travail est divisé en deux phases : la première phase consiste à identifier le comportement du réseau et notamment des délais du trafic selon la configuration de l'ordonnanceur WRR implanté dans les commutateurs Ethernet. A partir de cette identification, la deuxième phase a pour objectif d'utiliser le modèle identifié pour reconfigurer dynamiquement le réseau en fonction du trafic et des contraintes applicatives. Différentes techniques ont été envisagées

durant cette thèse comme notamment l'utilisation des plans d'expériences, mais nous présentons uniquement dans ce mémoire une démarche basée sur une modélisation floue que nous avons menée jusqu'au bout. Le recours à la logique floue dans notre cas se justifie d'une part parce que l'on traite des applications ayant des contraintes temps-réel souples, où le dépassement du seuil du délai est accepté si cela ne dure pas longtemps. D'autre part le délai que nous cherchons à modéliser présente plusieurs non linéarités (Oh et Pedrycz, 2000). Dans ce cas une identification par les méthodes classiques s'avère fastidieuse et quelquefois difficilement applicable. La méthodologie d'identification du modèle du délai présentée dans ce chapitre est facilement reproductible pour d'autres SCR.

Dans cette étude, nous utilisons une architecture Ethernet commutée implantant le protocole d'ordonnancement WRR. Notre objectif est d'agir sur les poids du WRR afin que les délais de bout en bout soient en dessous d'un certain seuil. Plus concrètement, on cherche à ce que la valeur du délai reste dans une plage de variation qui assure un bon fonctionnement de l'application temps-réel souple tout en garantissant le maximum de bande passante au trafic non contraint temporellement. Dans ce cas (applicatif), une expertise a permis de définir 3 classes de délai correspondant aux actions à effectuer sur le réseau. Ces classes sont Délai Parfait « DP », Délai Intermédiaire « DI » et Délai Non Toléré « DNT ». La classe DNT correspond aux délais qui peuvent causer l'instabilité du système. Les classes DI et DP correspondent aux délais tolérés par l'application.

Pour les applications temps-réel souple, les classes de Délai, définies par l'expert du domaine, ne sont pas disjointes. Il n'y a pas de limites temporelles strictes pour les délimiter, cette séparation est plutôt subjective et dépend principalement de l'application. C'est pourquoi, nous avons choisi d'exprimer ces catégories de manière linguistique et de nous placer dans le cadre de la Théorie des Ensembles Flous (Zadeh, 1965). Ce choix se justifie par le fait que la logique floue offre une capacité de modélisation et de traitement de données exprimées de manière linguistique (Zadeh, 1975). Cette identification floue nous permet d'établir des règles d'inférences qui mettent en relation les poids du commutateur qui représentent les entrées du système avec le délai qui en est la sortie. A partir de cela, nous souhaitons concevoir une commande floue permettant d'avoir un comportement du délai qui satisfasse les contraintes temps-réel de l'application. Selon (Richalet, 1993), dans la conception de systèmes de commande basés sur un modèle, la modélisation et l'identification du process peut prendre la majeure partie du temps et des coûts, ce qui s'est avéré être le cas pour nos travaux.

La logique floue a déjà été utilisée dans le contexte des communications temps-réel utilisant le réseau Ethernet (Kweon *et al.*, 2000) (Carpenzano *et al.*, 2002) (Lo Bello *et al.*, 2005). (Kweon *et al.*, 2000) ont proposé un mécanisme d'adaptation nommé « Harmonic-Increase and Multiplicative-Decrease » (HIMD) qui réagit avec la détection de collisions sur une période de temps spécifique. (Carpenzano *et al.*, 2002) ont proposé une méthode de lissage basée sur la logique floue qui montre que des délais aller-retour obtenus avec cette méthode sont inférieurs à ceux obtenus avec la méthode HIMD (Kweon *et al.*, 2000).

Après un rapide rappel théorique sur la logique floue, nous détaillons notre méthode d'identification par classification. Puis nous exposons la stratégie de commande floue qui permet d'assurer un bon fonctionnement de l'application temps-réel souple tout en garantissant le maximum de bande passante au trafic non contraint temporellement. Enfin, on conclut par une comparaison qui montre l'apport de notre proposition pour les applications temps-réel souple.

## 4.2. Présentation de la logique floue

La logique floue a été initiée par Zadeh en 1965 (Zadeh, 1965) dans le but de formaliser des données de nature imprécise et cela afin de résoudre des problèmes complexes. Selon (Bouchon Meunier, 1995), il existe en général trois natures d'imperfections liées aux connaissances : l'incertitude, l'imprécision et l'incomplétude. Dans notre cas la principale imperfection reste l'incomplétude. En effet, il est quelquefois difficile, au vu du procédé expérimental, d'obtenir des données sur toute la plage de variation du délai. Dans notre cas, l'identification se fera à partir de simulations réalisées sur le logiciel OPNET afin d'évacuer les problèmes liés à l'incertitude ou à l'imprécision. Les délais unidirectionnels entre deux stations peuvent donc être mesurés de manière très précise. En revanche, dans une plateforme réelle, cette synchronisation n'est jamais parfaite et le délai devient imprécis.

Le recours à la simulation se justifie donc par le fait qu'il peut être difficile de mesurer précisément des délais unidirectionnels en pratique. L'utilisation de la simulation se justifie aussi par le fait qu'on peut facilement tester ce que nous allons proposer en termes d'algorithmes d'adaptation de poids qui seront implantés dans les commutateurs.

Dans ce paragraphe, nous présentons brièvement la logique floue et montrons l'intérêt de l'utiliser dans le cas des systèmes temps-réel souple. Le lecteur intéressé par plus de détails concernant la logique floue pourra se référer à (Bouchon Meunier, 1995).

### 4.2.1. Concept de sous-ensemble flou

Le concept de sous-ensemble flou a été énoncé pour manipuler des classes non disjointes où les données n'appartiennent pas strictement à l'une d'elles. Il est aussi possible d'effectuer des opérations sur ces sous-ensembles flous (par exemple : l'égalité, l'inclusion, l'intersection l'union, produit cartésien, etc.).

Définition d'un sous-ensemble flou (Bouchon Meunier, 1995) : Un sous-ensemble flou  $A$  de  $X$  est défini par une fonction d'appartenance qui associe à chaque élément  $x$  de  $X$ , le degré  $f_A(x)$ , compris entre 0 et 1, avec lequel  $x$  appartient à  $A$  :  $f_A : X \rightarrow [0,1]$ .

Définition d'un sous-ensemble flou normalisé (Bouchon Meunier, 1995) : Le sous-ensemble flou  $A$  de  $X$  est normalisé si sa hauteur  $h(A)$  est égale à 1.

Définition du produit cartésien de sous-ensembles flous (Bouchon Meunier, 1995) : Soient des sous-ensembles flous  $A_1, A_2, \dots, A_r$ , respectivement définis sur  $X_1, X_2, \dots, X_r$ , on définit leur produit cartésien  $A = A_1 \times A_2 \times \dots \times A_r$ , comme un sous-ensemble flou de  $X$  de fonction d'appartenance :  $\forall x = (x_1, \dots, x_r) \in X, f_A(x) = \min(f_{A_1}(x_1), \dots, f_{A_r}(x_r))$ .

#### 4.2.2. Concept de relation floue

Selon (Zadeh, 1971), ces relations floues montrent les liaisons graduelles entre des éléments appartenant à un même ensemble.

Définition de relation floue (Bouchon Meunier, 1995) : Une relation floue  $R$  entre  $r$  ensembles de référence  $X_1, X_2, \dots, X_r$ , est un sous-ensemble flou de  $X_1 \times X_2 \times \dots \times X_r$ , de fonction d'appartenance  $f_R$ .

#### 4.2.3. Notion de variable linguistique

La notion de variable linguistique a été introduite par (Zadeh, 1975), celle-ci est utilisée afin de modéliser les connaissances imprécises d'une variable dont la valeur précise peut être inconnue.

Définition d'une variable linguistique (Bouchon Meunier, 1995) : Une variable linguistique est un triplet  $(V, X, T_v)$ , dans lequel  $V$  est une variable définie sur un ensemble de référence  $X$ . L'ensemble  $T_v = \{A_1, A_2, \dots\}$ , fini ou infini contient des sous-ensembles flous normalisés de  $X$ , utilisables pour caractériser  $V$ .

#### 4.2.4. Notion de proposition floue

Dans ce sens, on distingue les propositions floues élémentaires et les propositions floues générales.

Définition d'une proposition floue élémentaire (Bouchon Meunier, 1995) : Etant donné un ensemble  $L$  de variables linguistiques et un ensemble  $M$  de modificateurs, une proposition floue élémentaire est définie à partir d'une variable linguistique  $(V, X, T_v)$  de  $L$  par la qualification «  $V$  est  $A$  », où  $A$  est une caractérisation floue appartenant à  $T_v$  ou à  $M(T_v)$ .

Définition d'une proposition floue générale (Bouchon Meunier, 1995) : Une proposition floue générale est obtenue par l'utilisation conjointe de propositions floues élémentaires «  $V$  est  $A$  », «  $W$  est  $B$  » ... pour des variables  $V, W$  ... supposées non-interactives.

Selon (Bouchon Meunier, 1995), il est possible de construire des propositions floues par conjonction («  $V$  est  $A$  et  $W$  est  $B$  »), par disjonction («  $V$  est  $A$  ou  $W$  est  $B$  ») ou par

implication (« V est A implique que W est B ») portant sur des propositions floues quelconques.

#### 4.2.5. Notion de règle floue

Les règles floues permettent de représenter des connaissances. Dans cette étape, on s'intéresse à l'identification des règles floues de type « SI X est A ALORS Y est B ». La partie « X est A » constitue la prémisse de la règle et la partie « Y est B » constitue la conclusion de la règle. Le type de raisonnement utilisé est déductif car on cherche à déduire les sorties en fonction des entrées. Selon (Dubois et Prade, 1996), les règles floues peuvent être décomposées en deux types : les règles implicatives qui constituent une extension des règles classiques et les règles conjonctives (comme celles dites de Mamdani) qui sont généralement utilisées en contrôle flou. (Dubois et Prade, 1996) ont distingué d'une part les règles à possibilité et les règles anti-graduelles pour les règles conjonctives et d'autre part les règles à certitude et les règles graduelles pour les règles implicatives. Ils ont proposé l'interprétation suivante pour les 4 différents types de règles :

- Règles implicatives à certitude : Selon (Dubois et Prade, 1996), les règles implicatives à certitude de type « SI X est A ALORS Y est B » peuvent s'exprimer comme suit : « Plus X est A, plus il est certain que Y est B ».
- Règles implicatives graduelles : Selon (Dubois et Prade, 1996), les règles implicatives à certitude de type « SI X est A ALORS Y est B » peuvent s'exprimer comme suit : « Plus X est A, plus Y est B ».
- Règles conjonctives à possibilité : Selon (Dubois et Prade, 1996), les règles conjonctives à possibilité de type « SI X est A ALORS Y est B » peuvent s'exprimer comme suit : « Plus X est A, ALORS plus il est possible que Y soit B » tel que l'opérateur ALORS correspond à une conjonction.
- Règles conjonctives antigraduées : Selon (Dubois et Prade, 1996), les règles conjonctives antigraduées de type « SI X est A ALORS Y est B » peuvent s'exprimer comme suit : « Plus X est A, et moins Y est rapproché de X, moins Y est B ». Le nom de « règles antigraduées » est issu du fait qu'elles ont un comportement inverse à celui des règles implicatives graduelles.

En se basant sur cette différenciation (Jones, 2007) a proposé de suivre le raisonnement suivant pour définir le type de règle à utiliser. Selon (Jones, 2007), si la règle représente une contrainte, une obligation ou une restriction, il faudrait choisir une règle implicative et dans ce sens il distingue entre les deux cas suivants :

- Si celle-ci représente la certitude ou la typicalité, il faut opter pour une règle à certitude.
- Si elle représente une évolution graduelle ou la proximité, il faut choisir une règle graduelle.

(Jones, 2007) a alors proposé d'utiliser une règle conjonctive à possibilité lorsqu'on a des exemples de ce qui est possible ou des données. Notre cas applicatif correspond à cette dernière catégorie de règles.

### 4.3. Identification du modèle du délai par classification floue

Le but de cette identification floue est de pouvoir interpréter le comportement du système à travers des règles qui mettent en relation les poids du protocole d'ordonnancement avec le délai de l'application temps-réel souple. Cette modélisation est faite à partir de résultats de simulations, en relation avec les définitions des classes de fonctionnement. Le modèle obtenu concerne uniquement les applications temps-réel souple qui tolèrent dans certaines limites le dépassement du seuil strict du délai.

#### 4.3.1. Présentation du système étudié et des conditions de simulation

Dans cette partie, nous allons décrire le système sur lequel se porte notre étude. Ensuite, nous allons déterminer les différentes classes de délais. Enfin, nous allons montrer l'impact de l'évolution des poids de l'ordonnanceur sur le délai du trafic temps-réel.

##### 4.3.1.1. Description du système étudié

Le système pour lequel l'identification des délais a été effectuée est représenté dans la figure 4.1. Cette identification a été faite en se basant sur des simulations qui ont été effectuées en utilisant le logiciel OPNET. Dans ces simulations, l'analyse porte sur le délai de l'application temps-réel qui varie en fonction des poids du WRR et en fonction de la charge.

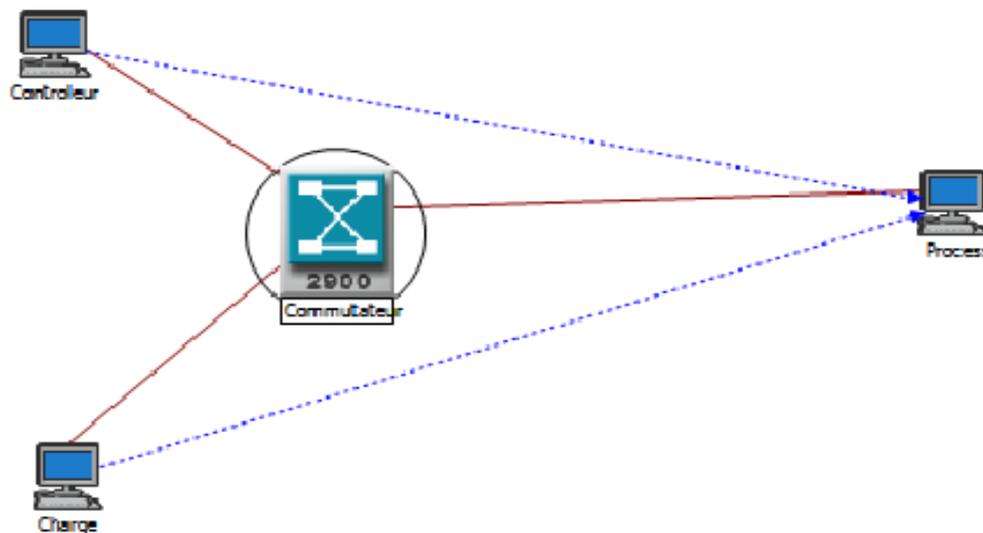


Figure 4.1. Architecture utilisée pour l'identification du délai du trafic temps-réel souple

Comme l'illustre la figure 4.1, le système se compose d'un contrôleur, d'un process, d'une station de charge qui génère le trafic de fond et d'un commutateur qui permet de mettre en place le protocole d'ordonnancement WRR. Les liens de notre architecture sont à 10 Mb/s. On dispose de deux flux. Un flux est envoyé du contrôleur vers le process appelé trafic temps-réel. Pour l'identification, on s'intéresse uniquement à ce flux. Par symétrie nous allons appliquer les résultats de l'identification au flux du process vers le contrôleur. On considère dans la suite que le process envoie au contrôleur un flux aux caractéristiques identiques à celles du flux du contrôleur vers le process. Dans cette étude, la taille des paquets du flux temps-réel est de 125 octets et ils sont envoyés chaque 10 ms. Un trafic de fond est envoyé par la station de charge vers le process. Les deux flux transitent obligatoirement par le commutateur.

Pour chaque configuration de poids du commutateur, trente simulations (avec des « *seed* » différents) ont été effectuées dans le but de rendre notre estimateur fiable (OPNET, 2005). Le comportement du système a été étudié avec des charges dont les moyennes sont les suivantes : 5 Mb/s, 6 Mb/s et 7 Mb/s et pour chaque moyenne de charge (5 Mb/s, 6 Mb/s et 7 Mb/s) 314 configurations de poids ont été effectuées. La durée de chaque simulation est de 5 minutes. La taille des paquets du trafic de fond varie autour de 1500 octets selon une loi exponentielle. L'inter-arrivée des paquets du trafic de fond suit également une loi exponentielle qui varie autour de 1,712 ms pour une charge moyenne de 7 Mb/s, de 2 ms pour une charge moyenne de 6 Mb/s et de 2,398 ms pour une charge moyenne de 5 Mb/s .

L'identification du délai a été effectuée en se basant sur la borne supérieure des maxima moyens obtenus pour un intervalle de confiance de 90 %. La première étape a été de définir les classes de délais imposées par le système de commande. Dans notre cas (présenté dans le paragraphe 2.3.3.1), le délai du contrôleur vers le process  $D_{CP}$  ainsi que le délai du process vers le contrôleur  $D_{PC}$  se compose d'une partie qui est fixe  $D_F$  et d'une partie qui est variable. Seule l'identification de la partie variable nous intéresse ici.

En notant :

- $D_{Com1_{CP}}$  : Le délai de traversée du commutateur 1 quand un paquet transite du contrôleur vers le process (figure 2.14).
- $D_{Com2_{CP}}$  : Le délai de traversée du commutateur 2 quand un paquet transite du contrôleur vers le process (figure 2.14).
- $D_{Com1_{PC}}$  : Le délai de traversée du commutateur 1 quand un paquet transite du process vers le contrôleur (figure 2.14).
- $D_{Com2_{PC}}$  : Le délai de traversée du commutateur 2 quand un paquet transite du process vers le contrôleur (figure 2.14).

