Université d'Ibn Khaldoun – Tiaret
Faculté des Mathématiques et de l'Informatique
Département Informatique

Pour l'obtention du diplôme de Master II

Spécialité : Réseaux et Télécommunication.

Thème

Développement de politiques d'ordonnancement de paquets IP pour un routeur d'internet

Présenté par:
BENMESTOURA Khaldia
BENSAHRAOUI Khaldia

Dirigé par :
Mr. NASSANE samir

Année universitaire 2014-2015

Sommaire

Dédicace

Je dédie ce modeste travail

A mes très chers parents

Qui m'ont toujours encouragé et orienté durant

Mes années d'études.

A mes chers frères et sœurs

A mes tantes et mes oncles.

A mes cousins et mes cousines.

A toute la famille.

A tous mes ami(e)s

A toute la promo sortante 2014– 2015

BENSAHRAOUI KHALDIA

Dédicace

Je dédie ce modeste travail

A mes très chers parents

Qui m'ont toujours encouragé et orienté durant

Mes années d'études.

A mes chers frères et sœurs

A mes tantes et mes oncles.

A mes cousins et mes cousines.

A toute la famille.

A tous mes ami(e)s

A toute la promo sortante 2014– 2015

BENSAHRAOUI KHALDIA

Dédicace

Je dédie ce modeste travail

A mes très chers parents

Qui m'ont toujours encouragé et orienté durant

Mes années d'études.

A mes chers frères et sœurs

A mes tantes et mes oncles.

A mes cousins et mes cousines.

A toute la famille.

A tous mes ami(e)s

A toute la promo sortante 2014– 2015

BENMASTOURA KHALDIA

Remerciement

Nous tenons tout d'abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce Modeste travail.

Ce mémoire n'aurait pas été possible sans l'intervention, consciente, d'un grand nombre de personnes.

Nous souhaitons ici les remercier.

Mon encadreur, Mr. Nassane Samir pour son entière disponibilité, ses conseils et ses encouragements et pour nous avoir proposé ce sujet.

Un grand Merci aux enseignants ainsi que l'administration de la faculté informatique qui a veillé sur notre formation et notre suivi durant tout le cursus d'étude.

Nous remercions également les membres du jury d'avoir accepté d'évaluer notre modeste travail.
En fin nous adressons nos remerciements à tous ceux qui ont contribué par leurs conseils ou leurs encouragements à l'aboutissement de ce travail.

Résumé

Internet est aujourd'hui un réseau multi-applications, qui nécessite une meilleure gestion des connexions utilisées, pour garantir le bon fonctionnement de certaines applications multimédia ayant des exigences spécifiques en terme de QoS (*Quality of Service* : Qualité de service).

Pour pouvoir garantir la QoS des flux transportés, on doit utiliser des mécanismes permettant de traiter de manière différente tous les catégories de trafic dans les organes du réseau. Parmi ces mécanismes de QoS, on s'intéresse dans ce travail aux politiques d'ordonnancement de paquets IP.

Ainsi, nous présentons en détail les politiques les plus importantes disponibles sur les matériels réseaux actuels. Nous commencerons par les mécanismes d'ordonnancement les plus "anciens" et les plus "simples" tels que FIFO (*First In/First Out*), SPQ (*Strict Priority Queuing*), et CQ (*Custom Queuing*) tout en décrivant leurs avantages/inconvénients. Ensuite nous étudierons les approches les plus récentes WFQ (*Weighted Fair Queuing*) et ses dérivées CBWFQ (*Class Based WFQ*) et LLQ (*Low Latency Queuing*).

Mots-clés : *QoS, l'ordonnancement,* SPQ , FQ,WFQ, LLQ.

Introduction Générale

Depuis quelques années, il y a un réel changement dans les usages des réseaux en termes d'applications véhiculées ainsi que dans leur nombre. On voit apparaître de plus en plus d'applications contraintes en termes de délai, comme par exemple la Téléphonie sur IP, ainsi que d'applications gourmandes en ressources comme par exemple le Streaming Vidéo, la croissance en volume de ces applications commence à poser des problèmes de congestion dans les réseaux filaires et sans fil.

Les distances qui séparent les stations émettrices des stations réceptrices sont importantes et induisent des délais de transmission, la traversée des différents éléments du réseau engendre, des délais, de même, les capacités des liens sont limitées. Ainsi, lors de la circulation de plusieurs flots, les caractéristiques du réseau engendrent des aspects négatifs sur la transmission de ces informations.

Ces derniers peuvent se traduire par des délais d'acheminement importants, par une incapacité d'obtenir une bande-passante suffisante ou encore par la perte de paquets. Selon la nature de l'application, ces effets sont nuisibles à la qualité de service requise. En effet, dans des conditions de surcharge du réseau, les applications temps-réel qui ne peuvent pas disposé de ressources suffisantes ne pourront pas offrir la qualité de service attendue.

Pour remédier à ces problèmes de partage de ressources, des mécanismes intelligents en termes d'ordonnancement et de gestion des files d'attente, sont implémentés dans les routeurs, ce qui permet de bien administrer la circulation des flux sur le réseau.

Dans ce travail, nous étudions les algorithmes d'ordonnancement, les plus connues, tout en décrivant leurs avantages et inconvénients.

Chapitre I : Les algorithmes d'ordonnancement

I.1 Introduction

Un réseau d'aujourd'hui doit supporter des applications ayant des contraintes diverses y Compris les contraintes du temps réel. Pour fournir une certaine garantie de service (QoS : Quality of Service), les algorithmes d'ordonnancement des trafics tel que (priorité fixe, WFQ, …) ont été développé et déployé dans les réseaux multimédias pour fournir une bande passante constante à chaque flux servi et assurent une borne maximale.

Dans ce chapitre, nous présentons les différents politiques d'ordonnancement de paquets IP pour un routeur d'internet.

I.2 Qualité de service

I.2.1 Définition et objectif

La qualité de service généralement noté par l'abréviation QoS pour (Quality of Service), se définit comme un concept visant à optimiser les ressources d'un réseau informatique afin de garantir des performances minimales requises. Ainsi ce service permet de véhiculer dans de bonnes conditions des informations à travers d'un routeur.[7]

I.2.2 Paramètres de qualité de service

Les principaux aspects connus de la qualité de service sont le délai, la gigue, le débit, la bande passante et la disponibilité (souvent exprimée en termes de taux d'erreurs).

I.2.2.1 Le délai

Le délai est la métrique de performance la plus importante pour les applications temps-réel. Le délai est défini comme étant l'intervalle de temps entre le départ de données de la source jusqu'à leur arrivée à la destination et est généralement appelé le délai de bout-en-bout.

I.2.2.2 La gigue

La gigue est une métrique importante pour les applications temps-réel. Elle se réfère généralement à la variation du délai.

I.2.2.3 Bande passante (Débit)

Le terme est employé dans les domaines de l'informatique et de la transmission numérique pour signifier en fait le débit binaire d'un canal de communication ; c'est la quantité de bits fournie dans un temps donné.

Dans les réseaux à commutation de paquets, la garantie de bande passante est un paramètre de performance très important pour garantir la QoS aux flux temps-réel.

I.2.3Architecture de qualité de service.

Les réseaux IP ont été conçus pour transporter des flux sans contrainte. Avec le développement des applications multimédia, l'acheminement des trafics avec la garantie d'une certaine qualité de service devient plus que jamais nécessaire .Le déploiement des applications temps réel sur un réseau exige la garantie de temps de réponse des paquets transmis. C'est la raison pour laquelle l'Internet d'aujourd'hui intègre deux types d'architecture de la qualité de service (QoS) :

I.2.3.1 Le modèle à intégration de services (IntServ) :

Consiste à réserver les ressources nécessaires au niveau de tous les nœuds du réseau avant de faire transiter les flux. Cette approche adopte le protocole RSVP (Ressource Reservation Protocol) qui fait appel à une couche de contrôle d'admission supplémentaire pour s'assurer que la bande passante requise est bien disponible à un instant T.

Dans IntServ, les routeurs doivent maintenir des tables pour mémoriser l'état de chaque flux et des allocations de ressources.[SIT02]

I.2.3.2 Le modèle à différenciation de services (DiffServ):

L'identification et le marquage du paquet est donc fait l'entrée de réseau et les nœuds intermédiaires ou de cœur du réseau se contente d'appliquer les politiques de gestion du flux en fonction de cette valeur.

L'approche DiffServ permet un déploiement et une exploitation simplifiée, et offre des performances meilleures pour un réseau à grande échelle.[SIT02]

Dans un réseau internet, les informations sont transmises sous la forme de paquets, échangés de routeur en routeur jusqu'à la destination souhaitée,Tous les traitements vont donc s'opérer sur ces paquets.

Le principe de la qualité de service dans un routeur se décompose en quatre étapes :

- Classification
- Contrôle d'admission
- Gestion des files d'attentes
- Ordonnancement

Nous nous concentrerons évidement sur cette dernière étape qui décrit simplement la manière dont les paquets vont être ordonnés pour être finalement émis sur l'interface de sortie.

I.3 L'ordonnancement

C'est l'un des composants les plus critiques dans l'architecture d'un réseau à QoS. Effet, les paquets des différents flux subissent des délais d'attente plus ou moins importants. Ces délais d'attente dans les tampons sont néfastes pour les applications sensibles aux délais.

L'idée du l'ordonnancement est d'adapter la politique de transmission des paquets en attente dans une file, en fonction des besoins en QoS des flux. Le choix de la politique d'ordonnancement a un impact important sur le délai et la gigue.[4]

Nous définissons quelques objectifs des ordonnanceurs:

- **Isolation des flux:** quel que soit le débit des flux sources, la portion de service allouée au différents flux doit être la plus proche de celle requise. En présence de flux à débit élevé, les flux à faible débit ne doivent pas être pénalisés.
- **L'utilisation**: la bande passante du réseau doit être utilisée efficacement.
- **L'équité:** la bande passante doit être partagée équitablement entre les différents usagers actifs de la même classe.
- **Simplicité:** l'algorithme d'ordonnancement doit être facile à implémenter et simple à gérer.

- **Facteur d'échelle:** l'algorithme devra gérer un grand nombre de connexions.

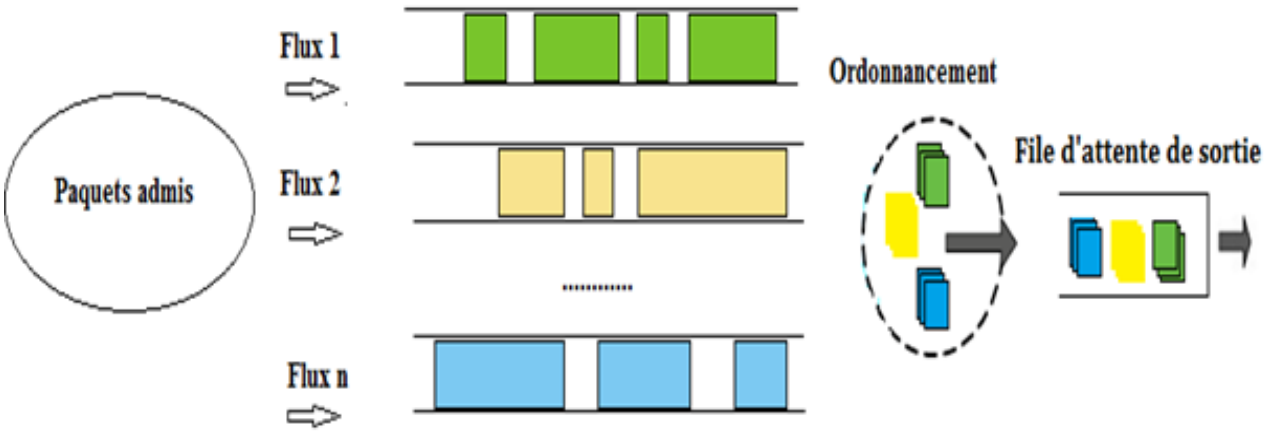


Figure.1 : Ordonnancement.

I.4 Les type d'ordonnancement

Les principales qualités attendues d'un ordonnanceur sont une bonne isolation des flots (traitement séparé des flots, ou découplage des différents flots en classe de trafic), une Implémentation aisée et évolutive, un temps de traitement faible et la qualité de QoS apportée.

Principalement trois types d'ordonnancement ont été envisagés : l'ordonnancement à priorité fixe, l'ordonnancement basé sur le paradigme temps partagé généralisé et celui basé sur la notion de trame temporelle.

I.4.1 Ordonnanceurs à Priorité Fixe

Les ordonnanceurs de type priorité fixe (SPQ) transmettent les paquets dans l'ordre de leurs priorités. Un paquet ne peut être transmis s'il y a un paquet de priorité supérieure en attente.

Ces ordonnanceurs garantissent les délais les plus faibles possibles pour les paquets de priorité haute (temps réel). Toutefois, un des inconvénients majeurs de ce type de politique est que les paquets de priorités inférieures sont défavorisés. Ce type d'algorithme est très simple à implémenter et donc rapide.[4]

I.4.2 Ordonnanceurs basés sur le paradigme temps partagé généralisé

Les ordonnanceurs basés sur le paradigme temps partagé généralisé (FQ, WFQ) fournissent un partage différencié de la bande passante avec un minimum garanti à chaque session. Cette bande passante minimale est basée sur un poids associé à la session.

Une session peut disposer temporairement d'un débit supérieur quand les paquets d'une ou plusieurs autres sessions ne sont pas en attente de transmission.

Ainsi, le débit est partagé équitablement entre les sessions présentes suivant leurs poids.[4]

I.4.3 Ordonnanceurs basés sur la notion de trame temporelle

Les ordonnanceurs basés sur la notion de trame temporelle, comme l'algorithme WRR (Weighted Round Robin), divisent le temps en trames de tailles variables. Ces trames sont allouées aux sessions à la façon round robin. En termes de garanties de délai et de gigue, ces ordonnanceurs sont moins performants que les ordonnanceurs basés sur le paradigme. En revanche, ils sont plus simples à implémenter.[4]

I.6 Les algorithmes d'ordonnancement

Les Algorithmes d'ordonnancement permet de gérer les différentes files d'attente (ou classe) du routeur. Il réorganise en fait l'ordre de sortie des paquets afin de leur donner la qualité de service à laquelle ils ont droit. Cette section présentera brièvement les principaux algorithmes d'ordonnancement.[6]

Les algorithmes d'ordonnancement proposé sont :

1. First In First Out (FIFO).

2. Round Robin (RR).
3. Weighted Round Robin (WRR).
4. Déficit Weighted Round Robin (DWRR).
5. Strict Priorité Queuing (SPQ).
6. Fair Queuing (FQ).
7. WeightedFair Queuing (WFQ).
8. Custom Queuing (CQ).
9. Classe-Basé Weighted Fair Queuing (CBWFQ).
10. Low Latency Queue (LLQ).

Chaque algorithme d'ordonnancement a été conçu pour résoudre un problème spécifique de trafic réseau et a un effet particulier sur les performances du réseau, comme décrit dans la suite:

I.6.1 OrdonnancementFirst In First Out (FIFO)

L'ordonnancement de type FiFo (First In First Out) est l'algorithme le plus facile à implémenter parmi les politiques présentes.

FIFO est également appelée Premier arrivé, premier servi(PAPS) : tous les paquets sont traités de la même manière en les plaçant dans une seule file d'attente, l'algorithme se base sur le principe suivant : les paquets sont envoyés vers la sortie dans le même ordre de leur réception.

FIFO ne peut en aucun cas assurer la fonction de distinction des flots puisque elle n'utilise pas de multiples files.

Généralement, FIFO est supporté sur un port de sortie lorsqu'aucune autre discipline d'ordonnancement n'est configurée.

a) Les avantages du mécanisme FIFO sont :

- Il est simple à implémenter.
- L'ordre des paquets est conservé et le délai peut être prévisible, en fait, le délai maximal correspond à la profondeur de la fille d'attente.

b) Les inconvénients de FIFO sont :

- Si la file d'attente n'a pas d'espace pour stocker les paquets arrivant, la file d'attente supprime les paquets.
- Il n'y a pas de séparation entre les flux et par conséquent, un flux peut perturber le fonctionnement des autres flux. [1][8]

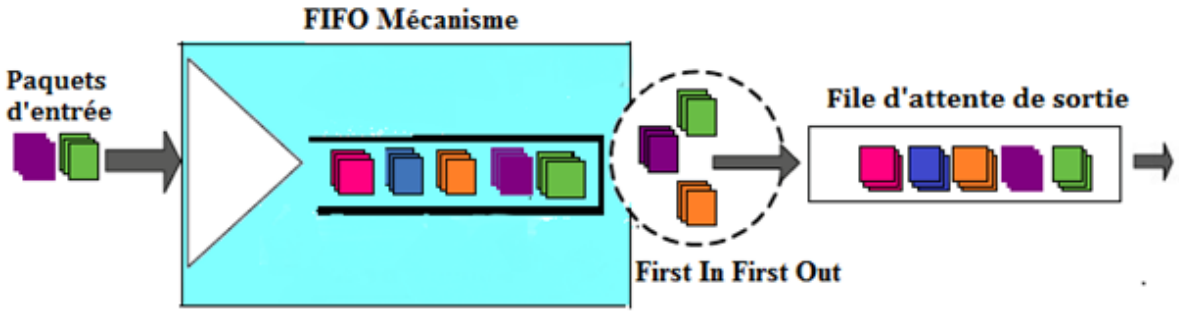


Figure 2 : L'ordonnancement FIFO.