Ainsi :

$$D_{CP} = D_F + D_{Com1_{CP}} + D_{Com2_{CP}}$$

$$D_{PC} = D_F + D_{Com1_{PC}} + D_{Com2_{PC}}$$

Pour  $D_{CP}$ , la partie qui est variable est  $D_{Com1_{CP}} + D_{Com2_{CP}}$ . Et pour  $D_{PC}$ , la partie qui est variable est  $D_{Com1_{PC}} + D_{Com2_{PC}}$ .

Dans le cas de l'architecture Ethernet commutée considérée dans cette thèse (paragraphe 2.3.3.1 et 2.3.3.2), nous avons :  $D_F = 1,44 \text{ ms}$ . Ce terme  $D_F$  correspond au délai subi par le trafic temps-réel au niveau des liens de l'architecture Ethernet commutée. L'identification du délai de traversée d'un commutateur en fonction des poids de l'ordonnanceur WRR qui sera faite dans la suite servira pour l'architecture Ethernet commutée considérée dans cette thèse laquelle se compose de deux commutateurs (figure 2.14) et dont les équations du contrôleur et du process ont été présentées dans les paragraphes 2.3.3.1.

#### 4.3.1.2. Notion de délai associé à une contrainte souple

Pour cette application temps-réel souple, le seuil est de 10 ms. Donc, le seuil (souple) du délai à ne pas dépasser pour traverser un commutateur est de 4,28 ms ( $= \frac{(10 \text{ ms} - D_F)}{2}$ ).

De ce fait, selon notre expertise, les 3 classes adaptées à notre application sont les suivantes :

- 1ère classe : Délai Parfait (DP) pour des  $D(w_1, w_2, \text{moyenne de charge})$  inférieurs ou égaux à 0,458 ms.
- 2ème classe : Délai Intermédiaire (DI) pour des  $D(w_1, w_2, \text{moyenne de charge})$  supérieurs à 0,458 ms et inférieurs ou égaux à 4,28 ms.
- 3ème classe : Délai Non Tolérés (DNT) pour des  $D(w_1, w_2, \text{moyenne de charge})$  supérieurs à 4,28 ms.

Il y a une certaine tolérance par rapport au seuil délimitant les classes du délai (temps-réel souple). Ainsi, nous proposons de modéliser cette tolérance en traduisant le délai sous la forme d'une variable linguistique. Dans ce cas la variable linguistique floue (paragraphe 4.2.3) est définie par le triplet :

- V c'est la variable Délai définie sur un référentiel X.
- Le domaine de variation de V est :  $X = [0, +\infty[$ .
- Tv est le vocabulaire choisi pour définir de manière linguistique les valeurs de V (Parfait, Intermédiaire, Non Toléré).

Le lien entre la valeur du délai et la variable linguistique qui le représente est donné par les fonctions d'appartenance définies dans la figure 4.2. Sur la figure 4.2, le  $\mu$  représente le degré d'appartenance :  $\mu \in [0, 1]$ .

$\mu$  : degré d'appartenance

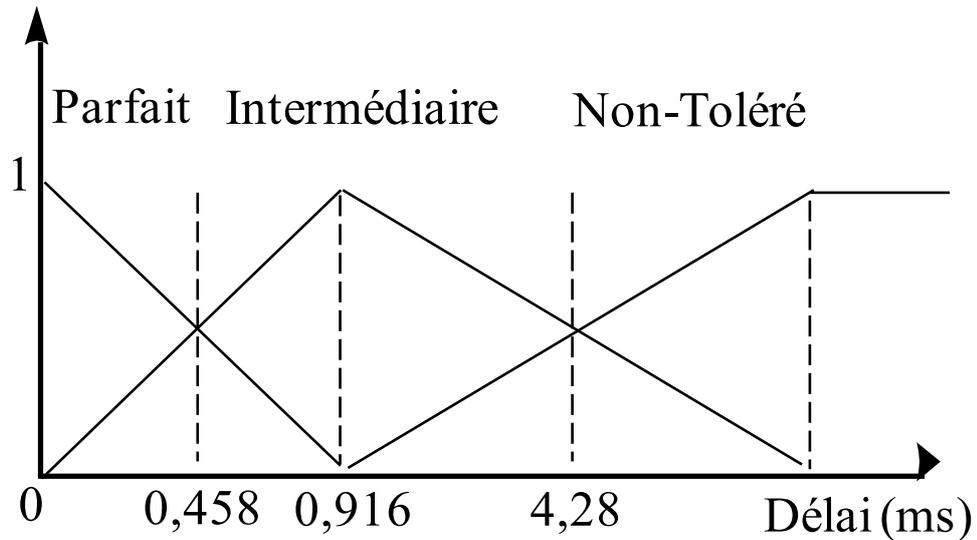


Figure 4.2. Définition linguistique du délai

En résumé, comme le montre la figure 4.2, on a :

- Quand le délai est égal à 0,458 ms alors :  $\mu(DP) = \mu(DI) = 0,5$
- Quand le délai est égal à 4,28 ms alors :  $\mu(DI) = \mu(DNT) = 0,5$

#### 4.3.1.3. Comportement du délai en fonction des entrées $w_1$ et $w_2$

Le comportement du système peut être décrit à partir de connaissances expertes. En effet, dans notre système on sait qu'à charge constante on a les règles suivantes :

- Si  $w'_1$  est supérieur à  $w_1$  ALORS  $D(w'_1, w_2) \leq D(w_1, w_2)$
- Si  $w'_1$  est inférieur à  $w_1$  ALORS  $D(w'_1, w_2) \geq D(w_1, w_2)$
- Si  $w'_2$  est supérieur à  $w_2$  ALORS  $D(w_1, w'_2) \geq D(w_1, w_2)$
- Si  $w'_2$  est inférieur à  $w_2$  ALORS  $D(w_1, w'_2) \leq D(w_1, w_2)$

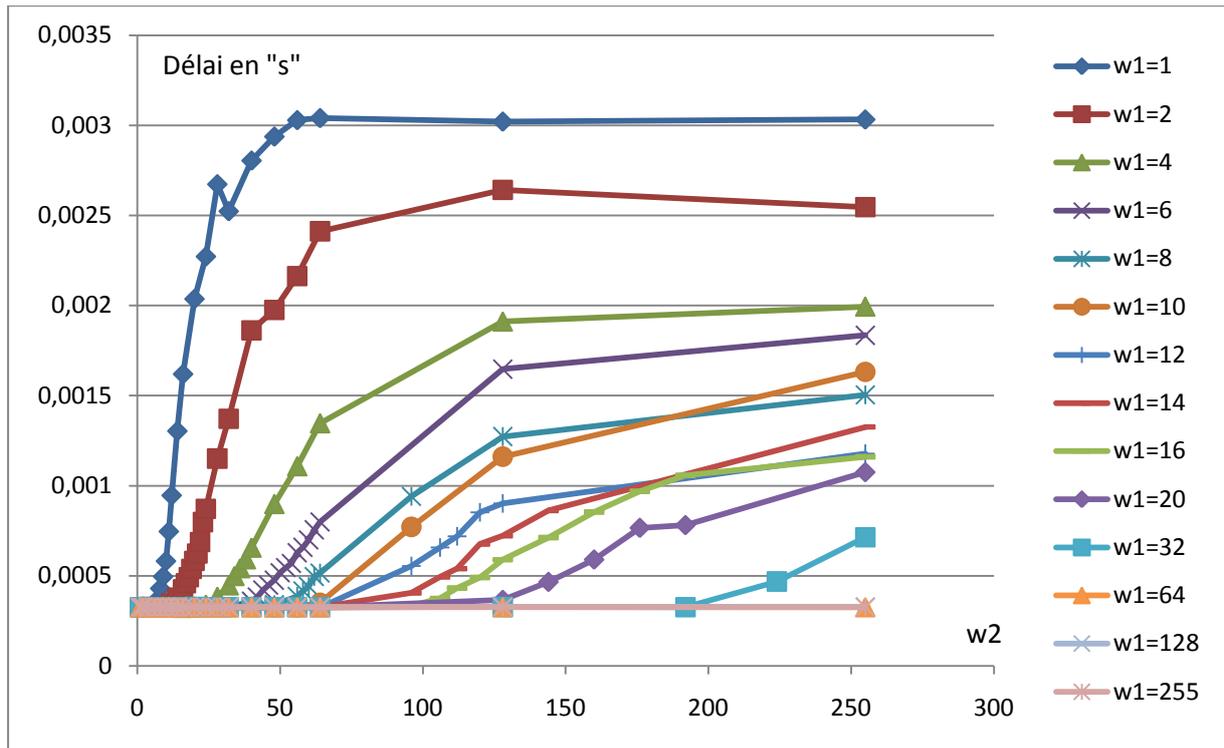
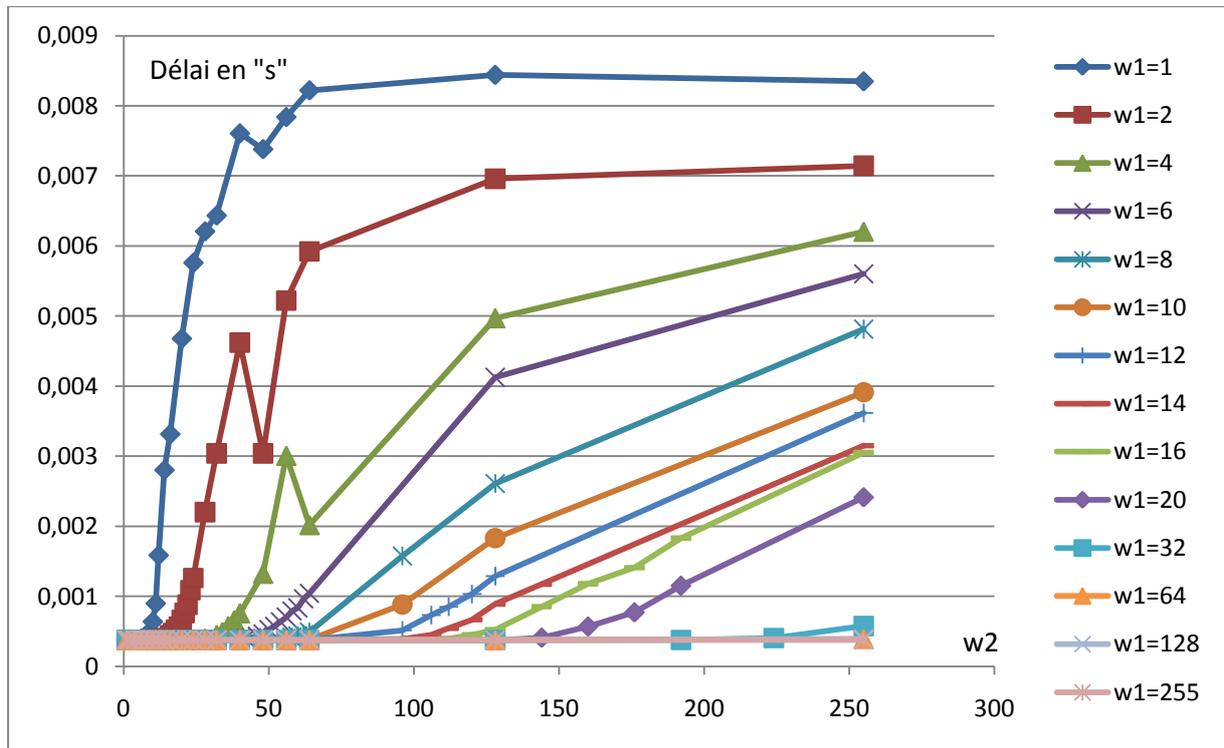
En revanche, on ne peut pas connaître le comportement de l'ordonnanceur WRR à charge constante dans le cas où «  $w'_1$  est supérieur à  $w_1$  ET  $w'_2$  est supérieur à  $w_2$  » et dans le cas où «  $w'_1$  est inférieur à  $w_1$  ET  $w'_2$  est inférieur à  $w_2$  » (tableau 4.1).

ET	$w'_2$ est supérieur à $w_2$	$w'_2$ est inférieur à $w_2$
$w'_1$ est supérieur à $w_1$	Modèle inconnu	$D(w'_1, w'_2) \leq D(w_1, w_2)$
$w'_1$ est inférieur à $w_1$	$D(w'_1, w'_2) \geq D(w_1, w_2)$	Modèle inconnu

Tab. 4.1. Comportement du délai à charge constante en variant simultanément les deux poids

De plus, si à charge constante on peut dire « Si  $w'_2$  est supérieur à  $w_2$  ALORS  $D(w_1, w'_2) \geq D(w_1, w_2)$  », on ne connaît pas la valeur au-delà de laquelle le délai devient quasi-égal. La figure 4.3 montre par exemple que pour  $w_1 = 1$  le délai reste quasi constant pour des valeurs de  $w_2$  supérieures ou égales à 56. On n'a pas une idée précise sur l'augmentation du délai quand elle a lieu (figures 4.3, 4.4 et 4.5).

On ne peut donc pas se contenter de ce type de règles. Dans notre cas en effet, la charge qui perturbe le système temps-réel n'est pas constante. Comme nous cherchons à identifier le modèle du délai qui tient compte aussi de la charge, notre modèle est forcément incomplet (voir paragraphe 4.2). Nous avons procédé à 942 simulations pour pouvoir établir le comportement du délai en fonction des poids et de la charge. Les figures 4.3, 4.4 et 4.5 montrent cette évolution. Nous proposons de baser l'identification sur ce lot de données afin d'établir un modèle qui régit le comportement du délai en fonction des poids de l'ordonnanceur et de la charge.

Figure 4.3. Délai en fonction de  $w_2$  à  $w_1$  constants et avec une charge moyenne de 5 Mb/sFigure 4.4. Délai en fonction de  $w_2$  à  $w_1$  constants et avec une charge moyenne de 6 Mb/s

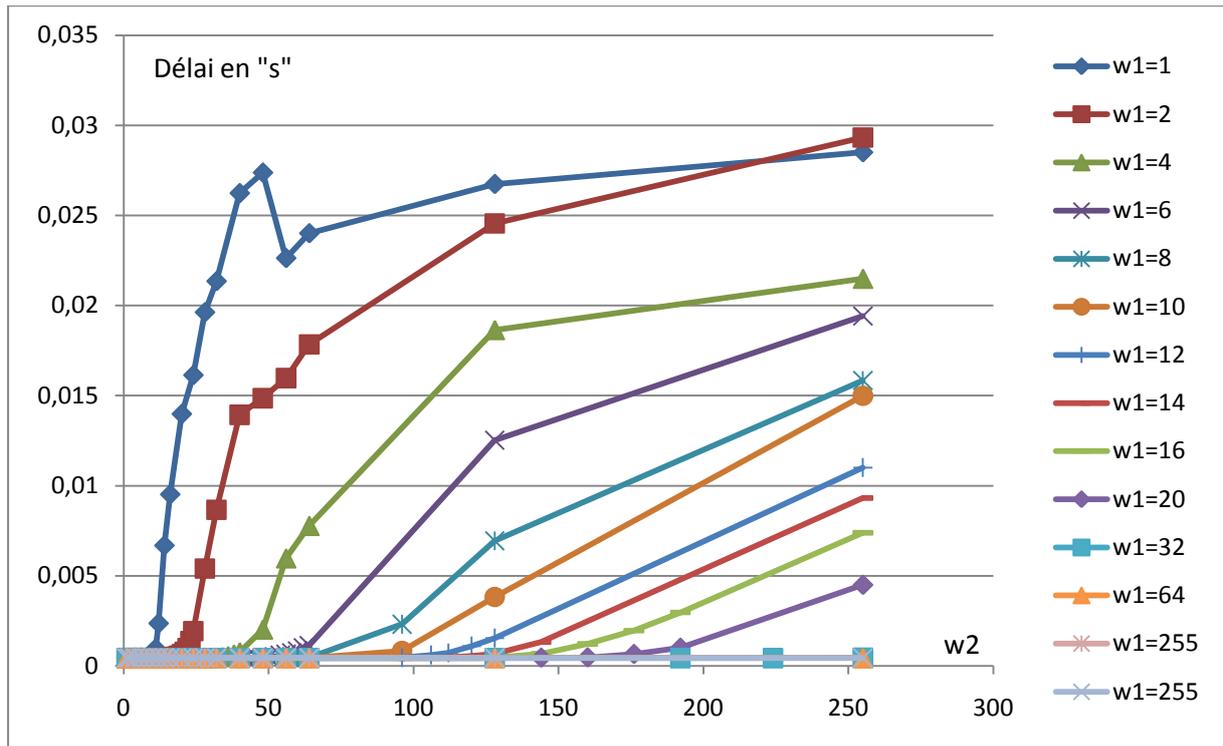


Figure 4.5. Délai en fonction de  $w_2$  à  $w_1$  constants et avec une charge moyenne de 7 Mb/s

Les résultats illustrés dans les figures 4.3, 4.4 et 4.5, corroborent les connaissances expertes. Effectivement, on constate que l'on a généralement : « Si  $w'_2$  est supérieur à  $w_2$  ALORS  $D(w_1, w'_2) \geq D(w_1, w_2)$  » et « Si  $w'_2$  est inférieur à  $w_2$  ALORS  $D(w_1, w'_2) \leq D(w_1, w_2)$  » sauf pour un nombre très restreint de points. En effet, pour  $w_1 = 2$  dans le cas d'une charge moyenne de 6 Mb/s (figure 4.4), lorsqu'on passe d'une valeur de 40 à une valeur de 48 pour  $w_2$ , le délai n'augmente pas comme il aurait dû, d'après la règle « Si  $w'_2$  est supérieur à  $w_2$  ALORS  $D(w_1, w'_2) \geq D(w_1, w_2)$  ».

On constate aussi (figures 4.3, 4.4 et 4.5), que le comportement du système présente plusieurs non-linéarités. Par exemple, la courbe qui représente l'évolution du délai pour un  $w_1 = 2$  dans le cas d'une charge moyenne de 7 Mb/s (figure 4.5). Ces non-linéarités nous ont conduits à ne pas utiliser une identification classique du modèle du délai.

### 4.3.2. Identification du système par règles floues conjonctives

Comme les connaissances sur le comportement du système sont partielles voire incomplètes, il est difficile de modéliser le fonctionnement par des règles implicatives. Ainsi nous proposons d'obtenir un modèle du système en exploitant les données. Nous nous orientons donc vers un modèle à base de règles conjonctives à possibilité (Dubois et Prade, 1996) (Jones, 2007).