I.6.2Ordonnancement à Tour de Rôle (Round Robin)

Le principe de Round Robin est très adapté à l'isolement des flux grâce au placement des paquets sur plusieurs files d'attente.

L'ordonnanceur parcourt en séquence les files et prend le premier élément. Si une file est vide, l'ordonnanceur passe à la file suivante. L'équité offerte par ce mécanisme dépend de la taille moyenne des paquets dans chaque file: une file contenant des paquets 2 fois plus gros que les autres obtient 2 fois plus de bande passante. Il est donc nécessaire de modifier, l'algorithme pour limiter le nombre d'octets que l'ordonnanceur peut retirer de chaque file afin d'assurer l'équité.

a) Le principal avantage de ce mécanisme est la possibilité de séparer les flux, chaque flux est isolé et son comportement n'altère pas le fonctionnement des autres flux.

b) D'ailleurs, ce mécanisme a des inconvénients :

1. Quand le nombre des flux augmente, la gestion d'un grand nombre de file d'attente sera plus compliquée.
2. RR vise à partager équitablement la bande passante entre les flux, cela ne peut être assuré que si tous les paquets de tous les flux ont la même taille. De plus, RR n'est pas conçu pour supporter des flux avec différents besoins de bande passante. [9]

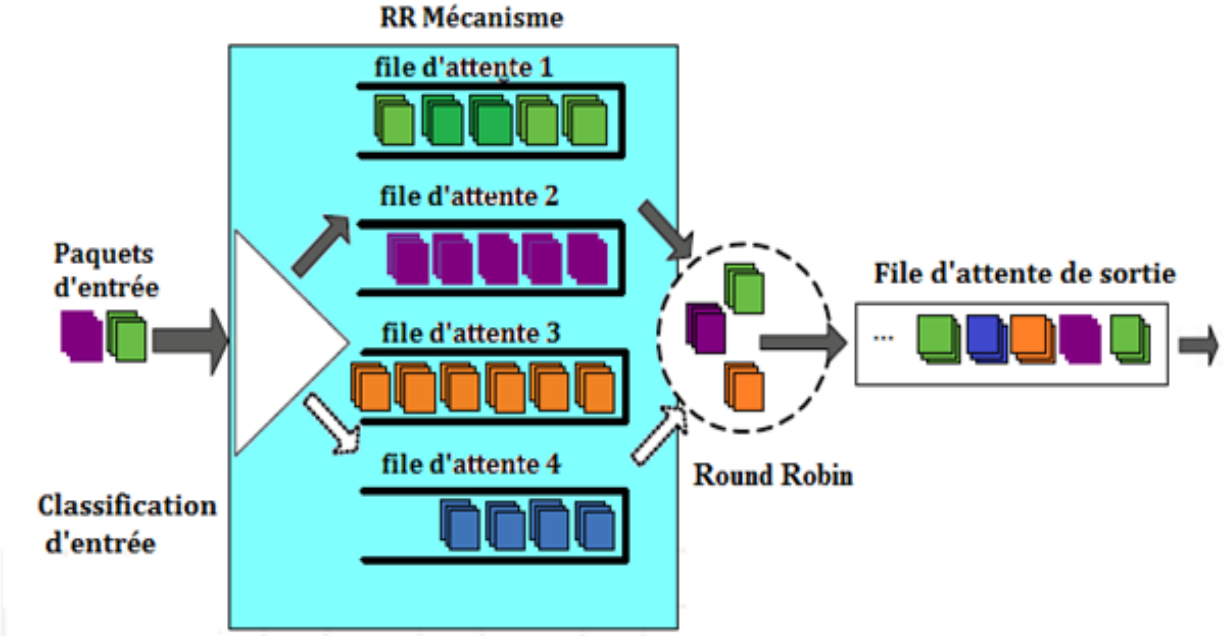


Figure 3 : l'ordonnancement RR.

I.6.3 Weighted Round Robin (WRR)

Dans cette discipline d'ordonnancement, chaque flux de paquets ou une connexion a sa propre file d'attente.

L'ordonnanceur parcourt les différentes files et les sert en fonction du poids associé à chacune d'elles. Cet ordonnanceur reste simple. L'équité est vérifiée si les paquets des différentes files ont la même longueur. Dans le cas contraire, ce sont les files avec les paquets de plus grandes tailles qui seront favorisées. [11]

a) Les avantages de mécanisme WRR sont :

1. WRR assure que toutes les files d'attente ont accès à au moins une certaine quantité configuré de bande passante du réseau.

b) Les inconvénients de WRR sont :

1. plus grande complexité.
2. l'allocation précise de la portion de bande passante est difficile à calculer.

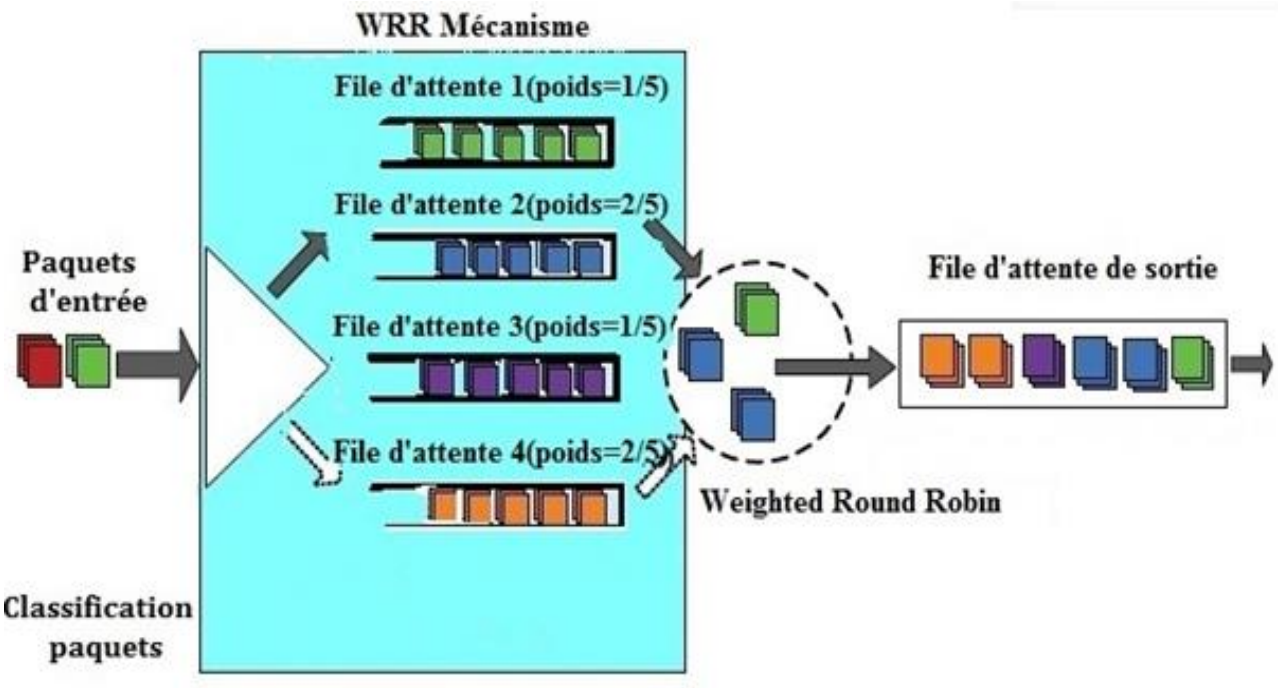


Figure 4 : l'ordonnancement WRR.

I.6.4 Déficit Weighted Round Robin (DWRR)

Déficit Weighted Round Robin (DWRR) a été proposé par M. Shreedhar et G. Varghese en 1995. DWRR est la base d'une classe de disciplines d'ordonnancement qui est conçue pour répondre aux limites des modèles WRR et WFQ.

Dans DWRR, chaque file d'attente est configuré avec un certain nombre de paramètres :

- Un poids qui définit le pourcentage de la bande passante affecté à la file d'attente.
- Un Deficit Counter qui spécifie le nombre total d'octets que la file d'attente est autorisée à transmettre à chaque fois qu'elle est visitée par l'ordonnanceur. Le Deficit Counter permet à une file d'attente qui n'était pas autorisé à transmettre au tour précédent (parce que le paquet à la tête de la file d'attente était plus grand que la valeur

de la Deficit Counter)de sauvegarder des "crédits" et de les utiliser au cours de la prochaine ronde de service.

- Un quantum de service qui est proportionnelle au poids de la file d'attente et qui est exprimé en termes d'octets.