L'objectif de cette identification est de pouvoir établir les règles d'inférence qui mettent en relation le délai de traversée du commutateur avec les poids de notre ordonnanceur WRR (figure 4.2)

On cherche à identifier le modèle de telle sorte à ce qu'il soit utilisable ensuite pour la commande de réseau. Le but est donc de fournir un modèle ayant un jeu de règles qui soit interprétable. Pour cela, nous allons nous appuyer sur des techniques issues du domaine de la reconnaissance de formes en traitement d'images. Les poids  $w_1$  et  $w_2$  sont considérés comme étant des primitives en entrée d'un classificateur. L'interprétabilité nous interdit l'utilisation de classificateurs « boîte noire » comme les réseaux de neurones (Tou *et al.*, 2009) ou les Séparateurs à Vaste Marge SVM (en anglais : *Support Vector Machine*) (Hao *et al.*, 2007). Le choix d'un système à base de règles semble le plus adapté (Roubos *et al.*, 2003) (Nauck et Kruse, 1998) (Nauck et Kruse, 1999).

Il existe plusieurs classificateurs permettant d'obtenir des règles comme les arbres de décision (DTM) (Michie *et al.*, 1994) et les algorithmes génétiques (Cordon *et al.*, 1999). Le principal inconvénient de ces méthodes est qu'elles nécessitent un grand nombre d'échantillons. C'est pourquoi nous avons choisi d'utiliser un classificateur flou à base de règles linguistiques - Fuzzy Rule Classifier (FRC) - (Bombardier et Schmitt, 2010) adapté aux lots de données réduits. Sa capacité de généralisation par rapport à d'autres classificateurs (K-ppv, RdN, SVM) a été montrée dans (Bombardier et Schmitt, 2010).

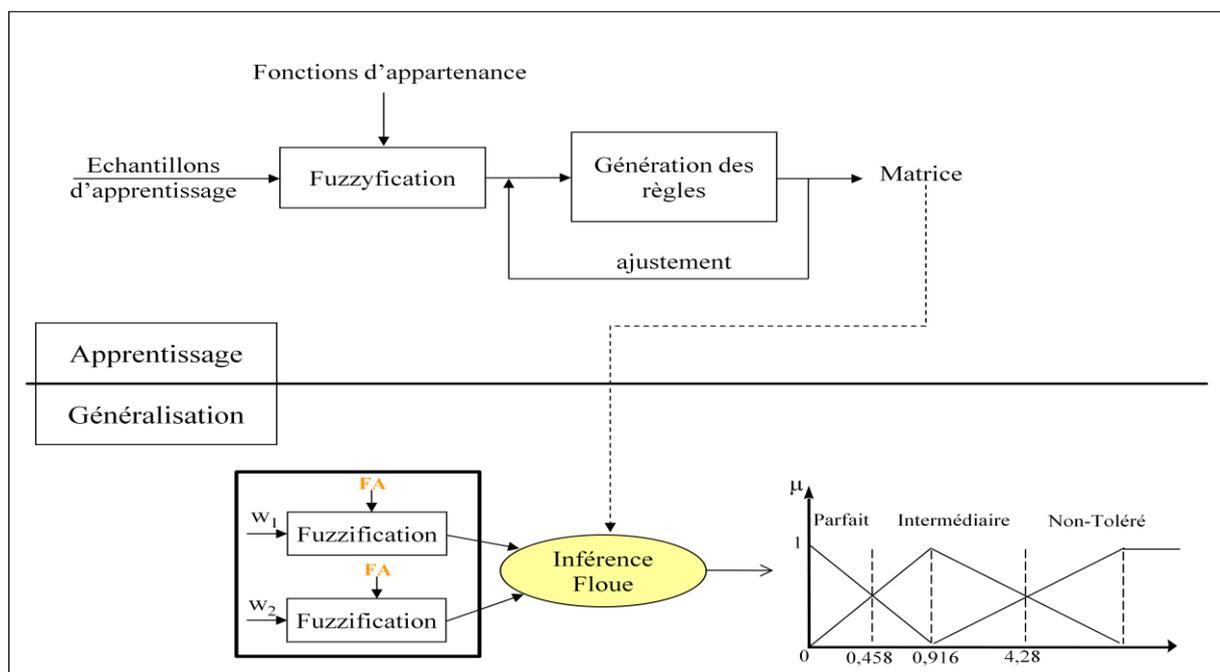


Figure 4.6. Description générale du système à identifier en utilisant le FRC (Bombardier et Schmitt, 2010)

Comme on le voit d'après la figure 4.6, la méthode de classification floue utilisée est un mécanisme par apprentissage supervisé qui inclut deux parties : l'apprentissage pour obtenir le modèle du système et l'exploitation pour classer une donnée inconnue. Dans notre cas nous n'utilisons que la phase d'apprentissage qui se décompose en trois étapes, la fuzzification, la génération des règles floues et l'ajustement du modèle.

#### 4.3.2.1. Fuzzification

Le système à identifier (figure 4.6) a pour entrées les poids de l'ordonnanceur  $w_1$  et  $w_2$ . Il s'agit donc de traduire ces variables numériques en variables linguistiques (Zadeh, 1975) (Bouchon Meunier, 1995) (paragraphe 4.2.3). C'est l'objet de l'étape de fuzzification. Dans ce cas, le triplet  $(V, X, Tv)$  est le suivant :

- $V$  c'est la variable  $(w_1, w_2)$  définie sur un référentiel  $X$ .
- Le domaine de variation de  $V$  est :  $X = \llbracket 1, 255 \rrbracket^2$ .
- $Tv$  est le vocabulaire choisi pour définir de manière linguistique les valeurs de  $V$  (Petit, Moyen, Grand).

L'ensemble  $Tv = \{A1, A2 \dots\}$  qui peut être fini ou infini, contient les sous-ensembles flous normalisés de  $X$  qui sont utilisés dans le but de caractériser la variable  $V$ . Chaque  $A_i$  est défini par un degré d'appartenance  $\mu_{A_i}(x)$ . Par exemple, comme le montre la figure 4.2, le délai est décomposé de manière réparties en trois termes : « Parfait » (DP), « Intermédiaire » (DI) ou « Non Toléré » (DNT). Ainsi, ces variables sont caractérisées par un vecteur composé de trois degrés d'appartenance :  $[\mu_P(x), \mu_I(x), \mu_{NT}(x)]$ .

La forme des fonctions d'appartenance a aussi un rôle important dans l'étape de fuzzification car elle peut avoir un impact sur les taux de reconnaissance. Il existe trois formes de base pour les fonctions d'appartenance : triangulaire, trapézoïdale, gaussienne qui peuvent être combinées. Dans notre cas, la forme la plus adaptée est la forme trapézoïdale-triangulaire (figure 4.7). Cette forme a été choisie empiriquement à partir d'essais comparatifs faits avec les formes trapézoïdale, triangulaire, gaussienne et trapézoïdale gaussienne. Ce choix est compatible avec les résultats obtenus pour d'autres cas applicatifs (Bombardier et Schmitt, 2010).

Dans l'étape de fuzzification, il s'agit de déterminer le nombre de termes de la variable ainsi que leur répartition sur l'intervalle de variation. La solution la plus simple consiste à faire une partition régulièrement distribuée sur l'univers de discours. Cette solution n'est pas toujours la plus appropriée car elle aboutit à un grand nombre de termes et donc à plus de règles ce qui rend le système difficilement interprétable.

Il est aussi possible de recourir à des méthodes permettant d'effectuer une adaptation automatique de la fuzzification. Ces méthodes sont en général basées sur les algorithmes génétiques (Cordon *et al.*, 2004) ou sur les algorithmes de *clustering* (De Carvalho, 2007).

Cependant, en considérant les algorithmes génétiques, il est très difficile d'obtenir des résultats corrects par rapport aux lots de données dont nous disposons dans notre étude. En effet, ces techniques sont connues pour nécessiter un grand nombre de points d'apprentissage.

(Schmitt *et al.*, 2007) ont développé une méthode de fuzzification automatique adaptée aux lots d'échantillons réduits. Elle est basée sur l'étude des typicalités (Forest *et al.*, 2006) des classes de sorties. Néanmoins, le nombre de termes générés reste conséquent et inapproprié à notre étude. C'est pourquoi, nous avons préféré une fuzzification manuelle, adaptée en fonction des données d'entrée (expert) pour garantir l'interprétabilité du jeu de règles obtenu.

La fuzzification du poids  $w_1$  est illustrée dans la figure 4.7 et celle de  $w_2$  est illustrée dans la figure 4.8.

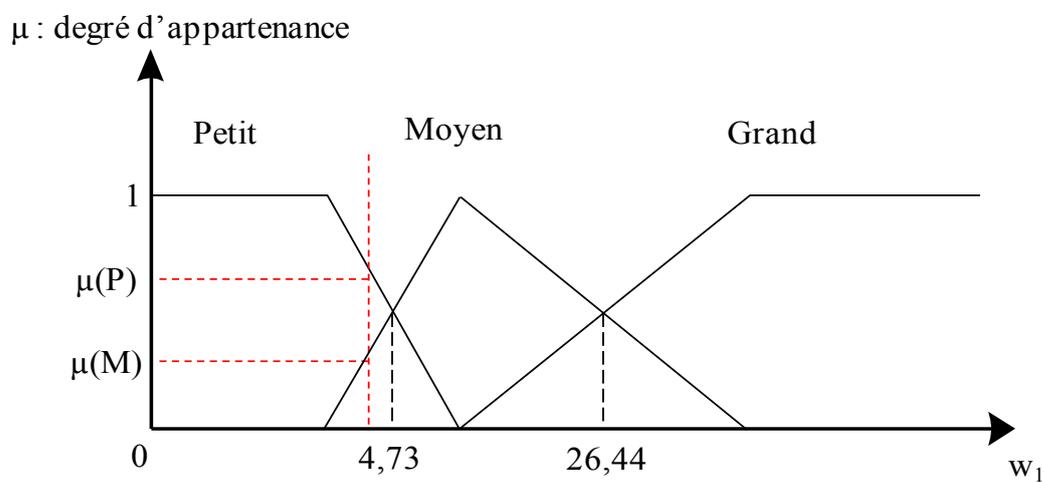


Figure 4.7. Fuzzification du poids  $w_1$  en 3 termes avec des courbes trapézoïdales-triangulaires

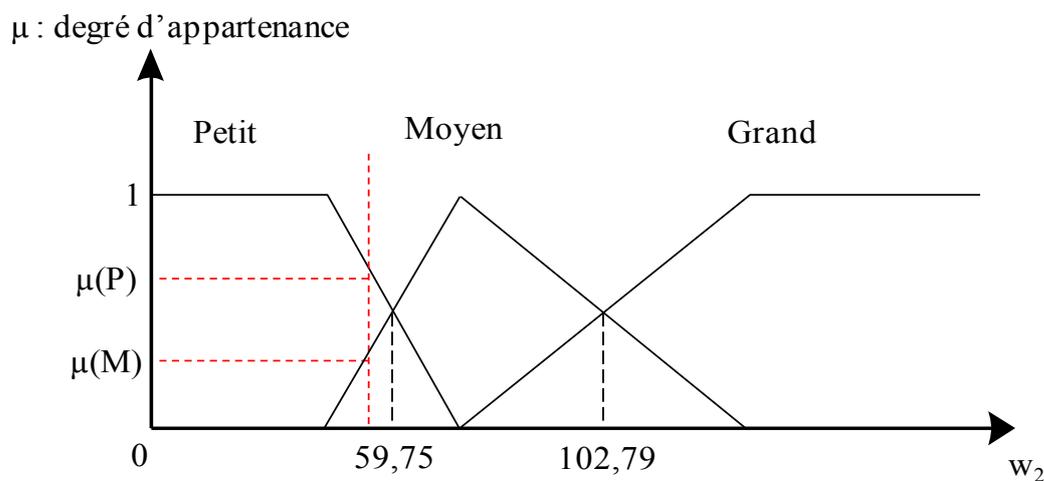


Figure 4.8. Fuzzification du poids  $w_2$  en 3 termes avec des courbes trapézoïdales-triangulaires

#### 4.3.2.2. Génération des règles floues

La deuxième étape consiste à obtenir le jeu de règles à partir des lots de données. Pour générer automatiquement les règles linguistiques, il est possible de recourir aux algorithmes génétiques (Cordon *et al.*, 1999) (Alcala *et al.*, 2007) qui présentent l'inconvénient d'être gourmands en nombre de données pour l'apprentissage. Cette méthode n'est pas réellement adaptée à notre problématique. Le FRC proposé par (Bombardier et Schmitt, 2010) s'appuie sur l'algorithme d'Ishibuchi (Ishibuchi *et al.*, 1992). Il correspond à un mécanisme d'inférence de type Larsen (Ishibuchi *et al.*, 1992), qui est préférable à celui de Mamdani surtout dans le cas où il y a plusieurs prémisses (Berthold, 2003). Les règles sont obtenues en favorisant la règle de réponse maximum à partir d'un lot de données préparé en se basant sur une expertise du domaine.

Chaque règle est activée en parallèle et un opérateur de disjonction est utilisé pour agréger les conclusions partielles permettant ainsi d'obtenir la décision finale.

#### 4.3.2.3. Ajustement du modèle

L'étape d'ajustement correspond à la partie itérative de l'algorithme (Ishibuchi *et al.*, 1997). Cet ajustement est effectué à partir des résultats donnés par le jeu d'échantillons d'apprentissage. L'algorithme génère un premier modèle et calcule un coefficient de confiance CF qui peut s'apparenter à un degré de vérité pour chaque règle.

Si le taux de classification est inférieur à un seuil défini par l'utilisateur, la partie itérative ajuste le modèle en modifiant ce coefficient de confiance CF. Il est augmenté si l'échantillon confirme la règle et diminué dans le cas contraire. En sortie, le système d'inférences floues délivre autant de degrés de possibilité que de classes de sortie, signifiant la possibilité que les poids donnés conduisent à un délai de type « Parfait », « Intermédiaire » ou « Non Toléré ». Ainsi, trois degrés de possibilité sont associés au délai concerné. La décision finale est prise à l'aide d'un opérateur de disjonction flou (Max).

#### 4.3.2.4. Résultats de la classification

Le modèle général que nous avons obtenu en appliquant la méthode sur toutes les mesures (942) prises pour les trois charges moyennes (5 Mb/s, 6 Mb/s et 7 Mb/s) présente l'avantage d'être au final indépendant de la charge.

Le tableau 4.2 donne les taux de reconnaissance obtenus à l'aide du jeu de règles après apprentissage. Les taux sont certes significatifs du bon comportement du modèle obtenu (80,36 %), mais il est tout de même important de noter que les réglages du FRC n'ont pas été choisis pour fournir des taux optimaux. L'objectif a été de définir les règles principales du

système. Le manque d'échantillons pour certaines classes de délais explique les faibles taux obtenus pour les délais intermédiaires et non-tolérés (surtout pour les délais non tolérés).

Lot de données	5M	6M	7M	Total
DP	228	233	231	692
	98,70%	97,40%	97,00%	97,70%
DI	86	62	50	198
	37,0%	38,70%	32,00%	34,80%
DNT	0	19	33	52
		31,60%	18,20%	23,10%
Total	80,89%	83,12%	79,30%	80,36%

Tableau 4.2. Taux de reconnaissance

#### 4.3.2.5. Règles floues et modèle obtenus

L'obtention du jeu de règles se fait en analysant la matrice obtenue lors de la phase itérative. Le jeu de règles obtenues après apprentissage sur le lot de données est illustré dans le tableau 4.3. On dit qu'une règle est « sûre » si son CF est proche de 1, tandis que les règles associées à un CF faible correspondent à des zones d'ambiguïté du modèle.

N° de Règle	W1	W2	Délai	CF
0	P	P	DP	0,863131
1	M	P	DP	0,98942
2	G	P	DP	0,999997
3	P	M	DI	0,467206
4	M	M	DP	0,923221
5	G	M	DP	0,999997
6	P	G	DNT	0,955389
7	M	G	DI	0,956875
8	G	G	DP	0,989456

Tableau 4.3. Règles générées

La règle 2 s'interprète par exemple de la façon suivante :

SI l'entrée  $w_1$  est GRAND (G) ET l'entrée  $w_2$  est PETIT (P) ALORS la sortie Délai est PARFAIT (DP) avec un coefficient de confiance égal à 0,999997.

On peut donc en déduire que si l'on veut un délai « Parfait » en sortie du réseau, il est possible de choisir un  $w_1$  GRAND (G) et un  $w_2$  PETIT (P).

D'autre part, la règle 3 s'interprète de la façon suivante :

SI l'entrée  $w_1$  est PETIT (P) ET l'entrée  $w_2$  est GRAND (G) ALORS la sortie Délai est Intermédiaire (DI) avec un coefficient de confiance égal à 0,467206.

Certes, la règle 3 a un faible coefficient de confiance mais cela n'a aucune incidence dans notre cas car cette règle ne s'avère pas utile pour notre stratégie de commande (paragraphe 4.5).

On conclut après cette étape d'identification qu'il est possible de déterminer une configuration de poids permettant d'avoir un Délai Parfait (DP) et cela avec un coefficient de confiance égal à 0,999997 grâce à la règle 2 du tableau 4.3. Ainsi, on pourra mettre en place notre stratégie de reconfiguration en se basant sur cette règle. Cette stratégie de reconfiguration des poids est détaillée dans le paragraphe 4.5.

#### 4.4. La commande floue

La commande floue a été introduite par (Zadeh, 1973) et elle constitue un champ d'applications pour la théorie des ensembles flous. Plusieurs exemples ont été traités dans divers domaines dans le but de montrer l'intérêt de cette commande floue (par exemple (Willaeys *et al.*, 1977) ont traité le cas d'une régulation de vitesse d'un moteur). La commande floue a suscité un intérêt particulier au Japon depuis le début des années 80.