Ainsi, l'ordonnanceur visite chaque file d'attente non vide et détermine le nombre d'octets du paquet à la tête de la file d'attente. Le Deficit Counter variable est incrémentée de la valeur de quantum. Si la taille du paquet à la tête de la file d'attente est supérieure à la Deficit Counter variable, alors l'ordonnanceur se déplace au service de la file d'attente suivante. Si la taille du paquet à la tête de la file d'attente est inférieure ou égale à la variable de Deficit Counter. Le Deficit Counter variable est réduite par le nombre d'octets du paquet et ce dernier est transmis sur le port de sortie. L'ordonnanceur décrémenter la variable de Deficit Counter par la taille du paquet transmis jusqu'à ce que la taille du paquet à la tête de la file d'attente est supérieure au Deficit Counter variable ou la file d'attente est vide. Si la file est vide, la valeur de Deficit Counter est mise à zéro. Dans ce cas, l'ordonnanceur passe au service de la prochaine file. (Voir la figure 5).

a) Les avantages d'ordonnanceur DWRR :

1. Fournit une protection entre les différents flux, de sorte qu'une classe de service d'une file d'attente ne peut pas influer sur les autres classes de service affectées à d'autres files d'attente.
2. Implémente un algorithme relativement simple et peu coûteux.

b) Comme les autres politiques, DWRR à ses limites:

1. DWRR n'offre pas de garanties de délai de bout-en-bout.

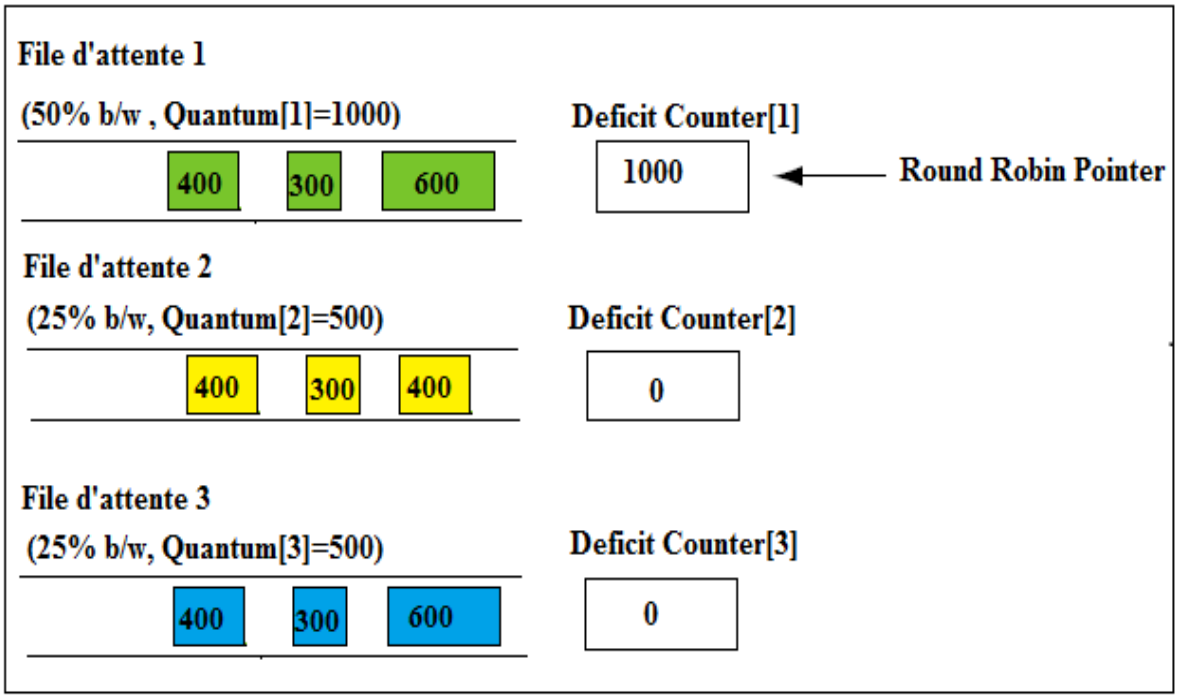


Figure 5 : l'ordonnancement DWRR

I.6.5 Strict Priorité Queuing (SPQ)

Cette technique est sans doute la meilleure alternative d'ordonnancement pour des flux tels que la téléphonie sur IP ou des jeux interactifs.

L'algorithme SPQ est le suivant: supposons qu'ils n'existent que 4 classes de service numérotées de 1à 4 en fonction de leur sensibilité au délai (la classe 1 étant la plus sensible), à chaque passage de l'ordonnanceur, la file 1 est servie. Tant qu'il y a des paquets sur cette file l'ordonnanceur les retire pour les envoyer sur l'interface. Quand la file 1 devient enfin vide, l'ordonnanceur traite les paquets de la file 2 et ainsi de suite. Quand un paquet arrive sur la file 1, il est servi dès que l'interface termine la transmission du paquet en cours quel que soit sa priorité comme présenté dans la (figure 6).

L'algorithme contient un défaut évident, la surcharge d'une classe prioritaire peut entraîner la famine des classes moins prioritaires. L'ordonnanceur étant occupé en

permanence à servir la file 1, et peut être la file 2, ne sert jamais les files 3 et 4, provoquant le débordement récurrent des files de ces deux classes.

a) Les avantages de cet algorithme d'ordonnancement sont :

1. PQ est simple à implémenter et ne nécessite pas beaucoup de temps de traitement.
2. il permet de gérer plusieurs classes et de fournir des traitements différents pour les paquets de différentes classes.

b) Ses inconvénients sont :

1. Si le trafic de plus haute priorité n'est pas contrôlé à l'entrée du réseau, les paquets des autres priorités peuvent subir des délais très élevés.
2. Tous les flux d'une même classe sont agrégés dans une même file d'attente par conséquent, un seul flux peut perturber le fonctionnement des autres flux de la même classe.[8][9].

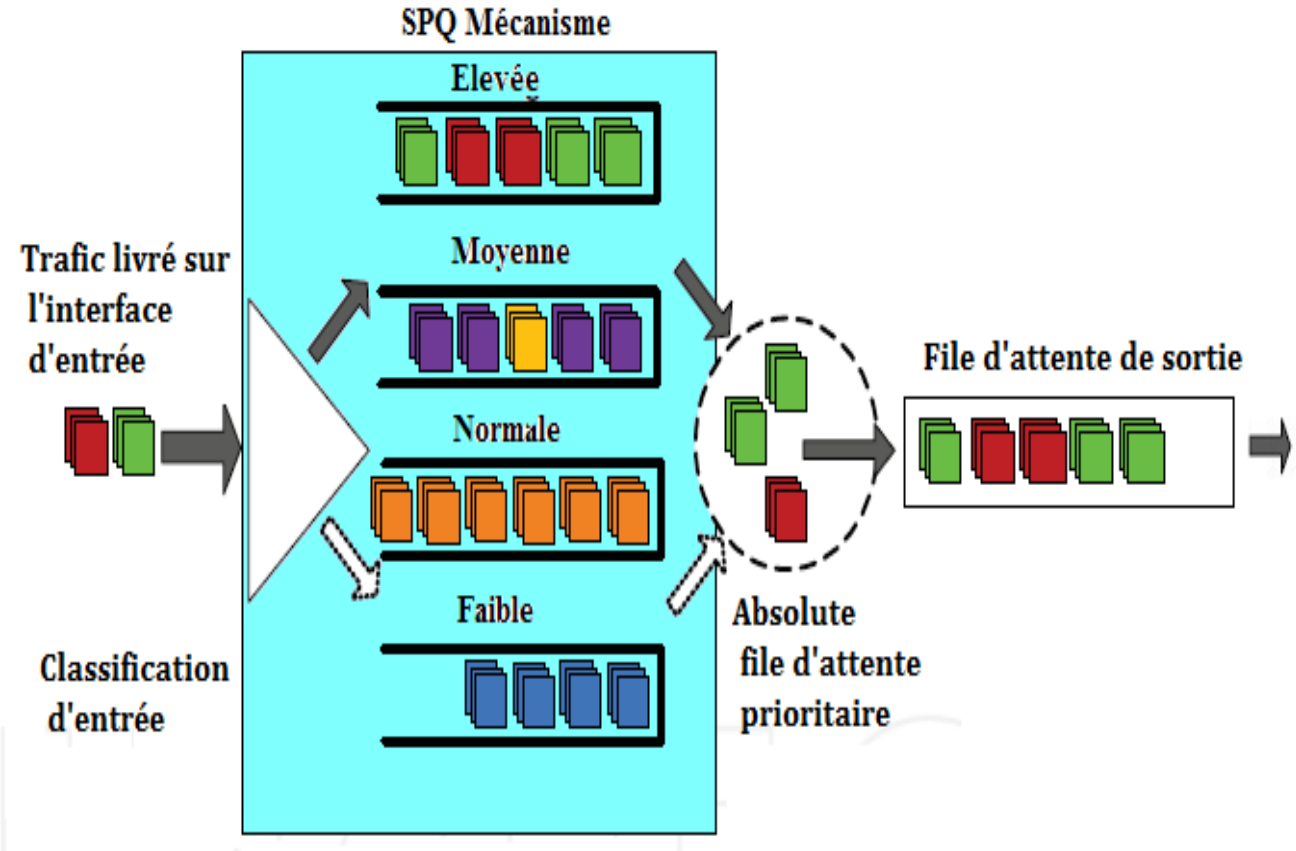


Figure 6 : l'ordonnancement SPQ.