La commande floue sert à traiter des problèmes de commande de process comme c'est le cas en automatique classique. Elle se base sur la connaissance d'experts qui ont une bonne connaissance du process à commander et elle peut dans certains cas être utilisée avec une commande classique. Ce type de commande est intéressant lorsque le process à commander n'est pas identifiable de manière mathématique comme dans le cas d'un système très complexe ou dans le cas où les connaissances sont exprimées de manière non numérique.

Selon (Bouchon Meunier, 1995), la commande floue est intéressante pour les raisons suivantes :

- La simplicité de réalisation, sa flexibilité et sa facilité d'adaptation aux conditions de fonctionnement du process ou à une utilisation particulière. En général, un petit nombre de règle est suffisant pour décrire le système.
- La facilité pour réaliser la synthèse d'avis de plusieurs experts.
- La possibilité de coordonner plusieurs objectifs.
- La robustesse dans la mesure où elle résiste bien aux perturbations pouvant affecter le processus.
- Et selon les utilisateurs, elle permet des économies d'énergie.

La commande floue est généralement appliquée à des process difficiles à modéliser avec des méthodes d'identification classique comme dans le cas traité dans cette thèse. La principale difficulté tient dans le caractère non-linéaire du système. L'inconvénient majeur de la commande floue est qu'il n'existe pas une procédure standard de réglage. L'autre problème majeur dans ce type de contrôleur est qu'il n'y a pas beaucoup de résultats permettant de garantir la stabilité et la robustesse de tels systèmes.

Dans (Tong, 1977), l'analyse des systèmes de commande flous se sont basés sur l'étude de la matrice correspondant au jeu de règles. Dans (Kickert et Mamdani, 1978), cette analyse de stabilité s'est basée sur le critère du cercle de stabilité. Dans (Garcia-Cerezo *et al.*, 1994), l'objectif a été de fournir des outils permettant de mettre en relation directe les caractéristiques du contrôleur flou avec les performances dynamiques du système en boucle fermée et cela dans le but de concevoir des contrôleurs flous stables et robustes. Dans les travaux récents concernant l'analyse de stabilité dans le cas de contrôleurs flous de type Mamdani, on trouve l'approche de Lyapunov (Sugeno et Taniguchi, 2004) et l'approche algébrique (Andújar *et al.*, 2004).

L'intérêt de notre approche tient dans la possibilité de maintenir le délai constamment ou presque dans une zone acceptable tout en maximisant la bande passante offerte au trafic temps-réel. Nous proposons un contrôleur flou qui se base sur l'inverse du modèle du process. Des méthodes pour inverser les modèles flous existent (Babuška et Verbruggen, 1995). (Babuška et Verbruggen, 1996) a même noté que pour les systèmes qui varient dans le temps, le modèle flou peut être adapté et utilisé pour re-concevoir le contrôleur en ligne (figure 4.9). Dans notre cas, le modèle flou est figé du fait qu'un échantillon (utilisé pour la modélisation floue) pour une configuration de poids est obtenu en effectuant 30 simulations de 5 minutes qui est une opération qui ne peut pas être réalisée en ligne.

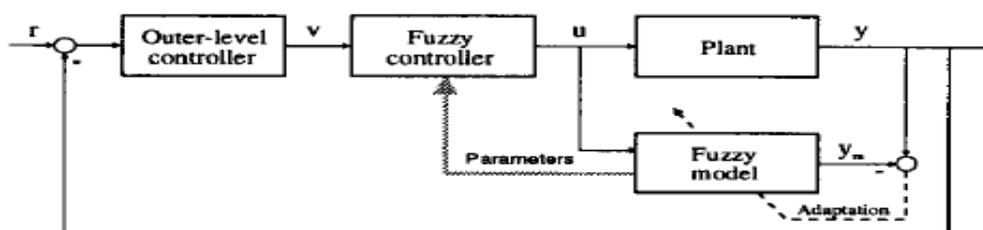


Figure 4.9. Le contrôleur flou est obtenu en inversant le modèle flou. Pour les systèmes qui varient dans le temps, le modèle peut être mis à jour dans la boucle (Babuška et Verbruggen, 1996)

#### 4.5. Synthèse de la commande floue pour la régulation du délai

Nous proposons de mettre en place un contrôleur flou pour stabiliser le comportement du système présenté dans le paragraphe 2.3.3 et étudié en utilisant le logiciel OPNET.

La stratégie de reconfiguration de poids proposée est la suivante :

- Quand le délai est à 100 % Non Toléré (càd supérieur à 7,644 ms : voir figure 4.10) : nous allons utiliser la règle n° 2 pour obtenir un Délai Parfait. Ainsi nous allons fixer  $w_1$  à 141 et  $w_2$  à 30.
- Dans le cas d'un délai inférieur ou égal à 7,644 ms, on se base sur la figure 4.10 :
  - Si  $\mu(DNT) = 0$  :
 
$$w_1 = w_1 + [-3.80167 + \mu(DP) * (-5 + 3.80167)]$$

$$\text{Et } w_2 = w_2 + [3.80167 + \mu(DP) * (5 - 3.80167)]$$
  - Si  $\mu(DNT) > 0$  :
 
$$w_1 = w_1 + [-3.80167 + \mu(DNT) * (5 + 3.80167)]$$

$$\text{Et } w_2 = w_2 + [3.80167 + \mu(DNT) * (-5 - 3.80167)]$$

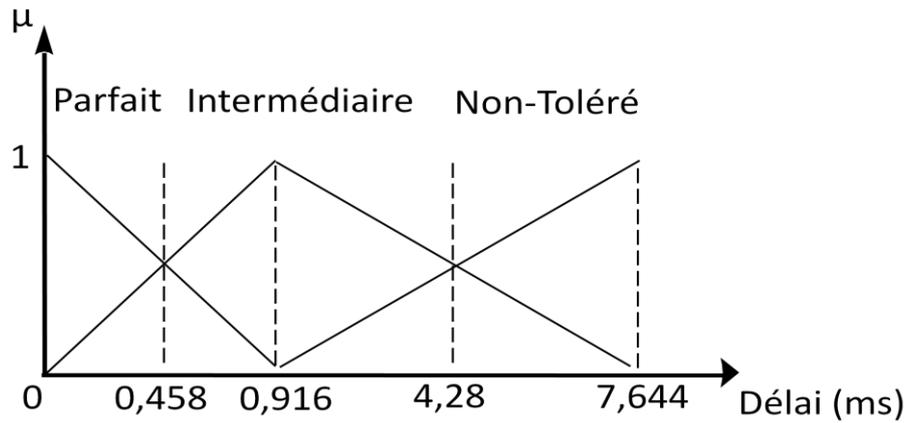


Figure 4.10.a. Fuzzification du délai

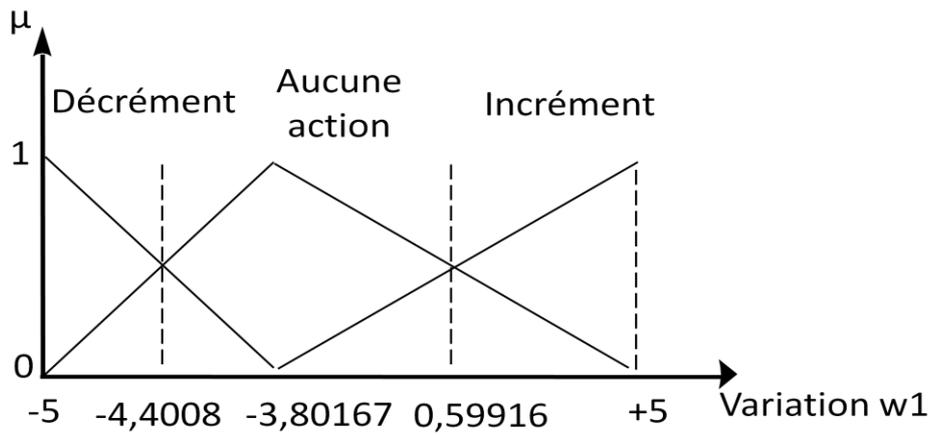
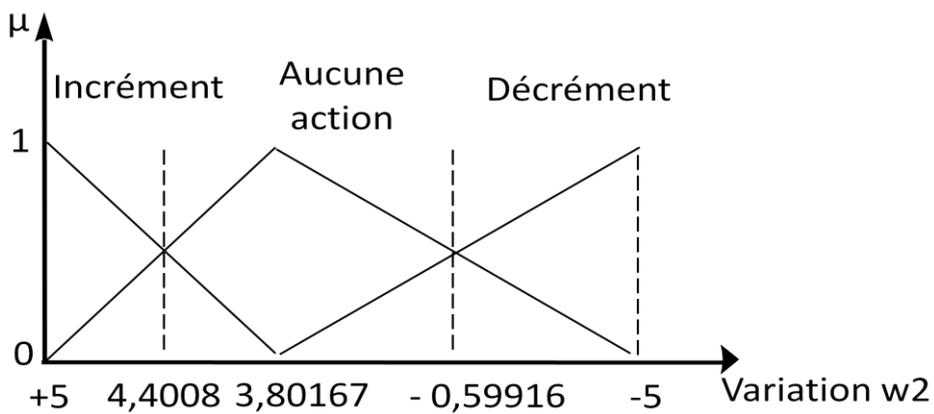
Figure 4.10.b. Fuzzification de la variation de  $w_1$ Figure 4.10.c. Fuzzification de la variation de  $w_2$ 

Figure 4.10. Commande floue dans le cas d'un délai inférieur à 7,644 ms

Pour conclure sur les variations de  $w_1$  et de  $w_2$  dans le cas d'un délai inférieur à 7,644 ms, on se base sur la figure 4.10 pour déduire la stratégie de reconfiguration de poids. Ainsi :

Exemple (figure 4.10) :

- Quand le délai est égal à 4,28 ms ( $\mu(DNT) = 0,5$ ) :

$$w_1 = w_1 + [-3.80167 + \mu(DNT) * (5 + 3.80167)] = w_1 + [0.599] = w_1 + 1$$

$$\text{Et } w_2 = w_2 + [3.80167 + \mu(DNT) * (-5 - 3.80167)] = w_2 + [-0.599] = w_2 - 1$$

## 4.6. Résultats des simulations

Dans cette partie, nous allons montrer les résultats des simulations qui illustrent l'intérêt de la commande floue dans le cas des applications temps-réel souple. Avant cela, nous allons présenter comment nous avons procédé pour ajouter le module d'adaptation dynamique en ligne des poids du WRR en fonction des délais subis par l'application temps-réel sur le logiciel OPNET (OPNET, 2005).

### 4.6.1. Adaptation en ligne des poids du WRR sur OPNET

L'un des objectifs de notre étude est de pouvoir régler en ligne les paramètres du WRR et cela pour pouvoir simuler le comportement de la stratégie de reconfiguration des poids présentée dans le paragraphe 4.5. Dans ce cadre, le modèle de commutateur Cisco disponible dans OPNET a été étendu de manière à prendre en compte 2 nouvelles fonctionnalités : la lecture de la variable de délai pour détecter un dépassement de délai et un réglage des poids du WRR pour la phase de reconfiguration du réseau.

#### 4.6.1.1. Récupération de la variable du délai du trafic temps-réel

Le but de cette partie est de récupérer le délai subi par un message traversant un port de sortie du commutateur et qui appartient à une classe de trafic donnée. Le logiciel OPNET (OPNET, 2005) fournit une sonde permettant de récolter ce délai. L'objectif est donc de pouvoir lire la valeur de ce délai au niveau d'un modèle de processus OPNET (OPNET, 2005) qui sera modifié pour être ensuite utilisé dans la phase de reconfiguration.

Afin de pouvoir utiliser la sonde qui nous intéresse (« Switch Output Port.Queueing Delay (sec) »), il a fallu ajouter un lien « stastic wire » et le ré-injecter au niveau du modèle standard d'un commutateur Ethernet comme le montre la figure 4.11.

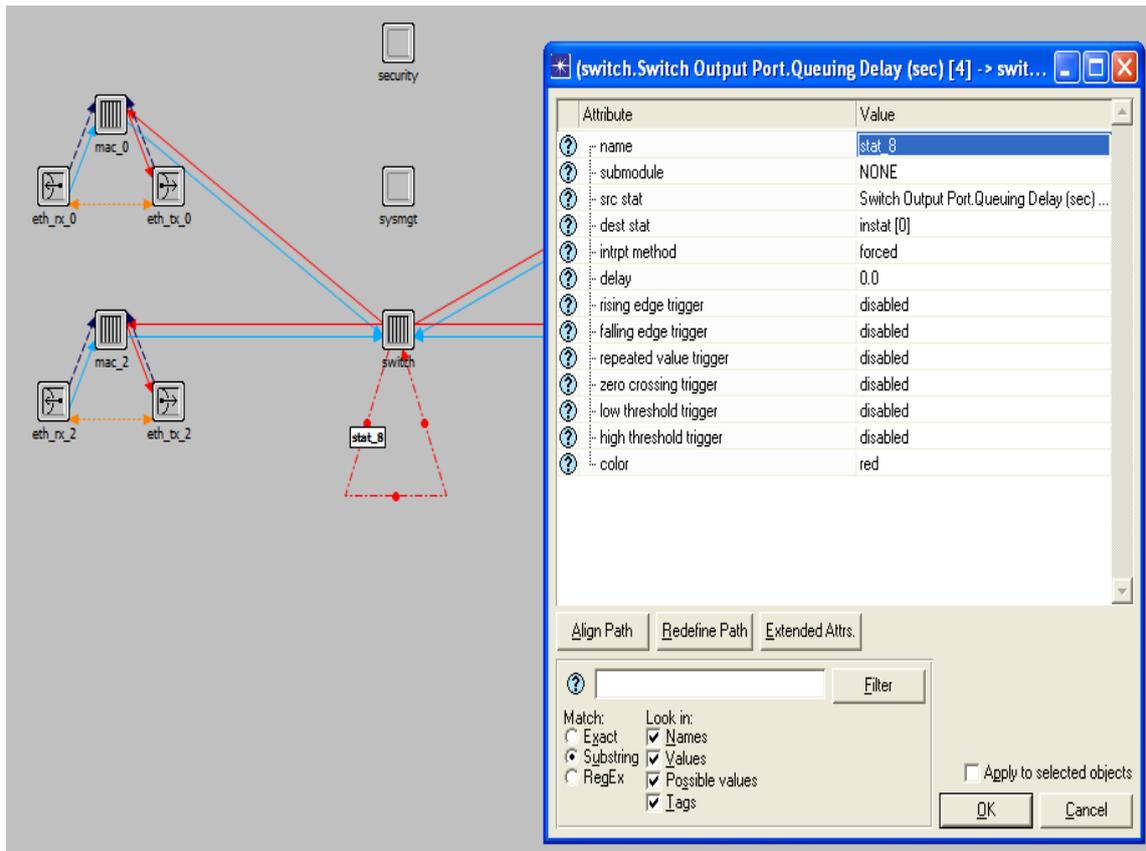


Figure 4.11. Ajout de la « statistic wire » permettant de récupérer la valeur du délai

La figure 4.11 montre les réglages à apporter au niveau du lien « statistic wire ».

#### 4.6.1.2. Adaptation en ligne des poids

Pour cela, nous nous sommes basés sur un modèle déjà existant sur le logiciel OPNET (OPNET, 2005) auquel nous avons ajouté la fonction d'ordonnancement dynamique. Le commutateur qui a servi de base est le Cisco C2924XL (OPNET, 2005).

Les process qui sont utilisés par le commutateur de base et qui ont dû être modifiés afin de permettre la mise en place de la classification de service sont le processus « bridge\_dispatch\_v2 » renommé « bridge\_dispatch\_v3 » et un de ces process fils : « bridge\_mac\_relay\_entity\_v2 » renommé « bridge\_mac\_relay\_entity\_v3 ».

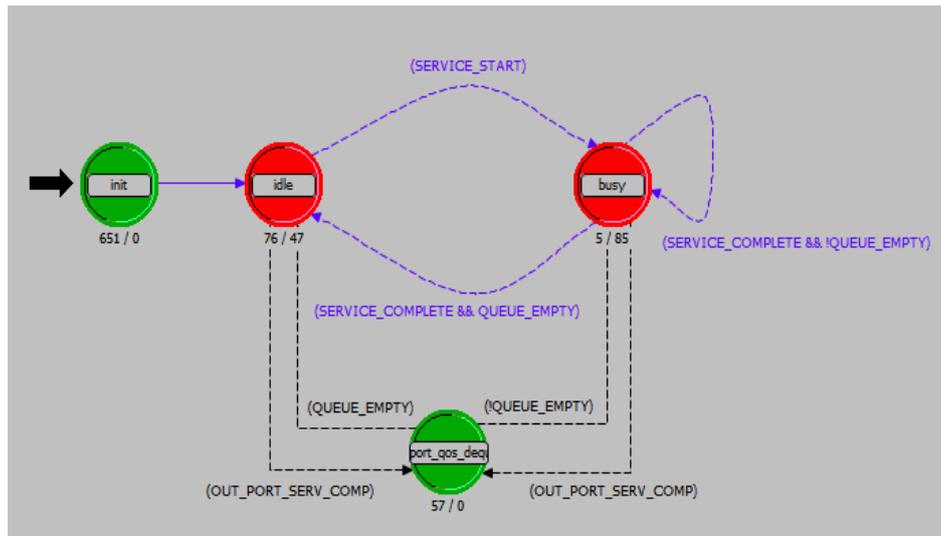


Figure 4.12. Process model bridge\_mac\_relay\_entity\_v3.pr.m

Comme le montre la figure 4.12, ce processus comporte 4 états :

- Init : Un état d'initialisation
- Idle : Un état de repos
- Busy : Un état actif
- Port\_qos\_dequeue : Etat qui permet d'extraire un paquet d'une file (lorsqu'un port devient prêt)

Le code de reconfiguration dynamique des poids a été implémenté dans l'état « idle ».

La figure 4.13 montre les changements apportés à l'interface de configuration du modèle de commutateur Cisco C2924XL d'OPNET où il est possible d'activer ou désactiver cette nouvelle fonctionnalité.