I.6.6Fair Queuing (FQ)

Fair Queuing a été proposé par John Nagle en 1987. FQ fonctionne de façon très similaire à RR. Il classifie d'abord les paquets en fonction des flots auxquels ils appartiennent. Par la suite, il insère chaque flot dans une file d'attente propre à ce flot. Ensuite chaque file est parcourue octets par octets. Lorsqu'un paquet a fini d'être parcouru, celui-ci est inséré dans la file de sortie,comme présenté dans la (figure 7)**.** Cette façon de faire est beaucoup plus

équitable que le Round Robin puisque les classes ayant des plus petits paquets ne sont pas désavantagées.

L'objectif est de fournir équitablement le même taux de service aux différents flux partageant les ressources.

a) Le principal avantage de FQ est qu'un flux très sporadique ou mauvaise conduite ne dégrade pas la qualité de service rendu aux autres flux, parce que chaque flux est isolé dans sa propre file d'attente**.**

b) FQ implique également plusieurs limites:

- FQ est sensible à l'ordre des arrivées de paquets. Si un paquet arrive dans une file d'attente vide immédiatement après la visite de l'ordonnanceur, le paquet doit attendre dans la file d'attente jusqu'à ce que toutes les autres files d'attente aient été traité avant qu'il puisse être transmis. [6][10]

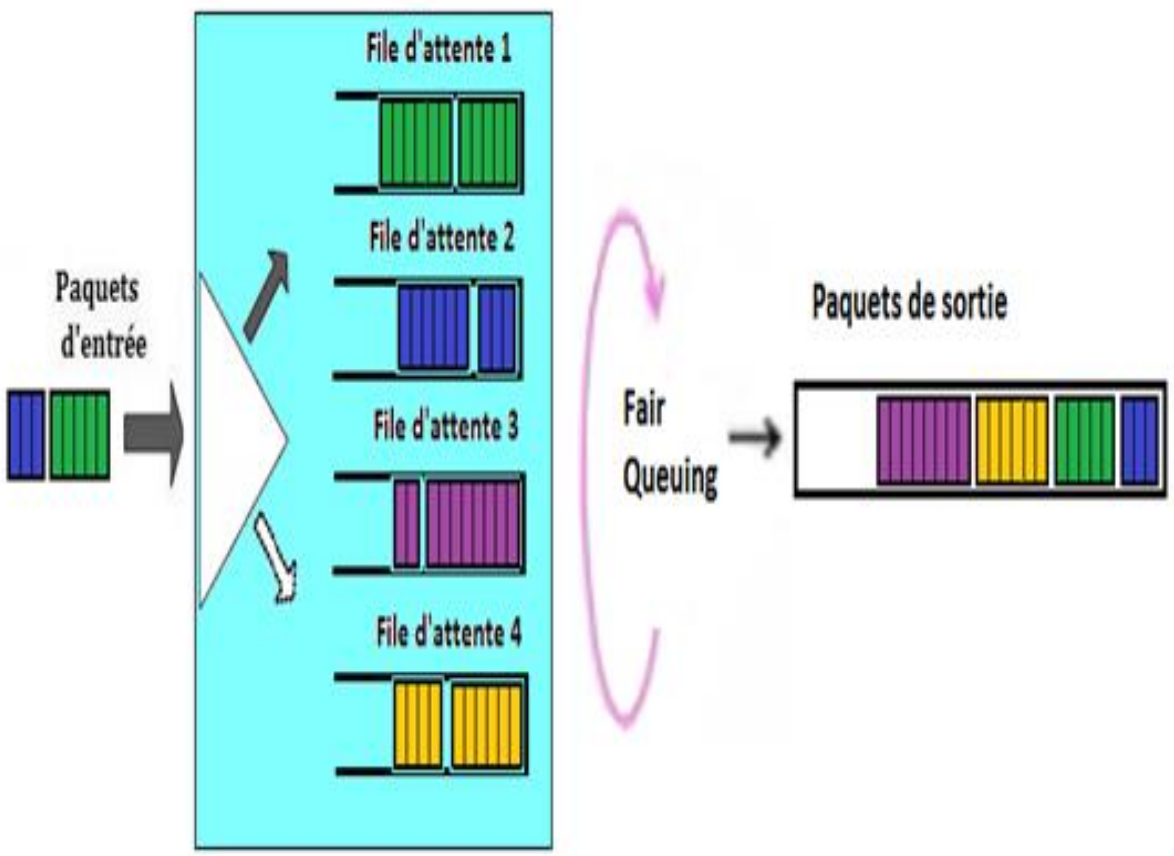


Figure 7: l'ordonnancement FQ.

I.6.7 Weighted Fair Queuing (WFQ)

Fair Queuing pondérée (WFQ) a été développé indépendamment en 1989 par Zhang Lixia et par Alan Demers, Srinivasan Keshav, et Scott Shenker.

WFQ permet de donner une priorité pondérée à certains flux, il permet de séparer le trafic arrivant dans un routeur en plusieurs flux, et d'allouer à chaque flux une portion de la bande passante. Le routeur va créer dynamiquement une file par flux détecté.

Contrairement aux mécanismes à priorités, WFQ tente de donner un pourcentage de la bande passante à des classes de service. En fait, au lieu de compter le nombre de paquets servis et de faire un service de type Round Robin pondéré sur les paquets, WFQ va faire de manière microscopique un Round Robin sur les bits d'un paquet.

a) Les avantages de cette approche sont donc :

- WFQ fournit une protection à chaque classe de service en garantissant un niveau minimum de bande passante indépendamment des autres classes de service.

b) Malgré ces avantages plutôt intéressants, on pourra noter :

- Le coût élevé au niveau du routeur (ordonnancement + calcul des priorités dynamiques).
- WFQ est coûteux en ressources CPU car le routeur doit calculer à chaque émission combien de paquets de chaque file seront émis.
- Impacts de la complexité des calculs sur l'évolutivité de WFQ en tentant de traiter un grand nombre de classes de service sur les interfaces à grande vitesse. [8][10]

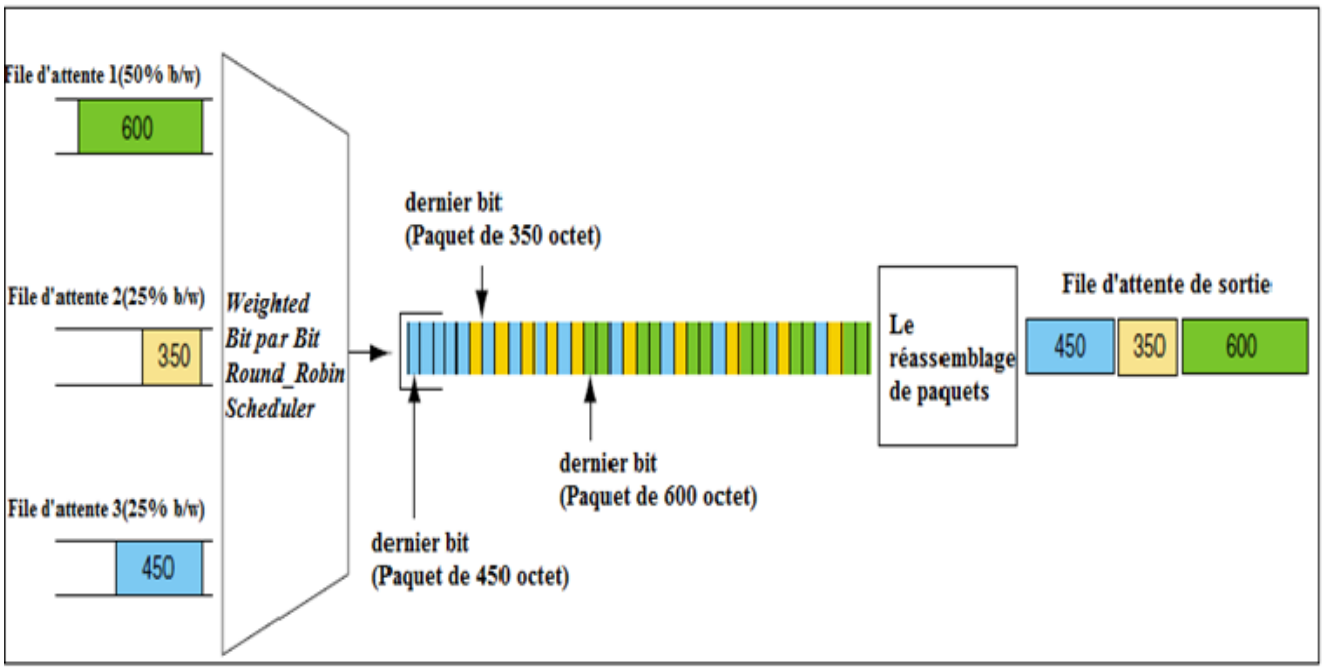


Figure 8: l'ordonnancement WFQ.