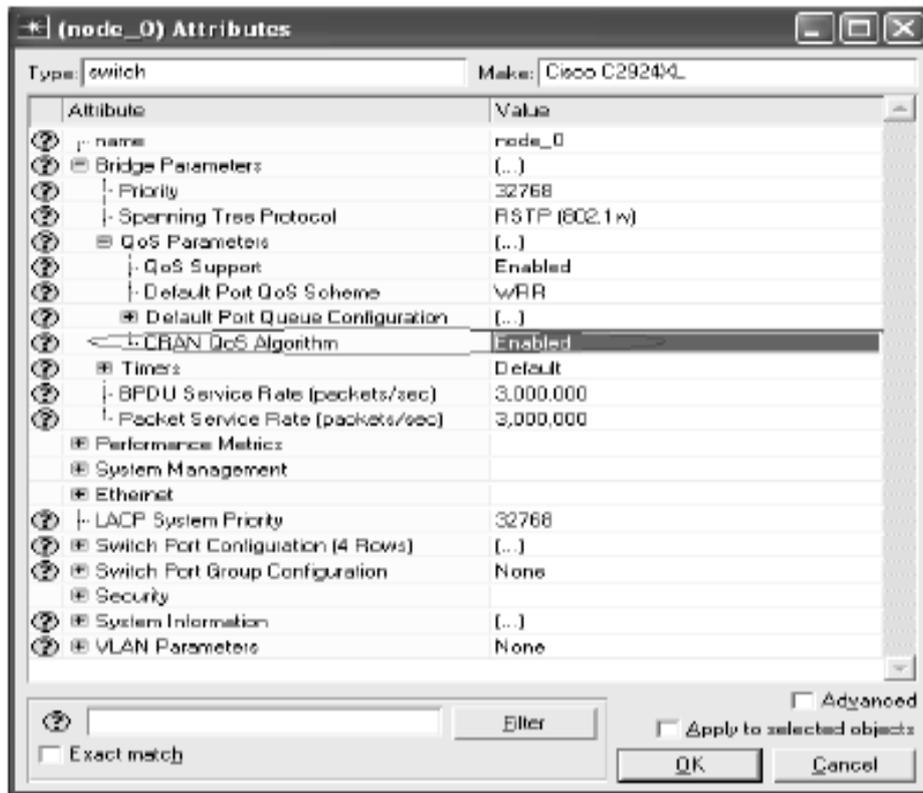


Figure 4.13. Commutateur permettant de mettre un WRR dynamique sur OPNET (Diouri *et al.*, 2010)

#### 4.6.2. Mise en place de la stratégie de reconfiguration dynamique des poids sur OPNET

Dans ce paragraphe, nous allons présenter les résultats des simulations effectuées sur le logiciel OPNET. Ces résultats ont été obtenus en appliquant la stratégie de reconfiguration dynamique des poids présentée dans le paragraphe 4.5. Le SCR simulé est le même que celui présentée dans les chapitres précédents (paragraphe 2.3.3). L'application des résultats précédents concernant la commande floue pour la régulation du délai nous a permis d'identifier les règles suivantes :

Si (*délai* == 0 s)  
 $w_1 = \max((w_1 - 5), 1)$ ;  
 $w_2 = \min((w_2 + 5), 255)$ ;

SinonSi (0 s < *délai* ≤ 0.0007644 s)  
 $w_1 = \max((w_1 - 4), 1)$ ;  
 $w_2 = \min((w_2 + 4), 255)$ ;

SinonSi (0.0007644 s < *délai* ≤ 0.0015288 s)  
 $w_1 = \max((w_1 - 3), 1)$ ;  
 $w_2 = \min((w_2 + 3), 255)$ ;

SinonSi (0.0015288 s < *délai* ≤ 0.0022932 s)  
 $w_1 = \max((w_1 - 2), 1)$ ;  
 $w_2 = \min((w_2 + 2), 255)$ ;

SinonSi (0.0022932 s < *délai* ≤ 0.0030576 s)  
 $w_1 = \max((w_1 - 1), 1)$ ;  
 $w_2 = \min((w_2 + 1), 255)$ ;

SinonSi (0.0030576 s < *délai* ≤ 0.003822 s)  
 $w_1 = w_1$  ;  
 $w_2 = w_2$  ;

SinonSi (0.003822 s < *délai* ≤ 0.0045864 s)  
 $w_1 = \min((w_1 + 1), 255)$  ;  
 $w_2 = \max((w_2 - 1), 1)$  ;

SinonSi (0.0045864 s < *délai* ≤ 0.0053508 s)  
 $w_1 = \min((w_1 + 2), 255)$  ;  
 $w_2 = \max((w_2 - 2), 1)$  ;

SinonSi (0.0053508 s < *délai* ≤ 0.0061152 s)  
 $w_1 = \min((w_1 + 3), 255)$  ;  
 $w_2 = \max((w_2 - 3), 1)$  ;

SinonSi (0.0061152 s < *délai* ≤ 0.0068796 s)  
 $w_1 = \min((w_1 + 4), 255)$  ;  
 $w_2 = \max((w_2 - 4), 1)$  ;

SinonSi (0.0068796 s < *délai* ≤ 0.0007644 s)  
 $w_1 = \min((w_1 + 5), 255)$  ;  
 $w_2 = \max((w_2 - 5), 1)$  ;

SinonSi (*délai* > 0.0007644 s)  
 $w_1 = 141$  ;  
 $w_2 = 30$  ;

FinSi

Ce sont ces règles qui ont été implantées pour cet exemple dans OPNET.

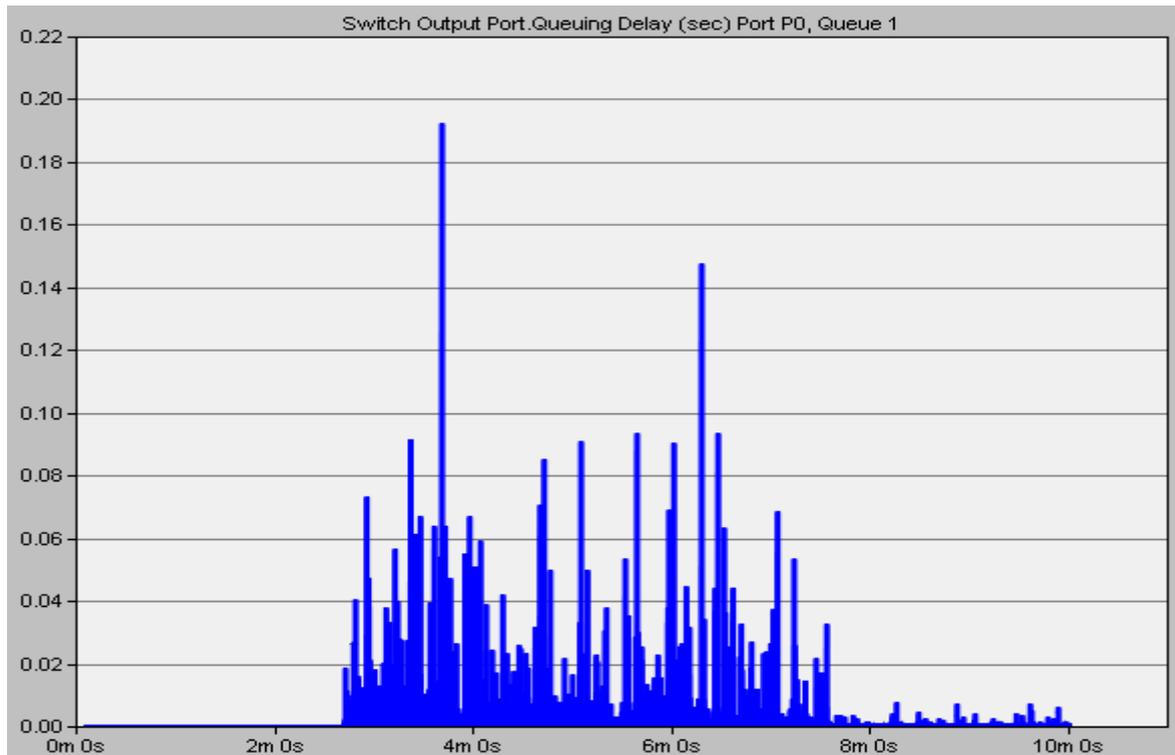


Figure 4.14. Variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du contrôleur vers le process

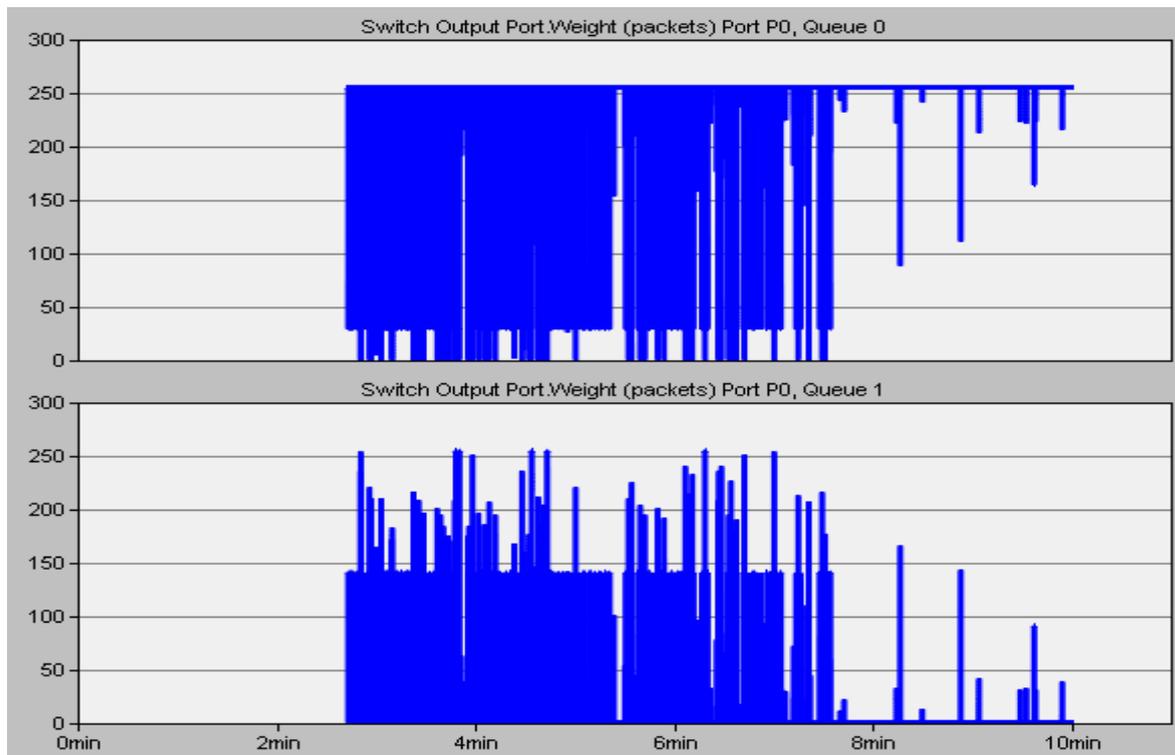


Figure 4.15. Variation des poids du commutateur 2 (figure 2.14) en fonction des délais de traversée de celui-ci par le flux allant du contrôleur vers le process

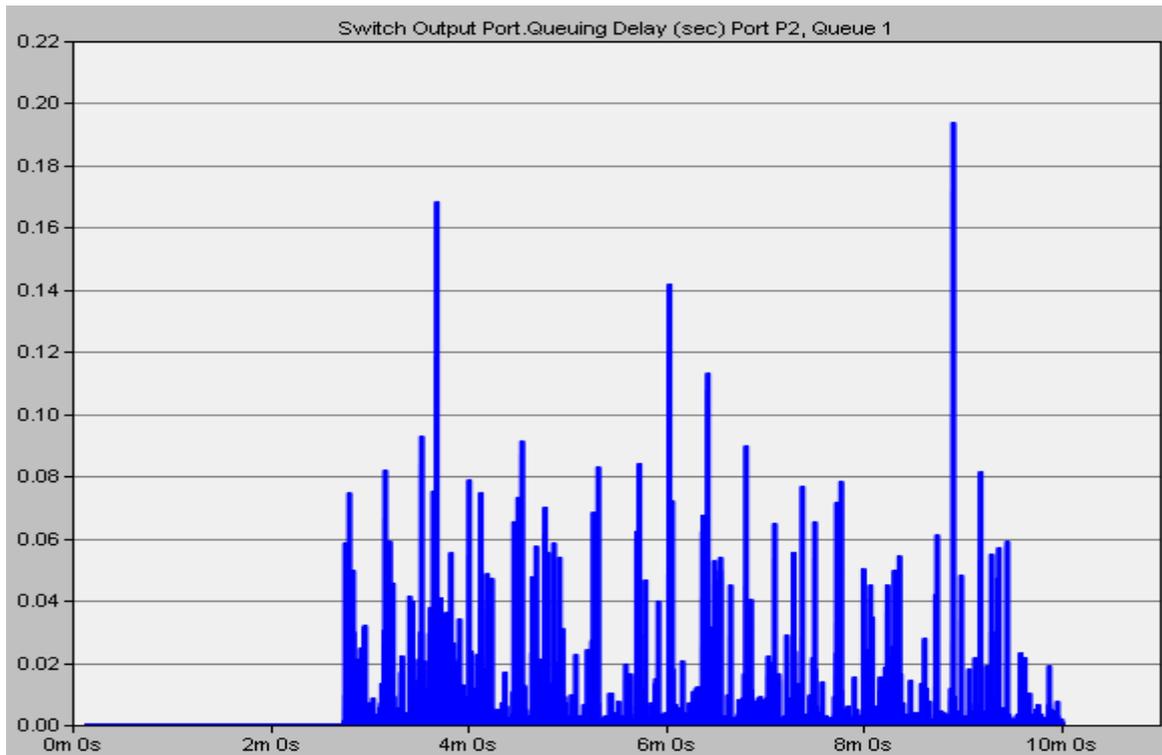


Figure 4.16. Variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du process vers le contrôleur

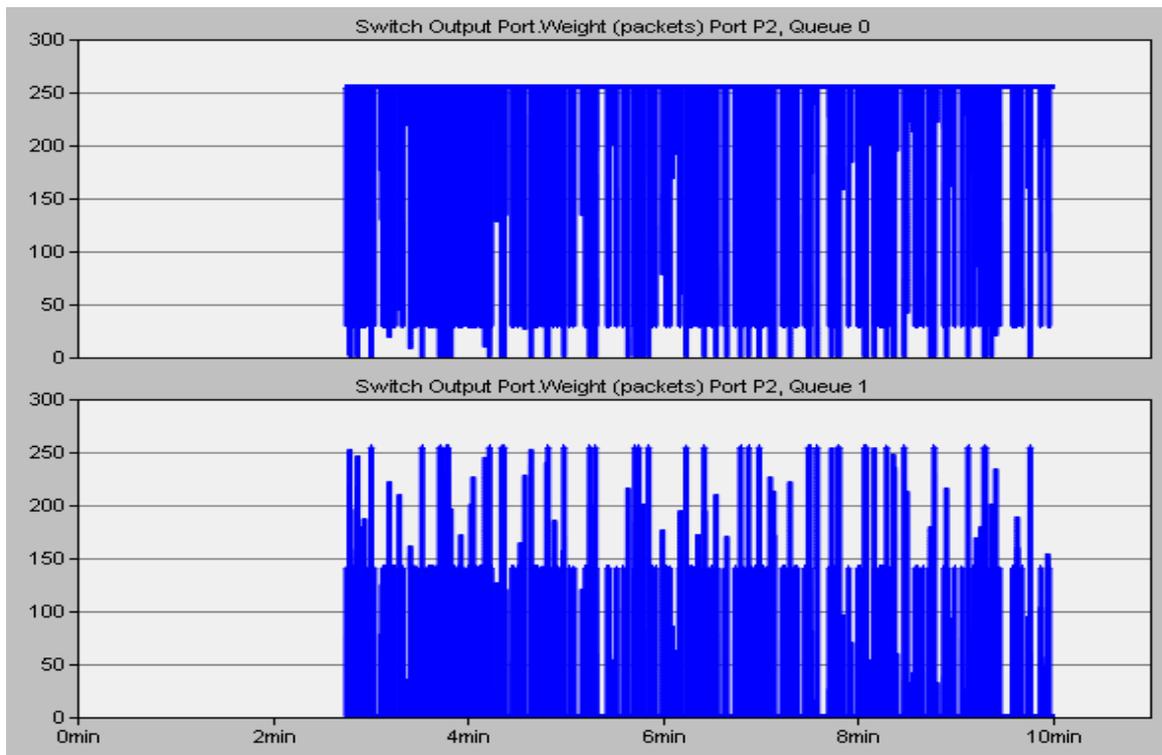


Figure 4.17. Variation des poids du commutateur 2 (figure 2.14) en fonction des délais de traversée de celui-ci par le flux allant du process vers le contrôleur

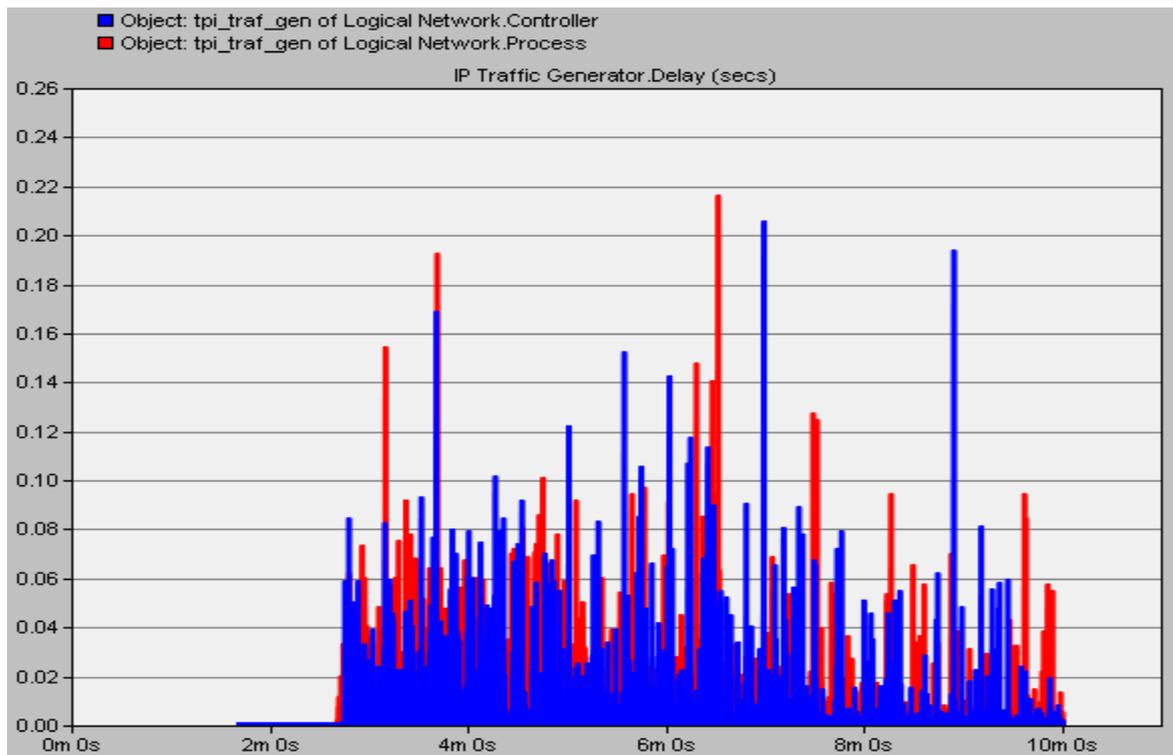


Figure 4.18. Délais subis aux niveaux du contrôleur et du process

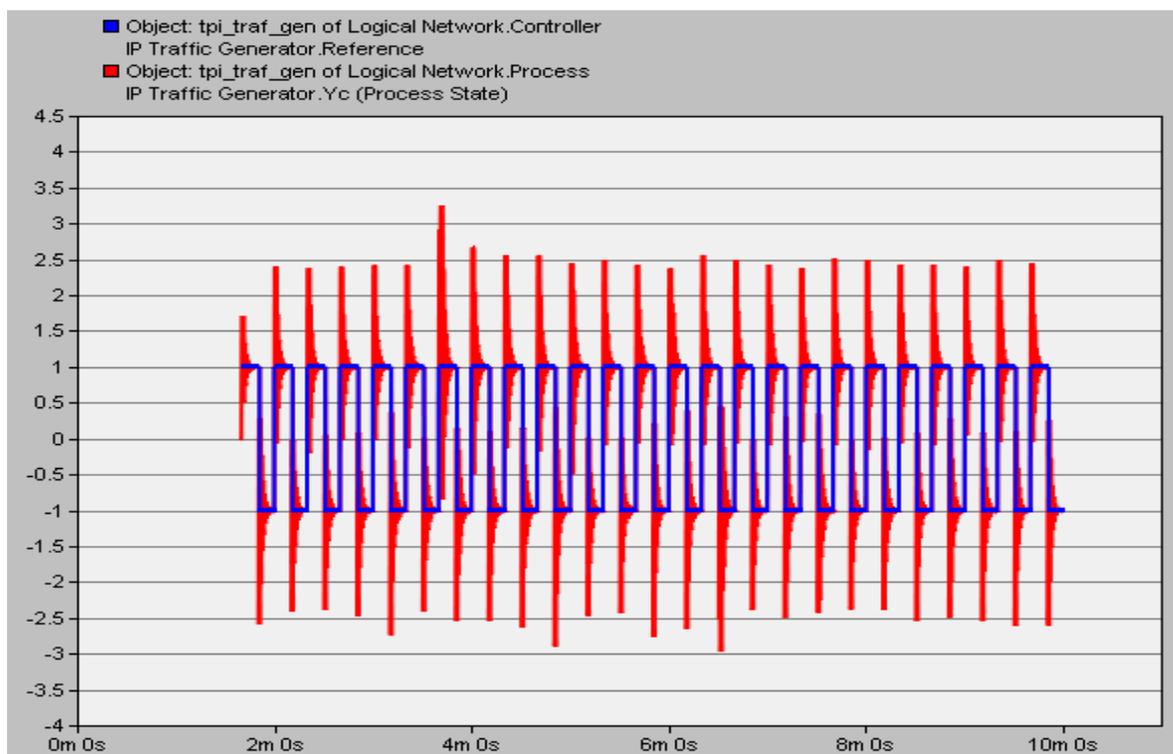


Figure 4.19. Sortie du système

Ici, on a choisi de présenter les délais de traversée ainsi que les variations des poids au niveau d'un commutateur uniquement (pour l'autre commutateur, c'est le même principe). Toutefois, nous allons également présenter les délais de bout en bout ainsi que la sortie du système pour montrer le comportement global du système.

Les figures 4.14 et 4.16 montrent respectivement la variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du contrôleur vers le process et la variation des délais de traversée du commutateur 2 (figure 2.14) subis par le flux allant du process vers le contrôleur.

Les graphes situés en haut des figures 4.15 et 4.17 illustrent les variations des poids des deux trafics de fond au niveau des deux ports de sortie du commutateur 2 : le port de sortie représenté dans la figure 4.15 concerne le flux du contrôleur vers le process et celui représenté dans la figure 4.17 concerne le flux du process vers le contrôleur. Les graphes situés en bas des figures 4.15 et 4.17 donnent les variations des poids des deux trafics temps-réel au niveau des deux ports de sortie du commutateur 2 : le port de sortie représenté dans la figure 4.15 concerne le flux du contrôleur vers le process et celui représenté dans la figure 4.17 concerne le flux du process vers le contrôleur. La figure 4.18 présente, quant à elle, les délais subis par le trafic temps-réel au niveau du contrôleur et du process. Enfin, la figure 4.19 montre le comportement de la sortie obtenu en appliquant la stratégie de reconfiguration dynamique des poids (paragraphe 4.5) au niveau des deux commutateurs.

Les résultats présentés dans les figures 4.14, 4.15, 4.16 et 4.17 montrent d'une part que les poids du WRR varient bien en fonction des délais du trafic temps-réel et cela pour chaque port de sortie du commutateur. Et d'autre part, on voit bien que la stratégie de reconfiguration des poids de l'ordonnanceur WRR permet de réduire le délai du trafic temps-réel à chaque fois que celui-ci dépasse le seuil de traversée du commutateur (ici 4,28 ms) de telle sorte à ce qu'il soit le plus rapidement possible inférieur à ce seuil. En effet, on remarque qu'à  $t = 3 \text{ min } 40 \text{ s}$  environ et à  $t = 6 \text{ min } 16 \text{ s}$  environ, il y a eu des délais plus grands que le seuil fixé ce qui a engendré des dépassements plus importants au niveau de la sortie du système (figure 4.19). Toutefois, la stratégie de reconfiguration des poids a permis de réduire rapidement ces délais et par conséquent cela n'a pas eu un impact grave sur la commande de notre système.

La figure 4.18 montre quant à elle que les délais du contrôleur vers le process ainsi que ceux dans le sens inverse dépassent rarement les 10 ms (seuil à ne pas dépasser de bout en bout), ce qui explique le comportement de la sortie du système obtenu. En effet, on remarque d'après la figure 4.19 que notre système reste stable durant toute la simulation.

En comparant ces résultats de simulations avec ceux obtenus en se basant sur le calcul réseau (paragraphe 3.7.2), on constate que le comportement de la sortie du système est légèrement meilleur lorsqu'on utilise le calcul réseau (en termes de dépassements de la consigne). On

remarque aussi d'après les résultats du chapitre 4 que le seuil du délai est parfois dépassé alors qu'en utilisant le calcul réseau cela n'arrive jamais.

On conclut à travers ce paragraphe que le fait d'appliquer cette commande nous a permis de réduire le pessimisme engendré par le résultat du calcul réseau qui interdit l'occurrence du dépassement du seuil et qui se place par conséquent dans l'hypothèse du pire cas. Cela apparaît à travers les figures 4.15 et 4.17 qui montrent bien que les poids du trafic de fond peuvent atteindre des valeurs allant jusqu'à 255.

## 4.7. Conclusion

Le chapitre précédent nous avait permis de conclure que le recours systématique à des théories comme le calcul réseau peut introduire un conservatisme, puisque l'étude du pire cas ne fait apparaître que des situations peu fréquentes. Dans ce dernier chapitre, nous avons cherché à caractériser davantage le comportement du réseau au travers de l'évolution du délai. Ce chapitre ne s'inscrit pas en porte à faux avec le précédent, il s'agit simplement d'une extension. En effet, on pourra se référer à ce chapitre lorsque l'application n'impose pas de contraintes strictes. On pourra également se référer à ces deux chapitres lorsqu'on s'intéressera à une application mixant exigences strictes et souples : tant qu'un seuil n'est pas atteint, on travaille de manière souple tandis que tout dépassement de ce seuil se caractérisera par un travail déterministe. On peut voir ici une analogie avec les algorithmes d'ordonnancement hybrides associant par exemple trois files WRR avec une file gérée par la priorité stricte.

Dans ce chapitre, nous avons commencé par identifier le comportement du réseau et plus particulièrement celui des délais du trafic temps-réel en fonction de la configuration des poids de l'ordonnanceur WRR. Cette identification a été réalisée en se basant sur la logique floue. A partir du modèle identifié (tableau 4.3), nous avons synthétisé une commande floue pour reconfigurer en ligne les poids du WRR. Il est important de garder à l'esprit que ce chapitre développe une méthodologie plus que des résultats analytiques. En effet, les résultats numériques qui y sont présentés ne valent que pour les hypothèses (le domaine de validité) de l'identification/classification ci-menée. On rappellera notamment les débits, les types de charges, le commutateur ou encore la taille des mémoires tampons associées à chaque file de chaque port de chaque commutateur.

La commande floue qui a été présentée dans ce chapitre permet de satisfaire les contraintes temps-réel souples de l'application tout en évitant au mieux les situations d'insuffisance voire de famine pour le trafic de fond. En effet, cette commande ne commence à privilégier le trafic temps-réel en termes de bande passante que dans le cas où l'on s'approche du seuil fixé par l'application (selon l'algorithme quand les délais sont supérieurs à 3,822 ms). D'un autre côté, quand le délai est supérieur à 7,644 ms, nous appliquons la règle n° 2 du tableau 4.3 et cela

dans le but d'avoir des délais inférieurs à 0,458 ms qui correspondent à la classe Délai Parfait (paragraphe 4.3.1.2).

	Trafic concurrent au flux de commande	Trafic concurrent au flux de mesure
<b>Commutateur 1</b>	9,993 Mb/s	9,991 Mb/s
<b>Commutateur 2</b>	9,994 Mb/s	9,987 Mb/s

Tableau 4.4. Bandes passantes moyennes offertes aux trafics non contraints temporellement par chaque commutateur (figure 2.14) en appliquant la logique floue

	Trafic concurrent au flux de commande	Trafic concurrent au flux de mesure
<b>Commutateur 1</b>	9,734 Mb/s	9,734 Mb/s
<b>Commutateur 2</b>	9,734 Mb/s	9,734 Mb/s

Tableau 4.5. Bandes passantes offertes aux trafics non contraints temporellement par chaque commutateur (figure 2.14) en appliquant le calcul réseau

Le tableau 4.4 montre les bandes passantes moyennes offertes aux trafics non contraints temporellement obtenus en utilisant la stratégie de reconfiguration dynamique floue tandis que le tableau 4.5 montre les bandes passantes offertes aux trafics non contraints temporellement obtenus par calcul réseau. Les tableaux 4.4 et 4.5 illustrent que le recours à logique floue a permis d'offrir des réservations de bandes passantes aux trafics non contraints temporellement légèrement plus importantes en comparant avec celles obtenues à l'aide du calcul réseau, la différence étant de 250 Kb/s sur des liens de 10 Mb/s.

D'un autre côté, les figures 3.15 et 4.19 montrent que le système reste stable dans les deux cas (commande floue et configuration basée sur le calcul réseau). La comparaison des deux figures 3.15 et 4.19 montre aussi que, dans le cas de la stratégie de reconfiguration dynamique basée sur la logique floue, les dépassements sont légèrement plus importants avec une différence maximale de l'ordre de 40 % par rapport au cas d'une configuration basée sur le calcul réseau. Cette commande floue permet donc de réduire le conservatisme engendré par le

calcul réseau qui repose sur une configuration statique en considérant le pire cas et qui ne prend pas en compte l'évolution de la charge. Toutefois, ce pessimisme dû au calcul réseau a été réduit avec l'utilisation de la courbe de service en escalier non-uniforme, qui a été proposée dans cette thèse, ce qui explique la faible différence entre les réservations de bandes passantes offertes aux trafics non contraints temporellement dans les deux cas.

Cependant, il est à noter que pour appliquer la méthodologie présentée dans le chapitre 4, il faut une bonne connaissance du système étudié qui dans notre cas a nécessité de faire de nombreuses simulations qui sont gourmandes en termes de temps.



# Conclusion générale

Cette thèse s'inscrit dans le contexte des Systèmes Contrôlés en Réseaux (SCR) qui sont des systèmes de contrôle/commande distribués à travers un réseau. La problématique de ces SCR réside dans le fait que les délais induits par le réseau peuvent réduire les performances du système et même conduire à l'instabilité de celui-ci. Dans nos travaux, nous avons considéré que le réseau utilisé par les SCR peut être partagé avec d'autres applications qui viennent occuper le réseau et cela de façon aléatoire.

Dans cette thèse, nous avons étudié les SCR s'appuyant sur des architectures Ethernet commutées car ce réseau est de plus en plus utilisé dans les systèmes industriels et les systèmes embarqués et que celui-ci offre beaucoup d'avantages en termes de débits, de réduction des domaines de collisions, d'interopérabilité, etc. L'inconvénient majeur du réseau dans les SCR est que le délai engendré par celui-ci reste difficile à modéliser ce qui complique sa prise en compte dans la conception du système.

L'objectif de cette thèse a été d'optimiser, en fonction des trafics et des contraintes applicatives, les ressources de communication à allouer pour chaque type de trafic : on distingue alors trafic temps-réel et trafic non contraint temporellement. Au delà de la problématique de déterminisme (à savoir le respect des exigences temporelles du trafic temps-réel), nous nous sommes intéressés dans cette thèse à la prise en compte du trafic de fond, non pas seulement comme entité perturbatrice, mais également comme trafic réclamant à part entière un service équitable. Après une étude bibliographique nous avons choisi d'étudier pour gérer les ressources de communication l'ordonnanceur Weighted Round Robin (WRR) car il permet de contrôler le partage de la bande passante relativement aux types de trafics et car il est implanté dans les commutateurs Ethernet du commerce.

Dans cette thèse, nous avons distingué deux classes d'applications, celle avec des contraintes temps-réel strictes (chapitre 3) et celle ayant des contraintes temps-réel souples (chapitre 4). Pour chaque type d'applications temps-réel, il a fallu dans un premier temps définir un modèle du délai qui prend en compte les paramètres du réseau et plus particulièrement les configurations des poids des ordonnanceurs WRR.

Pour les applications à contraintes temps-réel strictes, la théorie du calcul réseau a été utilisée pour définir des bornes de délais. Cette thèse propose une nouvelle courbe de service « courbe de service en escalier non uniforme » pour modéliser le comportement du WRR et montre que cette courbe est moins pessimiste qu'une courbe de service de type *rate-latency* car les résultats obtenus avec cette nouvelle courbe de service sont proches de ceux obtenus par un simulateur réseau. Cette identification du délai maximal a servi ensuite à identifier la configuration hors-ligne des poids qui permette de garantir les contraintes temps-réel strictes

de l'application tout en maximisant la bande passante offerte aux autres flux. La simulation de réseau montre que les délais maxima calculés/obtenus par le calcul réseau correspondent potentiellement à des pires cas réels mais rarement atteints et que l'on surprotège majoritairement le trafic temps-réel au dépend des autres. L'idée a été de repenser l'approche de configuration des poids du WRR désormais en ligne mais pour des applications temps-réel souple et cela pour augmenter la part de la bande passante allouée aux trafics non contraints temporellement.

En ce qui concerne les applications ayant des contraintes temps-réel souples, l'identification du modèle du délai a été faite en utilisant la logique floue. Celle-ci a permis d'établir des règles d'inférences qui mettent en relations les entrées sur lesquelles il est possible d'agir (à savoir les poids du WRR) avec la sortie (à savoir le délai dans notre cas). Ce modèle du délai a servi à mettre en place une commande de réseau en ligne qui permette de garder le délai en dessous du seuil fixé par l'application temps-réel tout en garantissant le maximum de bande passante aux flux non contraints temporellement.

La comparaison des deux méthodes (configuration basée sur le calcul réseau et commande floue) montre que dans les deux cas le système reste globalement stable et que le comportement de la sortie est légèrement amélioré dans le cas d'une configuration de poids basée sur le calcul réseau. La comparaison entre ces deux méthodes a permis aussi de constater que la bande passante allouée est légèrement supérieure dans le cas d'une commande floue.

Ainsi, la commande floue a permis de réduire le conservatisme engendré par le calcul réseau qui se base sur une configuration statique en considérant le pire cas et qui ne prend pas en compte l'évolution de la charge. Toutefois, ce pessimisme dû au calcul réseau a été réduit avec l'utilisation de la courbe de service en escalier non-uniforme, qui a été proposée dans cette thèse, ce qui explique la faible différence entre les réservations de bandes passantes offertes aux trafics non contraints temporellement dans les deux cas.

Finalement, ce travail de thèse a aussi nécessité de développer de nouvelles fonctionnalités dans les outils de simulation OPNET et TrueTime qui pourront servir à la communauté scientifique qui travaille notamment sur les SCR.

## Perspectives

Ce travail est une première étape dans la commande de réseau pour les SCR. Les perspectives de cette recherche sont nombreuses et peuvent se décrire selon trois types.

Le premier concerne le cas des applications ayant des contraintes temps-réel souples où il serait judicieux de mettre en place un contrôleur flou qui régule le délai de bout en bout de manière centralisée et cela au lieu de procéder par une série de contrôleurs flous distribués

aux niveaux de chaque ordonnanceur WRR qui ne prennent en considération que le délai de traversée du commutateur où est implanté le WRR. En effet, une comparaison entre les résultats obtenus par une telle régulation centralisée et ceux obtenus par une régulation décentralisée comme celle présentée dans le chapitre 4 pourrait s'avérer intéressante.