La figure 8 présente un ordonnanceur round-robin bit par bit pondérée gardant trois files d'attente.

Supposons que la file d'attente 1 est attribuée 50 %de la largeur de bande du port de sortie et que la file d'attente 2 et 3 sont chacune attribuées 25 % de la largeur de bande.

L'ordonnanceur transmet deux bits à partir de la file d'attente 1, un bit de la file d'attente2, et un bit de la file d'attente 3, puis retourne à la file d'attente1.

A la suite de cet ordonnancement pondérée, le dernier bit du paquet 600 octets est transmis avant le dernier bit du paquet de 350 octets, et le dernier bit du paquet 350 octets est transmis avant le dernier bit du paquet 450 octets. Cela provoque la séquence de transmission suivante : le paquet de 600 octets, ensuite le paquet de 350 octets, et enfin le paquet de 450 octets.

I.6.8 Custom Queuing (CQ)

La file d'attente personnalisée (CQ) assigne un certain pourcentage de la bande passante à chaque file d'attente pour assurer un débit prévisible pour les autres files d'attente. Il est conçu pour les environnements qui doivent garantir un niveau minimal de service pour l'ensemble du trafic. [3]

CQ se réserve la quantité de bande passante garantie à un point de congestion possible sous la forme d'un rapport constant d'assurance bande passante, tandis que le reste de la bande passante disponible est laissé à d'autres le trafic réseau. La gestion du trafic est effectuée par la procédure de répartissant selon l'espace libre dans la file d'attente pour chaque classe de paquets. [5]**[**SIT03**]**

CQ est composée de 17 files d'attente numérotées de 0 à 16. La file d'attente 0 est une file d'attente de système réservé pour le mécanisme et par conséquent ne peut pas être réglé par l'utilisateur.

CQ consiste à attribuer un pourcentage de la bande passante disponible à chacune 16 des files d'attente jusqu'à atteindre 100% de la disponibilité totale. Les files d'attente sont servis d'une manière circulaire comme l'algorithme WRR.

CQ est une solution est un peu meilleur pour éviter la congestion de QoS mais n'est pas encore idéale car insuffisamment dynamique. [3]

a) Les avantages de CQ sont :

1. Simple, rapide.
2. Priorité relative en fonction d'une demande de bande passante.

b) Les inconvénients de CQ sont :

1. Priorités statiques.
2. Comment bien répartir la bande passante.
3. Seulement 16 niveaux de priorité.[5]**[**SIT03**]**

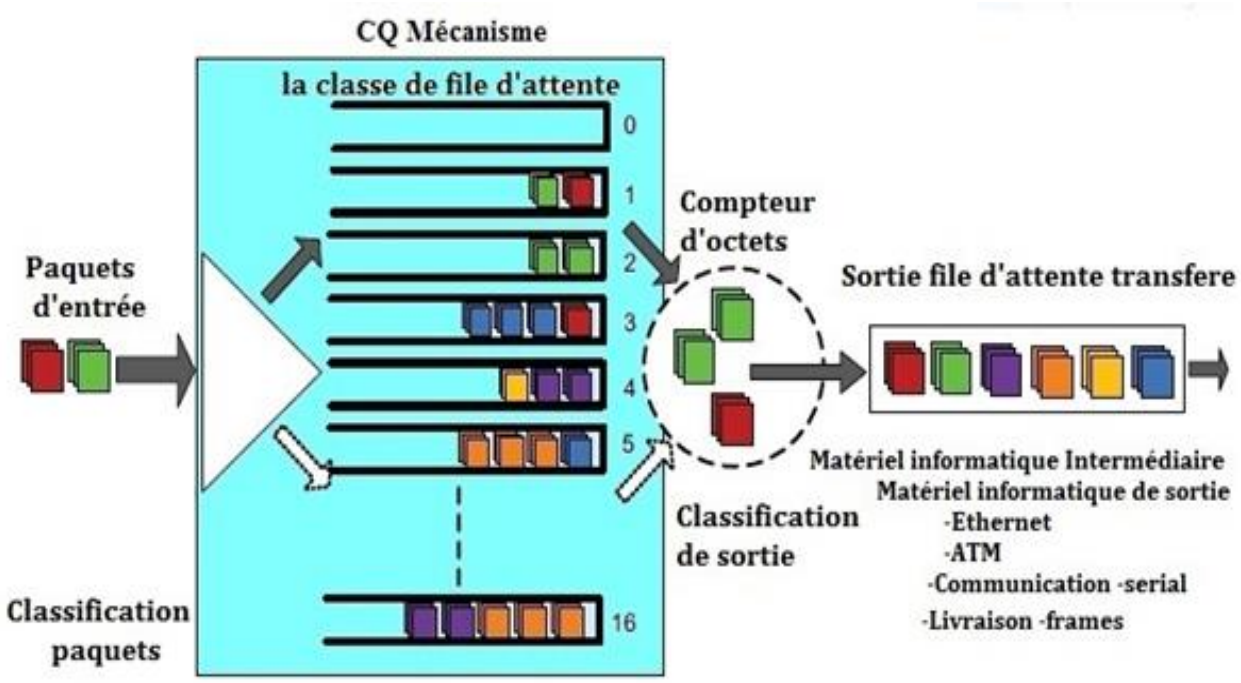


Figure 9: l'ordonnancement CQ.

I.6.9 Classe-Basé Weighted Fair Queuing (CBWFQ)

Le Fair Queuing pondéré de Classe-Basé (CBWFQ) est un outil moderne de gestion congestions, et il fournit une meilleure flexibilité dans l'octroi d'une quantité de bande passante minimum sur la base de la juste file d'attente ainsi que sur la base des classes définies par l'administrateur. Au lieu de fournir une file d'attente pour chaque flux de données, les classes définies par l'administrateur du réseau sont utilisé. Si le flux de trafic correspond à une classe défini par l'administrateur, il est immédiatement placé dans telle classe, où il a déjà une bande passante de liaison réservée. Si la circulation ne correspond pas à l'une des classes définis par l'administrateur, il ne peut utiliser que le lien restant de la bande passante, qui n'est pas réservé à une autre catégorie. Pour chaque classe définie une largeur minimale de bande nécessaire est garantie. Pour cela, nous pourrions fournir un exemple concret, où le mécanisme CBWFQ est très utile pour éviter les situations où plusieurs flux de faible priorité pourraient déborder la haute priorité.

La disponibilité de la bande passante peut être définie en spécifiant:

- Bande passante (en kbps).
- Pourcentage de largeur de bande (en pourcentage de la bande passante disponible sur interface de sortie).
- Pourcentage de la bande passante restante disponible.

a) Avantage :

1. Classifications personnalisées définies.
2. Allocation de bande passante minimale.
3. Granularité plus fine et l'évolutivité.

b) Inconvénient :