Le second type de recherche est relatif aux interactions entre des méthodes de contrôle/commande de niveau applicatif et nos méthodes de commande de réseau. Il serait intéressant d'étudier comment appliquer de manière intégrée ces deux niveaux de méthodes et de répondre à des questions relatives au choix entre faire de la commande du système et faire de la commande de réseau pour compenser ou agir sur le délai induit par le réseau. D'autre part, dans cette thèse, la commande de réseau est enclenchée selon l'évolution du délai. Il serait intéressant de voir si d'autres informations du réseau pourraient être utilisées pour enclencher une action sur le réseau comme la gigue, le niveau de remplissage des buffers ou si des informations applicatives (temps de réponse, dépassement, etc.) ne pourraient pas aussi être prises en compte. En effet, est-il nécessaire de donner plus de bande passante au trafic temps-réel si l'application temps-réel fonctionne globalement correctement ?

La méthode de commande de réseau pourrait aussi servir dans le diagnostic global du SCR pour détecter par exemple une dérive de délai induit par le réseau mais aussi pour séparer des dysfonctionnements du SCR (réseau, système de commande de l'application, capteurs, actionneurs, process industriel, etc.) en analysant l'efficacité de la commande de réseau sur l'état du SCR. Un impact nul pourrait permettre de conclure que le problème n'est pas forcément lié au réseau mais à d'autres composants du SCR.

Le troisième type de recherche pourrait concerner la résolution des problématiques de calcul de bout en bout dans des réseaux fortement cycliques. En effet, dans ces réseaux, il reste délicat d'établir un majorant du délai du fait de la corrélation des flux. Plus généralement, cela pose le problème de l'identification de la part de délai induite par tel ou tel flux concurrent. Du point de vue du calcul réseau, cet axe reste aujourd'hui un point ouvert.



# Liste des publications

## Conférences internationales avec comité de lecture

**Diouri I.**, J.-P. Georges et E. Rondeau (2006). Accommodation of the service offered by the network for networked control systems. 2nd Workshop on Networked Control System and Fault Tolerant Control, Rende, Italie.

**Diouri I.**, J.-P. Georges et E. Rondeau (2007). Accommodation of delays for networked control systems using classification of service. ICNSC'07, 2007 IEEE International Conference on Networking, Sensing and Control, Londres, Royaume-Uni.

**Diouri I.**, J.-P. Georges et E. Rondeau (2007). Adaptation of scheduling policy parameters for controlling networked systems. 3rd International Workshop on Networked Control Systems : Tolerant to faults, Nancy, France.

**Diouri I.**, C. Berbra, J.-P. Georges, S. Gentil et E. Rondeau (2008). Evaluation of a switched Ethernet network for the control of a quadrotor. MED'08, 16th Mediterranean Conference on Control and Automation, Ajaccio, Corsica, France.

## Conférence nationale avec comité de lecture

**Diouri I.**, V. Bombardier, J.-P. Georges et E. Rondeau (2010). Classification des délais pour les Systèmes Contrôlés en Réseau à ordonnancement WRR. CIFA 2010, 6ème Conférence Internationale Francophone d'Automatique, Nancy, France.

## Rapport de contrat

Slacky S., V. Palacka, M. Horna, R. Minarik, E. Rondeau, J.-P. Georges, **I. Diouri** (2006). Livrable 7-3 : Network performance evaluation. Projet STREP NeCST.



# Bibliographie

Abate A., M. Chen et S. Sastry (2006). Analysis of an implementable application layer scheme for flow control over wireless networks. In *Mathematical Theory of Networks and Systems*, pages 701–708, Kyoto, Japan.

AFNOR (1990). Bus FIP pour échange d'informations entre transmetteurs, actionneurs et automates - Couche liaison de données. NF C 46-603.

Altman E., T. Basar et R. Srikant (1999). Congestion control as a stochastic control problem with action delays. *Automatica*, vol. 35, pages 1937-1950.

Alcala R., J. Alcala-Fdez, F. Herrera et J. Otero (2007). Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation. *International Journal of Approximate Reasoning* 44, pages 45–64.

Alves M., E. Tovar et F. Vasques (2000). Ethernet goes real-time : a survey on research and technological developments. Technical Report HURRAY-TR-2K01. Groupe de recherche IPP-HURRAY, Polytechnic Institute of Porto (ISEP-IPP).

Andersson M., D. Henriksson et A. Cervin (2005). Truetime 1.3 – reference manual. Department of Automatic Control, Lund Institute of Technology.

Andújar J. M., J. M. Bravo et A. Peregrín (2004). Stability analysis and synthesis of multivariable fuzzy systems using interval arithmetic. *Fuzzy Sets and Systems*, vol. 148, pages 337-353.

Aubrun C., B. Brahimi, J.P. Georges, G. Juanole, G. Mouney, X.H. Nguyen et E. Rondeau (2010). « QoS-Aware Dynamic Network QoS Adaptation » : chapitre 3 (pages 105-145) du livre —*Co-design Approaches for Dependable Networked Control Systems*», C. Aubrun, D. Simon, Y.Q. Song, ISBN 978-1-84821-176-6, Wiley-ISTE, 2010.

Azimi-Sadjadi B. (2003). Stability of networked control systems in the presence of packet losses. In *Proceedings of the conference on decision and control, Hawaii, USA*, pages 676–681.

Babuška R. et H.B. Verbruggen (1995). Identification of composite linear models via fuzzy clustering. In *Proceedings European Control Conference, Rome, Italy*, pages 1207-1212.

Babuška R. et H.B. Verbruggen (1996). An overview of fuzzy modeling for control. In *Control Eng. Practice*, vol. 4, no. 11, pages 1593-1606.

Bauer P., M. Schitui, C. Lorand et K. Premaratne (2001). Total Delay Compensation in LAN Control Systems and Implications for Scheduling. In *Proceeding of the American Control Conference*, pages 4300-4305.

Bennett J. C. R. et H. Zhang (1996). WF2Q : Worst-case Fair weighted Fair Queueing. In *Proceedings of the IEEE INFOCOM'96, San Francisco, CA*.

Ben Gaid M. E. M., A. Cela et Y. Hamam (2006). Optimal Integrated Control and Scheduling of Networked Control Systems with Communication Constraints: Application to a Car Suspension System. *IEEE Transactions on Control Systems Technology* 14, 4, pages 776 – 787.

Berbra C., S. Gentil, S. Lesecq et J.-M. Thiriet (2007). Co-design for a safe networked control dc motor. In 3rd International NeCST IFAC Workshop on Networked Control Systems Tolerant to Faults (NECST'07), Nancy, France.

Berbra C., S. Lesecq, S. Gentil et J.-M. Thiriet (2008). Co-design of a safe network control quadrotor. 17th IFAC World Congress, pages 5506-5511.

Berbra C., S. Gentil, S. Lesecq et D. Simon (2010). « Implementation : Control and Diagnosis for Unmanned Aerial Vehicle » : chapitre 7 (pages 267-303) du livre —Co-design Approaches for Dependable Networked Control Systems», C. Aubrun, D. Simon, Y.Q. Song, ISBN 978-1-84821-176-6, Wiley-ISTE, 2010.

Berthold M.R. (2003). Mixed fuzzy rule formation. *Int. Journal of Fuzzy Sets and Systems*, vol. 32, pages 67-84.

Blake S., D. Black, M. Carlsson, E. Davies, Z. Wang et W. Weiss (1998). An architecture for differentiated services, RFC 2475.

Bombardier V. et E. Schmitt (2010). Fuzzy rule classifier: Capability for generalization in wood color recognition. *Engineering Applications of Artificial Intelligence*, vol. 23, pages 978-988.

Bouchon-Meunier B. (1995). *La logique Floue et ses applications*. Addison Wesley France, ISBN: 2-87908-073-8, Paris, France.

Boughanmi N., Y.Q. Song et E. Rondeau (2009). Online adaptation of the IEEE 802.15.4 parameters for wireless networked control systems. IFAC FeT2009, Ansan, Korea.

Bouillard A. et E. Thierry (2008). An algorithmic toolbox for network calculus. *Discrete Event Dynamic Systems*, vol. 18, no. 1, pp. 3-49.

Bouillard A., Gaujal B., Lagrange S. et E. Thierry (2008). Optimal routing for end-to-end guarantees using Network Calculus. *Performance Evaluation*, vol. 65, pages 883-906.

Boukas E. K. et Z. K. Liu. (2001). Robust  $H_\infty$  control of discrete time Markovian jump linear systems with mode-dependent time delays. *IEEE Transactions on Automatic Control*, 46(12), 1918-1924.

Brahimi B. (2007). Proposition d'une approche intégrée basée sur les réseaux de Petri de haut niveau pour simuler et évaluer les systèmes contrôlés en réseau. Thèse de doctorat, Université Henri Poincaré Nancy 1, Centre de Recherche en Automatique de Nancy.

Brahimi B., E. Rondeau et C. Aubrun (2008). Integrated approach based on high Level Petri Nets for evaluating networked control systems. 16th Mediterranean Conference on Control and Automation (MED'08). Ajaccio, France.

Branicky M.S., S. M. Phillips et W. Zhang (2000). Stability of Networked Control Systems: Explicit Analysis of Delay. In *Proceedings of the American Control Conference*, Chicago, Illinois.

Branicky M. S., V. Liberatore et S. M. Phillips (2003). Networked Control System Co-Simulation for Co-Design. In *Proceeding of the American Control Conference*, Denver, USA, vol. 4, pages 3341-3346.

Carpignano A., R. Caponetto, L. Lo Bello et O. Mirabella (2002). Fuzzy traffic smoothing: An approach for real-time communication over Ethernet networks. In *Proc. of the 4th IEEE Workshop on Factory Communication Systems, WFCS 2002*, Västerås, Sweden, pages 241-248.

Cervin, D. Henriksson, B. Lincoln, J. Eker et K.-E. Arzen (2003). How does control timing affect performance?. *IEEE Control Systems magazine*, vol. 23, pages 16-30.

- Chang C.S. (2000). *Performance Guarantees in Communication Networks*. Springer-Verlag, London, UK.
- Chen C.-C., A. Molin et S. Hirche (2009). *Guaranteed Performance Cost Control over Quality-of-Service Networks*. European Control Conference 2009 (ECC'09), Budapest, Hungary.
- COINC Projet, INRIA (2006). *COINC – Computational Issues in Network Calculus*. Arc INRIA Project.
- Cordon O., M.J. Del Jesus et F. Herrera (1999). A proposal on reasoning methods in fuzzy rule-based classification systems. *International Journal of Approximate Reasoning*, vol. 20, pages 21-45.
- Cordon O., F. Gomide, F. Herrera, F. Hoffmann et L. Magdalena (2004). Ten years of genetic fuzzy systems: current framework and new trends. *Fuzzy Sets and Systems*, vol. 141, pages 5-31.
- Cruz R. L. (1991a). A calculus for network delay, part I : Network elements in isolation. *IEEE Transactions on information theory*, 37(1), pages 114–131.
- Cruz R. L. (1991b). A calculus for network delay, part II : Network analysis. *IEEE Transactions on information theory*, 37(1), pages 132–141.
- De Carvalho F.A.T. (2007). Fuzzy c-means clustering methods for symbolic interval data, *Pattern Recognition Letters*, vol. 28, pages 423-437.
- Decotignie J.D. (2001). A perspective on Ethernet-TCP/IP as a fielbus. In : 4th IFAC International Conference on Fieldbus Systems and their Applications (FET'2001), Nancy, France, pages 138–143.
- Demers A., S. Keshav et S. Shenker (1989). Analysis and simulation of a fair queueing algorithm. In *ACM SIGCOMM Computer Communication Review*, vol. 19, no. 4, pages 1–12.
- Diouri I., J.-P. Georges et E. Rondeau (2007a). Accommodation of delays for NCS using classification of service. In *IEEE International Conference on Networking, Sensing and Control (ICNSC'07)*, London, United Kingdom, pages 410–415.
- Diouri I., J.-P. Georges et E. Rondeau (2007b). Adaptation of scheduling policy parameters for controlling networked systems. *3rd International Workshop on Networked Control System: Tolerant to faults*, Nancy, France.
- Diouri I., C. Berbra, J.-P. Georges, S. Gentil et E. Rondeau (2008). Evaluation of a switched Ethernet network for the control of a quadrotor. In *16th Mediterranean Conference on Control and Automation (MED'08)*, Ajaccio, Corsica, France.
- Diouri I., V. Bombardier, J.-P. Georges et E. Rondeau (2010). Classification des délais pour les Systèmes Contrôlés en Réseau à ordonnancement WRR. *6ème Conférence Internationale Francophone d'Automatique (CIFA 2010)*, Nancy, France.
- Divoux T., E. Rondeau et J.-P. Georges (2008). Ethernet : des îles au bureau, puis dans l'atelier. *La Revue de l'Electricité et de l'Electronique*, n° 1, janvier 2008, pages 95-101.
- Dubois D. et H. Prade (1996). What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, vol. 84, 169-185.
- Eidson J.C. (2006). *Measurement, Control, and Communication Using IEEE 1588*. Springer.
- Felicioni F., N. Jia, F. Simonot-Lion et Y.Q. Song (2008). Optimal on-line (m,k)-firm constraint assignment for real-time control tasks based on plant state information. *13th IEEE ETFA*, Hamburg, Germany.
- Felicioni F., F. Simonot, F. Simonot-Lion et Y.Q. Song (2009). Optimal control with packet drops in networked control systems. *IFAC FeT2009*, Ansan, Korea.

Fidler M. et V. Sander (2004). A parameter based admission control for differentiated services networks. *Computer Networks*, 44(4), pages 463-479.

Floyd, S. et K. Fall (1999). Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Transactions on Networking*, 7(4), pages 458-472.

Forest J., M. Rifqi et B. Bouchon-Meunier (2006). Class Segmentation to Improve Fuzzy Prototype Construction: Visualization and Characterization of Non Homogeneous Classes. *Proc. 15th Int. Conf. on Fuzzy Systems, FUZZIEEE'06, Vancouver, Canada*, pages 555-559.

Fraisse P. et A. Leleve (2003). Teleoperation over IP Network : Network Delay Regulation and Adaptive Control. *Journal of Autonomous Robots special Issue Internet Online Robots*, Kluwer Academics Publishers, vol. 15, no. 3, pages 225-235.

García-Cerezo A., A. Ollero et J. Aracil (1994). Dynamic analysis of fuzzy logic control structures. In: Kandel and Langholz, Editors, *Fuzzy Control Systems*, CRC Press, Boca Raton, pages 141-159.

Georges J.-P. (2005). Systèmes contrôlés en réseau : Evaluation de performances d'architectures Ethernet commutés. Thèse de doctorat, Université Henri Poincaré Nancy 1, Centre de Recherche en Automatique de Nancy.

Georges J.-P., T. Divoux et E. Rondeau (2005). Confronting the performances of a switched Ethernet network with industrial constraints by using the Network Calculus. *International Journal of Communication Systems*, 18(9), pages 877-903.

Goktas F. (2000). Distributed control of systems over communication networks. Ph.D. dissertation, University of Pennsylvania.

Habib G., P. Marangé, J.-F. Pétin et T. Divoux (2009). Evaluation de l'influence d'un réseau de communication sans fil sur la commande d'un SED. 7ème colloque francophone sur la Modélisation des Systèmes Réactifs (MSR'09), Nantes, France.

Halevi Y. et A. Ray (1988). Integrated communication and control systems: Part I-Analysis. In *J. Dynamic Systems, Measurement, and control*, vol. 110, pages 367-373.

Hamdaoui M. et P. Ramanathan (1995). A dynamic priority assignment technique for streams with (m, k)-firm deadlines. *IEEE Transactions on Computers*, 44(4), pages 1443-1451.

Hao P.Y., J.H. Chiang et Y.K. Tu (2007). Hierarchically SVM classification based on support vector clustering method and its application to document categorization. *Expert Systems with Applications* 33, pages 627-635.

Härtig H., L. Reuther, J. Wolter, M. Borriß et T. Paul (1999). Cooperating resource managers. In *Fifth IEEE Real-Time Technology and Applications Symposium (RTAS)*, Vancouver, Canada.

JONES H. (2007). Raisonement à base de règles implicatives floues – Inférence et Sémantique -. Doctorat de l'université de Toulouse.

Hoang H., M. Jonsson, U. Hagström et A. Kallerdahl (2002). Switched Real-Time Ethernet and Earliest Deadline First Scheduling - Protocols and Traffic Handling. 16th International Parallel and Distributed Processing Symposium (IPDPS 2002), Fort Lauderdale, Florida, USA.

Hong S. H. (1995). Scheduling algorithm of data sampling times in the integrated communication and control systems. *IEEE Transactions on Control Systems Technology*, 3(2), 225-230.

IEC (2006a). Digital data communications for measurement and control–fieldbus for use in industrial control systems : Part 3 : Data link service definition. IEC Standard 61158.

IEC (2006b). Digital data communications for measurement and control–fieldbus for use in industrial control systems : Part 4 : Datalink protocol specification IEC Standard 61158.

IEEE (1998). IEEE Standards for information technology – Telecommunications and information exchange between systems - Local and metropolitan area networks - Common specifications - Part 3: Media Access Control (MAC) bridges, ANSI/IEEE Std 802.1D, Edition 1998.

IEEE Computer Society (2002). IEEE standard for information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications. IEEE standard 802.3, Edition 2002.

Ishibuchi H., K. Nozaki et H. Tanaka (1992). Distributed representation of fuzzy rules and its application to pattern classification, *Fuzzy Sets and Systems*, vol. 52, pages 21-32.

Ishibuchi H., K. Nozaki et H. Tanaka (1997). A Simple but powerful heuristic method for generating fuzzy rules from numeric data. *Fuzzy sets and systems*, vol. 86, pages 251-270.

Lung, B. (2002). Contribution à l'Automatisation des Systèmes Intelligents de Production : Interopérabilité des Processus de Contrôle, Maintenance et Gestion Technique. Habilitation à Diriger des Recherches, Université Henri Poincaré Nancy 1, CRAN.

ISO (1993). Road vehicle - Interchange of digital information - Controller Area Network (CAN) for high-speed communication. ISO 11898.

Jacobson V. (1988). Congestion avoidance and control. *ACM Computer Communications Review*, 18(4), pages 314–329.

Jasperneite J., P. Neumann, M. Theis et K. Watson (2002). Deterministic real-time communication with switched Ethernet. In 4th IEEE WFCS, pages 11–18.

Juanole G. et G. Mouney (2006). Real time distributed systems: QoS and impact on the performances of process control applications. In *Mathematical Theory of Networks and Systems*, Kyoto, Japan.

Juanole G. et G. Mouney (2007). Networked control systems: definition and analysis of a hybrid priority scheme for the message scheduling. *IEEE RTCSA 2007*, Daegu, Korea.

Kelly F. P., A. Maulloo et D. Tan (1998). Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49, pages 237–252.

Kettunen M. (2006). Fault tolerant MPC with an embedded FDI system. Master Thesis. Supervisor : Professor Sirkka-Liisa Jämsä-Jounela.

Kickert W. et E.H. Mamdani (1978). Analysis of a Fuzzy Logic Controller. *Fuzzy Sets and Systems*, vol. 1, pages 29-44.

Kim D. K., P.-G. Park et J. W. Ko, (2004). Output-feedback  $H_\infty$  control of systems over communication networks using a deterministic switching system approach. *Automatica*, 40, pages 1205-1212.

Kim S. H. et P.-G. Park (2009). Networked-based robust  $H_\infty$  control design using multiple levels of network traffic. *Automatica*, 45, pages 764-770.

- Kopetz H., A. Damm, C. Kosa, M. Mulazzani, W. Schwalb, C. Senft et R. Zailinger (1989). Distributed Fault-Tolerant Real-Time Systems: The MARS approach. In *proc. of IEEE Micro*, pages 25-40.
- Koubaâ A. (2004). Gestion de la Qualité de Service temporelle selon la contrainte (m,k)-firm dans les réseaux à commutation de paquets. Thèse de doctorat de l'INPL, LORIA-TRIO.
- Kweon S., K. G. Shin et Q. Zheng (1999). Statistical real-time communication over ethernet for manufacturing automation systems. In *Proc. 5th IEEE Real-Time Technology and Application Symp.*, Vancouver, BC, Canada.
- Kweon S., K. G. Shin et G. Workman (2000). Achieving real-time communication over ethernet with adaptive traffic smoothing. In *Proc. Sixth IEEE Real-Time Technology and Applications Symp., RTAS 2000*, Washington, DC, pages 90-100.
- Lann G. et Rivière N. (1993). Real-time communications over broadcast networks: the CSMA/DCR and the DOD CSMA/CD protocols, Technical report 1863, INRIA.
- Le Boudec J.-Y. et P. Thiran (2004). Network calculus, a theory of deterministic queuing systems for the Internet. Springer Verlag LNCS 2050. .
- Lenzini L., E. Mingozzi, G. Stea (2008). A methodology for computing end-to-end delay bounds in FIFO-multiplexing tandems. *Performance Evaluation*, vol. 65, pages 922-943.
- Lian F.-L. (2001). Analysis, design, modeling and control of networked control systems. Ph.D. dissertation, Department of Mechanical Engineering, University of Michigan.
- Lian F.-L., J. Moyne et D. Tilbury (2001). Performance evaluation of control networks. *IEEE Control Systems magazine*, vol. 21, no. 1, pages 66-83.
- Lian F.-L., Y.-C. Tu, C.-W. Li (2004). Ethernet Switch Controller Design for Real Time Control Applications. *Proceedings of 2004 IEEE Conference on Control Applications*, Taipei, Taiwan.
- Lin C.-L., C.-H. Chen et H.-C. Huang (2008). Stabilizing control of networks with uncertain time varying communication delays. *Control Engineering Practice*, vol. 16, pages 56-66.
- Liu C.L. et J.W. Layland (1973). Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1), pages 46-61.
- Lo Bello L., G.A. Kaczynski et O. Mirabella (2005). Improving the Real-Time Behaviour of Ethernet Networks Using Traffic Smoothing. In *IEEE Transactions on Industrial Informatics*, vol.1, n° 3.
- Loeser J. et H. Haertig (2004). Low-latency hard real-time communication over switched ethernet. In *16th Euromicro Conference on Real-Time Systems*, Catania, Sicily.
- Mascolo S. (1999). Congestion control in high-speed communication networks using the Smith principle. *Automatica*, 35(12).
- Mascolo, S. (2006). Modeling the Internet congestion control using a Smith controller with input shaping. *Control Engineering Practice*, 14(4), pages 425-435.
- Mechraoui A., Z. H. Khan et J.-M. Thiriet (2009). Effect of Packet Loss on the Quality of Control of a Networked Mobile Robot. *30th IFAC Workshop on Real-Time Programming and 4th International Workshop on Real-Time Software (RTS'09)*, Mragowo, Poland, pages 97-101.
- Meng C., T. Wang, W. Chou, S. Luan, Y. Zhang et Z. Tian (2004). Remote surgery case: robot-assisted teleneurosurgery. In *IEEE Int. Conf. on Robot. and Auto. (ICRA'04)*, volume 1, pages 819-823.

Michaut, F. et F. Lepage (2003). Un cadre pour la prise en compte du retard dans la téléopération d'un robot mobile. Dans : JDA'03 –Journées Doctorales d'Automatique", Valenciennes.

Michie, D., D.J. Spiegelhalter et C.C. Taylor (1994). Machine Learning Neural and Statistical Classification. Ellis Horwood.

Michiels W., V. Van Assche et S.I. Niculescu (2005). Stabilization of time-delay systems with a controlled time-varying delay and applications, IEEE Trans. Autom. Control, 50, pages 493-504.

Nauck, D., Kruse, R., 1998. Nefclass-x—a soft computing tool to build readable fuzzy classifiers. BT Technology Journal 16 (3), 180–190.

Nauck, D. et R. Kruse (1999). Obtaining interpretable fuzzy classification rules from medical data. Artificial Intelligence in Medicine 16, 149–169.

Nilsson J. (1998). Real-Time Control Systems with Delays. Ph.D. dissertation, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Nutzer Organisation e.v. (1992). Profibus standard DIN 19245 Part I and II. Translation from German.

Oh S. et W. Pedrycz (2000). Identification of fuzzy systems by means of an auto-tuning algorithm and its application to nonlinear systems. Fuzzy Sets and Systems. Volume 115, Issue 2, pages 205-230.

Olfati-Saber R., J. Alexander Fax et R. M. Murray (2007). Consensus and cooperation in networked multiagent systems. Proc. IEEE, 95(1), pages 215–233.

OPNET (2005). OPNET, University program [Online], Available at: [http://www.opnet.com/university\\_program/index.html](http://www.opnet.com/university_program/index.html).

Parekh A. (1992). A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks. PhD thesis, Massachusetts Institute of Technology.

Parekh K. et R. G. Gallager (1993). A generalized processor sharing approach to flow control in integrated services networks: The single node case. IEEE/ACM Trans. Networking, vol. 1, Issue 3, pages 344–357.

Patton R. J., C. Kambhampati, A. Casavola, P. Zhang, S. Ding et D. Sauter (2007). A Generic Strategy for Fault-Tolerance in Control Systems Distributed over a Network. European Journal of Control, 13: 280-296.

Pedreiras P., R. Leite, L. Almeida (2003). Characterizing the real-time behavior of prioritized switched Ethernet. In 2nd Intl Workshop on Real-Time LANS in the Internet AGE, Porto, Portugal.

Pleinevaux P. et J.-D. Decotignie (1988). Time Critical Communication Networks: Field Busses, IEEE Network Magazine, vol. 2, pages 55-63.

Ramanathan P. (1999). Overload management in Real-Time control applications using (m,k)-firm guarantee. IEEE Transactions on Parallel and Distributed Systems, 10(6), pages 549-559.

Richalet J. (1993). Industrial applications of model based predictive control. Automatica 29, pages 1251-1274.

Richard J.P. (2003). Time-delay systems: an overview of some recent advances and open problems. Automatica, 39, pages 1667-1694.

Richard J.-P. et T. Divoux (2007). Systèmes commandés en réseau, Edition Lavoisier, Hermès sciences publications.

- Robert J., J.-P. Georges, E. Rondeau et T. Divoux (2010). Analyse de performances de protocoles temps-réel basés sur Ethernet. 6ème Conférence Internationale Francophone d'Automatique (CIFA 2010), Nancy, France.
- Roubos, J.A., Setnes, M., Abonyi, J., 2003. Learning fuzzy classification rules from labeled data. *Information Sciences* 150, 77–93.
- Schioler H., H. Schwefel et M. Hansen (2007). « CyNC – a Matlab/Simulink toolbox for network calculus ». In Proc. of VALUETOOLS.
- Schmitt E., V. Bombardier, P. Charpentier (2007). Self-fuzzification method according to typicality correlation for classification on tiny data sets. In : IEEE Conference on Fuzzy Systems, London, UK, pages 1072–1077.
- Schmitt J. et U. Roedig (2005). Sensor network calculus – a framework for worst case analysis. In Proc. IEEE/ACM DCOSS, pages 141-154.
- Schmitt J. et F. Zdarsky (2006). The DISCO Network Calculator - a toolbox for worst case analysis. In Proc. of VALUETOOLS, ACM.
- Seiler P. et R. Sengupta (2005). An  $H_\infty$  approach to networked control. *IEEE Transactions on Automatic Control*, 50(3), pages 356-364.
- Sename O., D. Simon et D. Robert (2003). Feedback scheduling for real-time control of systems with communication delays. *IEEE Conference ETFA'03*, Lisbonne, Portugal.
- Seto D., J. P. Lehoczky, L. Sha et K. G. Shin (1996). On task schedulability in real-time control system. *Proceedings of the 17th IEEE Real-Time Systems Symposium*, pages 13-21, Washington, DC.
- Seuret A., M. Dambrine et J.-P. Richard (2004). Robust exponential stabilization for systems with time-varying delays. In *5th IFAC Workshop on Time Delay Systems*, Leuven, Belgium.
- Simon D., D. Robert et O. Sename (2005). Robust control/scheduling co-design : application to robot control. In *RTAS'05*, San Francisco, CA, USA.
- Slacky S., V. Palacka, M. Horna, R. Minarik, E. Rondeau, J.-P. Georges, I. Diouri (2006). Livrable 7-3 : Network performance evaluation. *Projet STREP NeCST*. Song Y.-Q (2009). Networked Control Systems : From Independent Designs of the Network QoS and the Control to the Co-design. *8th IFAC International Conference on Fielbuses & neTworks in industrial & Embedded Systems (FeT 2009)*, Ansan, Korea.
- Stubbs A. et G.E. Dullerud (2001). Networked control of distributed systems : A testbed. In *ASME International Mechanical Engineering Congress and Exposition*.
- Sugeno M. et T. Taniguchi (2004). On improvement of stability conditions for continuous Mamdani-like fuzzy systems. *IEEE Trans. Systems, Man, Cybern. Part B*, vol. 34, pages 120-131.
- Tang A., K. Jacobsson, L. L. H. Andrew et S. H. Low (2007). An accurate link model and its application to stability analysis of FAST TCP. In *IEEE INFOCOM*, Anchorage, Alaska, USA.
- Tanwani A., J. Galdun, J.-M. Thiriet, S. Lesecq et S. Gentil (2007). Experimental networked embedded minidrone – Part I. Consideration of faults. *ECC 2007*, Kos, Greece.
- Tipsuwan Y. et M.-Y. Chow (2003). Control methodologies in networked control systems. *Control Engineering Practice* 11, pages 1099-1111.

Tipsuwan Y. et M.-Y. Chow (2004). Gain Scheduler Middleware : A Methodology to Enable Existing Controllers for Networked Control and Teleoperation\_\_Part II: Teleoperation. In IEEE Transactions on Industrial Electronics, vol. 51, no. 6.

Thiele L., S. Chakraborty, M. Gries, A. Maxiaguine et J. Greutert (2001). Embedded software in network processors models and algorithms. In: Proceedings of Embedded Software Workshop EMSOFT'2001.

Tobagi F. (1982). Carrier Sense Multiple Access with Message-Based Priority Functions. In proc. of IEEE Transactions on Communications, vol. Com-30, n° 1.

Tong M. R. (1977). Some Problems with the design and Implementation of Fuzzy controllers. British Steel Corp., Res. Associate. Control and management System Group. Univ. of Cambridge, England.

Tou, J.Y., Tay, Y.H., Lau, P.Y., 2009. Recent trends in texture classification: a review. In: Symposium on Progress in Information & Communication Technology, December 7–8, Kuala Lumpur, Malaysia, pp. 64–68.

Turner J. S. (1986). New directions in Communications. IEEE Comm. Magazine, 24(10), pages 8-15.

Vatanski N., J.-P. Georges, C. Aubrun, E. Rondeau et S.-L. Jämsä Jounela (2006). Control compensation based on upper bound delay in networked control systems. In 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS), Japan.

Vatanski N., JP. Georges, C. Aubrun, E. Rondeau et SL. Jämsä Jounela (2009). Networked control with delay measurement and estimation. Control Engineering Practice, vol.17, pages 231-244.

Walsh G.C., H. Ye et L. Bushnell (1999). Stability analysis of networked control systems. In Proceedings of the American Control Conference, San Diego, pages 2876-2880.

Willaeys D., P. Mangin et N. Malvache (1977). Use of Fuzzy Sets for System Modelizing and Control : Application to the Speed Control of a Strongly Perturbed Motor. Proc. of the 5th IFAC/IFIP Int. Conf. on Digital Computer Applications to Process Control, The Hague.

Xiong J. et J. Lam (2007). Stabilization of linear systems over networks with bounded packet loss. Automatica, 43, pages 80-87.

Yang T.C. (1990). Computational delay in digital and adaptive controllers. IEE Control 90 Conference, Brighton, UK.

Yu M., L. Wang, T. Chu et G. Xie (2004). Stabilization of networked control systems with data packet dropout and network delays via switching system approach. In 43rd IEEE conference on decision and control, Atlantis, Paradise Island, Bahamas, pages 3539–3544.

Yue D., Q.-L. Han et J. Lam (2005). Networked-based Robust  $H_\infty$  control of systems with uncertainty. Automatica, 41, pages 999-1007.

Zadeh L. A. (1965). Fuzzy sets. Information and Control, vol. 8, pages 338-353.

Zadeh L. A. (1971). Similarity Relations and fuzzy Orderings, Information Sciences 3, pages 177-200.

Zadeh L.A. (1973). Outline of New Approach to the Analysis of Complex Systems and Decision Process, IEEE Trans. Systems, Man and Cybernetics 3, pages 28-44.

Zadeh L.A. (1975). The concept of linguistic variable and its application to approximate reasoning, Information sciences, vol. 8, pages 199-249 ; pages 301-357.

Zampieri S. (2008). Trends in networked control systems. In 17th IFAC World Congress, Seoul, South Korea, pages 2886-2894.

Zhang W., S. Branicky et S. Phillips (2001). Stability of networked control systems. IEEE Control Systems magazine, vol. 21, pages 84–89.

Zhang P. et S.X. Ding (2007). An integrated trade-off design of observer based fault detection systems Automatica.

Zheng F. et J. Nelson (2007). An h-infinity approach to congestion control design for AQM routers supporting TCP flows in wireless access networks. Computer Networks, vol. 51, pages 1684–1704.







## Résumé

L'étude des Systèmes Contrôlés en Réseaux (SCR) repose sur l'identification des exigences de fonctionnement de l'application appelées Qualité de Contrôle (QdC) et sur l'évaluation de la Qualité de Service (QoS) offerte par le réseau. Les travaux sur les SCR se répartissent selon deux approches : la commande en réseau et la commande de réseau.

Cette thèse se positionne sur la deuxième approche avec une recherche axée sur la modélisation des mécanismes d'ordonnancement implémentés dans les équipements réseau et notamment dans les commutateurs Ethernet qui sont de plus en plus utilisés dans les applications industrielles. Ce travail de recherche étudie plus particulièrement comme paramètre de QoS, les délais qui engendrent des perturbations sur le système commandé.

Cette thèse propose deux modèles de classification de service reposant sur des ordonnanceurs WRR (Weighted Round Robin). La première modélisation suit une approche constructive en utilisant la théorie du calcul réseau. La seconde s'appuie sur une phase d'identification à partir de simulations numériques et de la logique floue. Dans les deux cas, le but est d'offrir une bande passante suffisante pour le trafic contraint temporellement tout en maximisant la bande passante dédiée aux autres trafics pour éviter des effets famine.

L'approche calcul réseau permet de configurer le réseau hors-ligne pour répondre à des contraintes temporelles strictes du SCR. La solution basée sur la logique floue autorise une commande dynamique de l'ordonnanceur pour ajuster en ligne le réseau en fonction des variations du trafic. Elle ne peut s'appliquer qu'à des SCR ayant des contraintes de temps souples.

## Mots clés

Applications temps-réel, systèmes contrôlés en réseaux, calcul réseau, logique floue, politique d'ordonnancement, qualité de service.

## Abstract

The study of the Networked Control Systems (NCS) is based both on the identification of the application functioning requirements called Quality of Control (QoC) and on the evaluation of the Quality of Service (QoS) offered by the network. The studies on the NCS are classified according to two approaches: the control over network and the control of network.

This thesis addresses the second approach and models the scheduling mechanisms implemented in the Ethernet switches that are more and more used in the industrial applications. The specific QoS parameter studied in this thesis is the delay disturbing the controlled system.

This thesis proposes two models of classification of service based on WRR (Weighted Round Robin) schedulers. The first modeling follows a constructive approach by using the network calculus theory. The second is based on an identification step from numerical simulations and from the fuzzy logic. In the two cases, the purpose is both to offer enough bandwidth for the time constrained traffic and to maximize the bandwidth dedicated to the others traffics to avoid famine effects.

The network calculus approach is used to configure off-line the network in respecting the NCS strict time constraints. The solution based on the fuzzy logic enables a dynamic control of the scheduler in order to tune on-line the network according to the traffic variations. This latter can be applied only to NCS with soft time constraints.

## Keywords

Real-time applications, networked control systems, network calculus, fuzzy logic, scheduling policy, quality of service.