1. Le trafic voix peut encore subir des retards inacceptables. **[**SIT01**][**8**]**

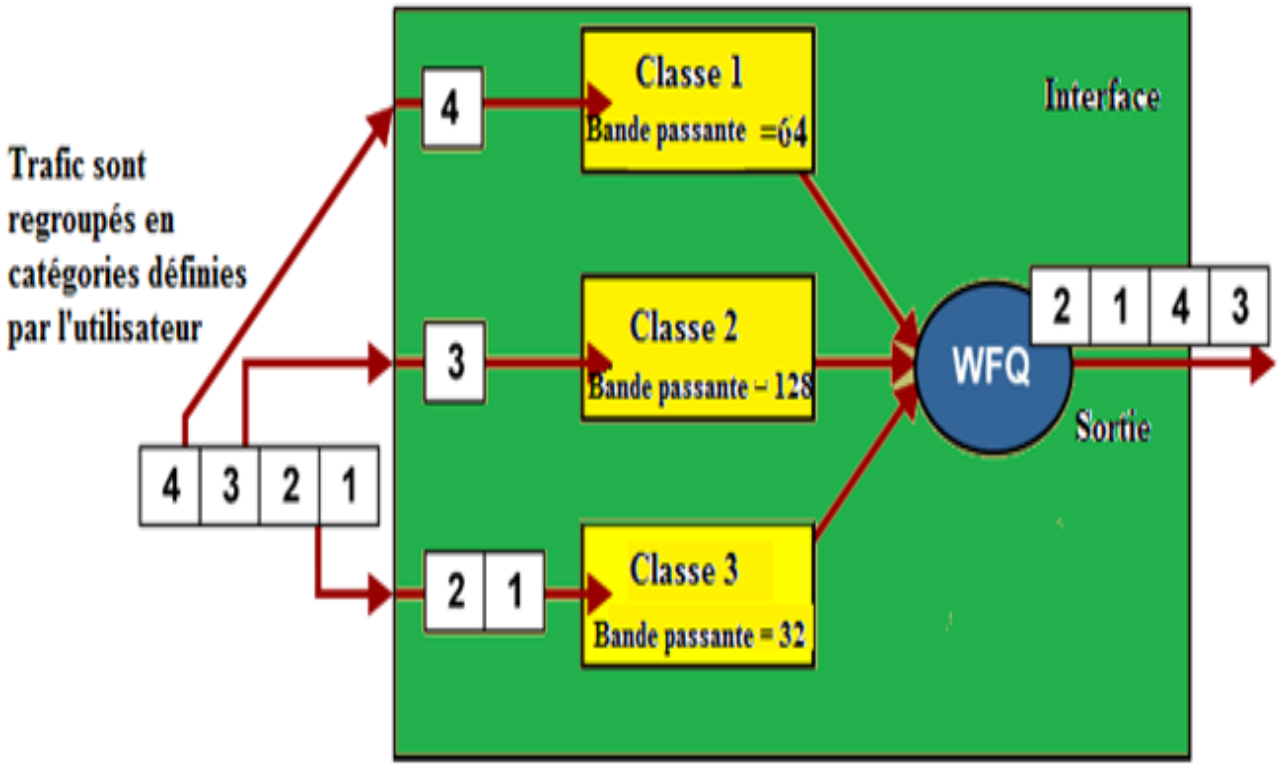


Figure 10: l'ordonnancement CBWFQ

I.6.10 Low Latency Queue (LLQ)

Low Latency Queuing (LLQ) est combine la fonction de réservation de bande passante de CBWFQ avec une file d'attente haute priorité de SPQ. Il est conçu exclusivement pour les applications voix et vidéo. L'objectif est d'offrir un délai minimum pour des trafics critiques (files d'attente prioritaire), tout en ayant un partage des ressources pour les autres trafics. [3]

L'ordonnanceur LLQ vérifie toujours la file d'attente à faible latence d'abord, et prend un paquet de cette file d'attente.

La file d'attente prioritaire est servie comme dans l'algorithme de priorité fixe(SPQ). Les files d'attente à partage de bande passante sont alors considérées comme une seule file ayant la plus petite priorité. Lorsque la file prioritaire est servie, l'algorithme CBWFQ est appliqué aux files d'attente restantes.

En faisant cela, les paquets dans la file d'attente qui sont transmises encore ont une très faible latence, mais LLQ empêche également le trafic à faible latence de consommer plus de son montant configuré de la bande passante. [SIT01] [SIT04]

Nous concluons LLQ C'est le modèle que nous utiliserons pour évaluer les critères de QoS dans la boucle d'optimisation. [3]

Latence : est le temps que met un paquet pour arriver d'un ordinateur à un autre ordinateur. On parle de temps de latence ou période de latence pour désigner le délai entre une action et le déclenchement d'une réaction, à savoir un retardement. On comprend bien que plus la latence (le temps) est faible, meilleure est la connexion. Lors des tests, la latence s'exprime en millisecondes (ms).

a) Avantages LLQ :

1. Classes de haut-prioritaires sont garantis:
   - Faible latence propagation de paquets.
   - Bande passante.
2. Configuration et le fonctionnement sont conformes à tous les types de médias.
3. Définit limite de confiance pour assurer une classification simple et l'entrée à une file d'attente. [SIT01] [SIT04]

b) Inconvénient de LLQ :

1. Malheureusement la complexité interne de cette approche un coût très élevé au niveau de l'OS du routeur. [8]

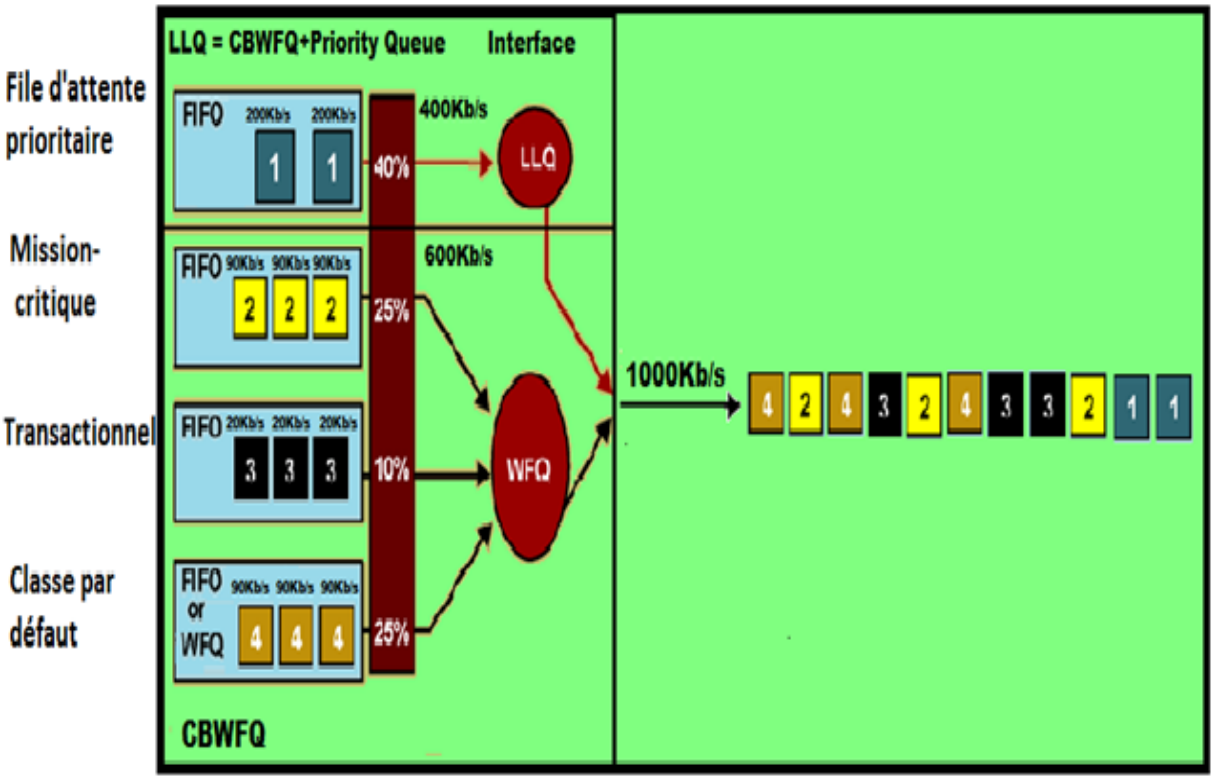


Figure 11: l'ordonnancement LLQ.

I.7 Conclusion

A travers ce chapitre, nous avons présenté le principe de l'ordonnancement des paquets et différents algorithmes proposés

Nous distinguons pour cela différents catégories d'ordonnancement :
Il y a des politiques qui offrent une garantie sur un paramètre de qualité de service (soit le délai, soit le débit) comme les ordonnanceurs basés sur la priorité (SPQ par exemple), les autres algorithmes tentent de fournir un partage des ressources équitable entre les flots concurrents (FQ par exemple).

Chapitre II : Implémentation et Réalisation

II.1 Introduction

Dans ce chapitre nous présentons l'environnement de programmation et les outils de développement choisis pour la réalisation de notre application, et nous donnons aussi une vue générale sur l'application.

II.2 Environnement du projet

Notre application est essentiellement codée en Java. Pour rappel, l'intérêt du langage Java vient principalement du fait que l'application peut être exécutée indépendamment de la plateforme grâce à la machine virtuelle JVM.

Java SE Développement Kit (JDK) est un pack d'outils pour le développement d'application via le langage Java. Il a les composants nécessaires à la conception et au test de projets avec diverses caractéristiques.

Figure 12: Logo Java

II.3 Langages et technologies utilisés

II.3.1 L'Eclipse

Pour la réalisation de notre projet on a choisi l'IDE Eclipse (la version Juno 4.2) comme un environnement de programmation. L'éclipse est simple à utiliser, il permet de développer, compiler et exécuter aisément un programme java.

Cet outil combine les fonctionnalités d'un éditeur de texte, d'un compilateur et d'un débugger. Il rend plus pratique la programmation et il dispose en général d'une documentation sur les outils et méthodes du langage concerné et facilite l'accès aux propriétés des librairies communes.

Figure 13 : Fenêtre principal d'Eclipse.

II.3.2 Edition d'interfaces : WindowBuilder (GWT)

Comme cité précédemment, l'application sera développée en langage Java, le code sera édité avec l'IDE Eclipse. Nous utiliserons du Swing/SWT pour générer les interfaces graphiques, en exploitant WindowBuilder plugin de Google Web Tools (GWT) permettant de faciliter l'édition des interfaces.

WindowBuilder est un plugin pour l'édition des interfaces graphiques sous JAVA.
le plugin a été offert à la fondation Eclipse sous licence open-source. Il est compatible avec Eclipse ainsi que tous les IDEs qui sont basés sur JAVA (RAD, RSA, MyEclipse, JBuilder…).

WindowBuilder est considéré l'un des meilleurs outils pour la construction d'IHMs multiplateformes, et s'avère extrêmement stable, Son utilisation est simple, l'outil permet la création de formulaires basiques tout comme des fenêtres complexes, générant derrière le code java des composants utilisés.

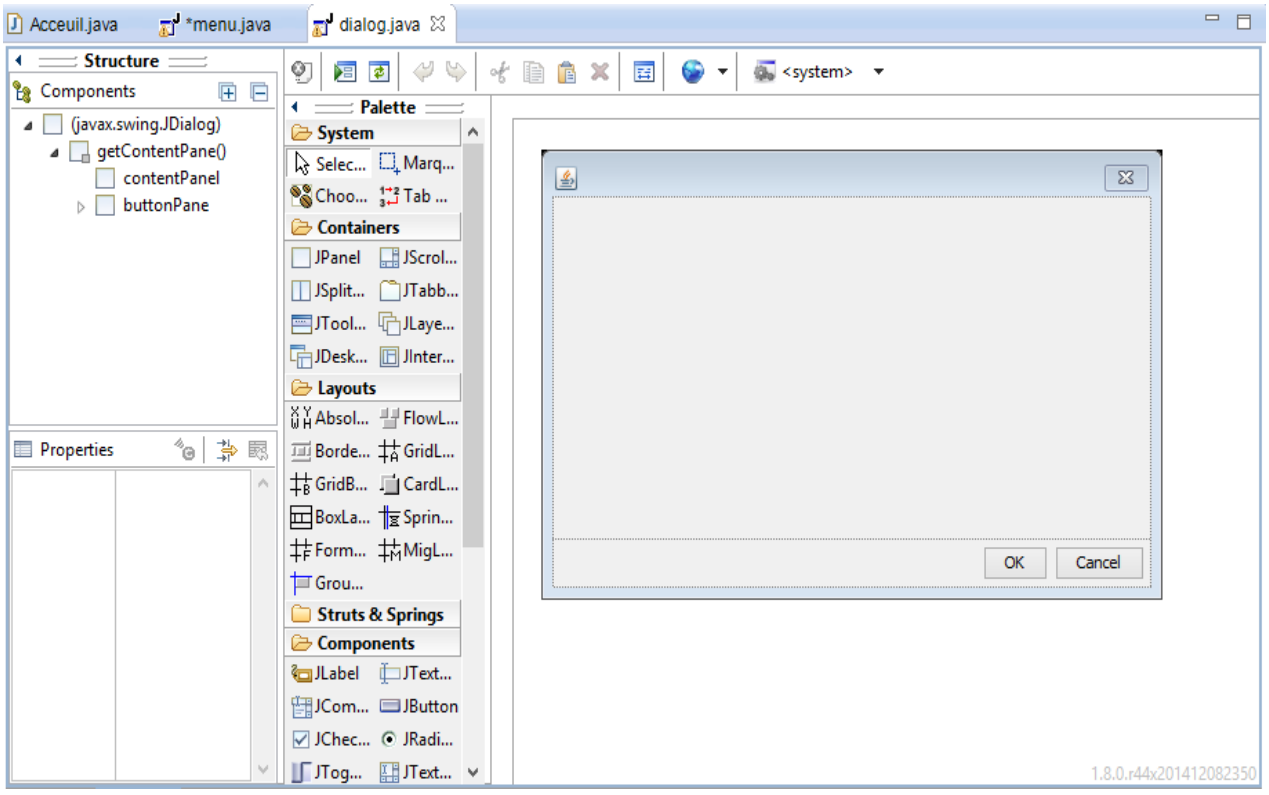


Figure 14 : Interface WindowBuilder

II.4 Présentation de l'application

Notre application est composée de plusieurs fenêtres chacune a une opération spécifique**.**

II.4.1 Fenêtre principal

Cette interface constitue la fenêtre d'accueil de notre application qui contient plusieurs choix d'algorithmes :

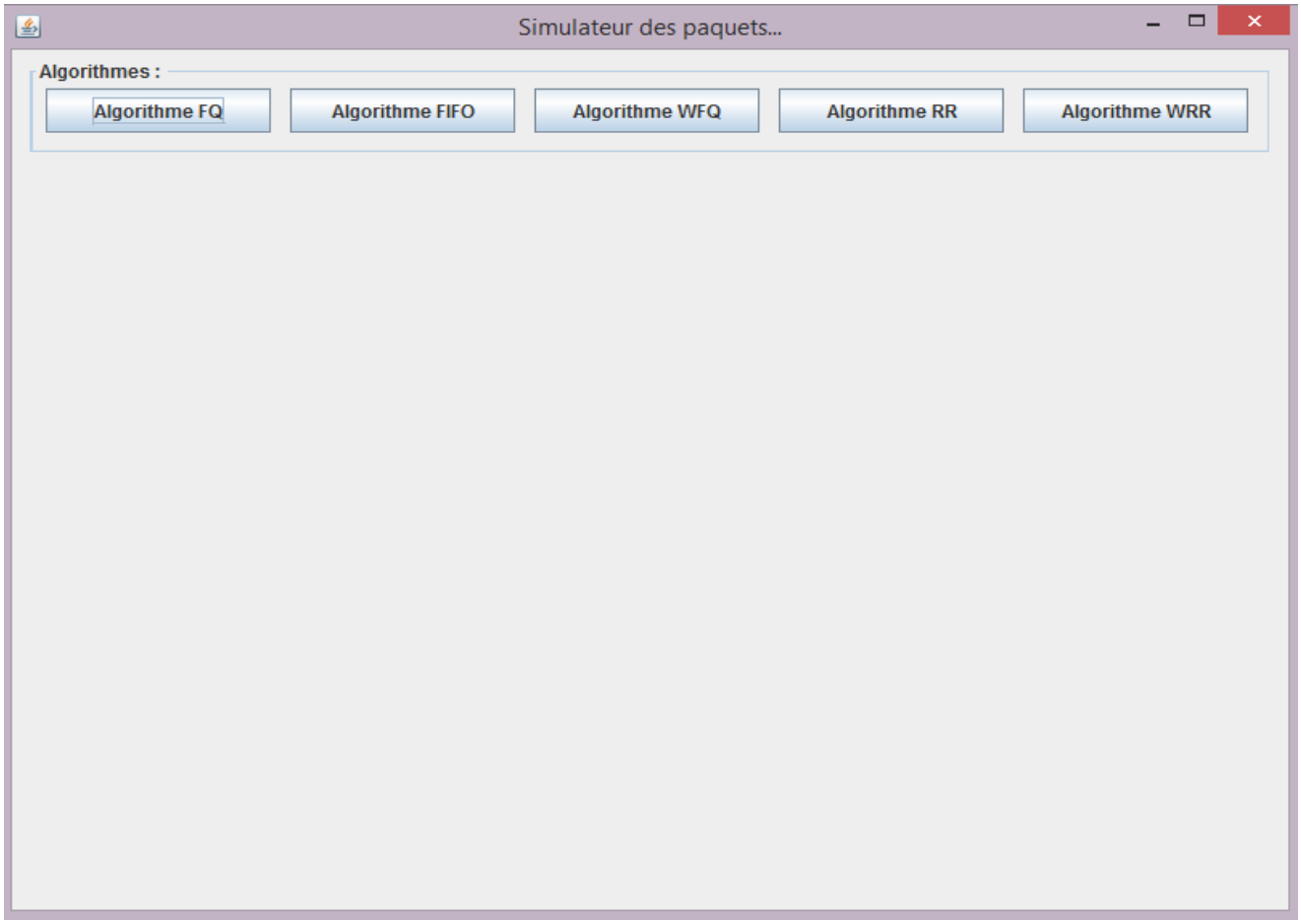


Figure 15 : Fenêtre Menu principal

II.4.2 Initialisation

Après le choix d'un algorithme, on obtient cette fenêtre qui permet de remplir les champs suivant :

Nbr file : on a fixé seulement 3 files.

Nbr paquet par file : nombre de paquet de chaque file.

K(T.E.octet) : on suppose que l'ordonnanceur traite un octet toutes les **k**=1ms.

Le butons initialiser : permet de sauvegarder les valeurs déjà saisies et affiche la partie de chargement des paquets.

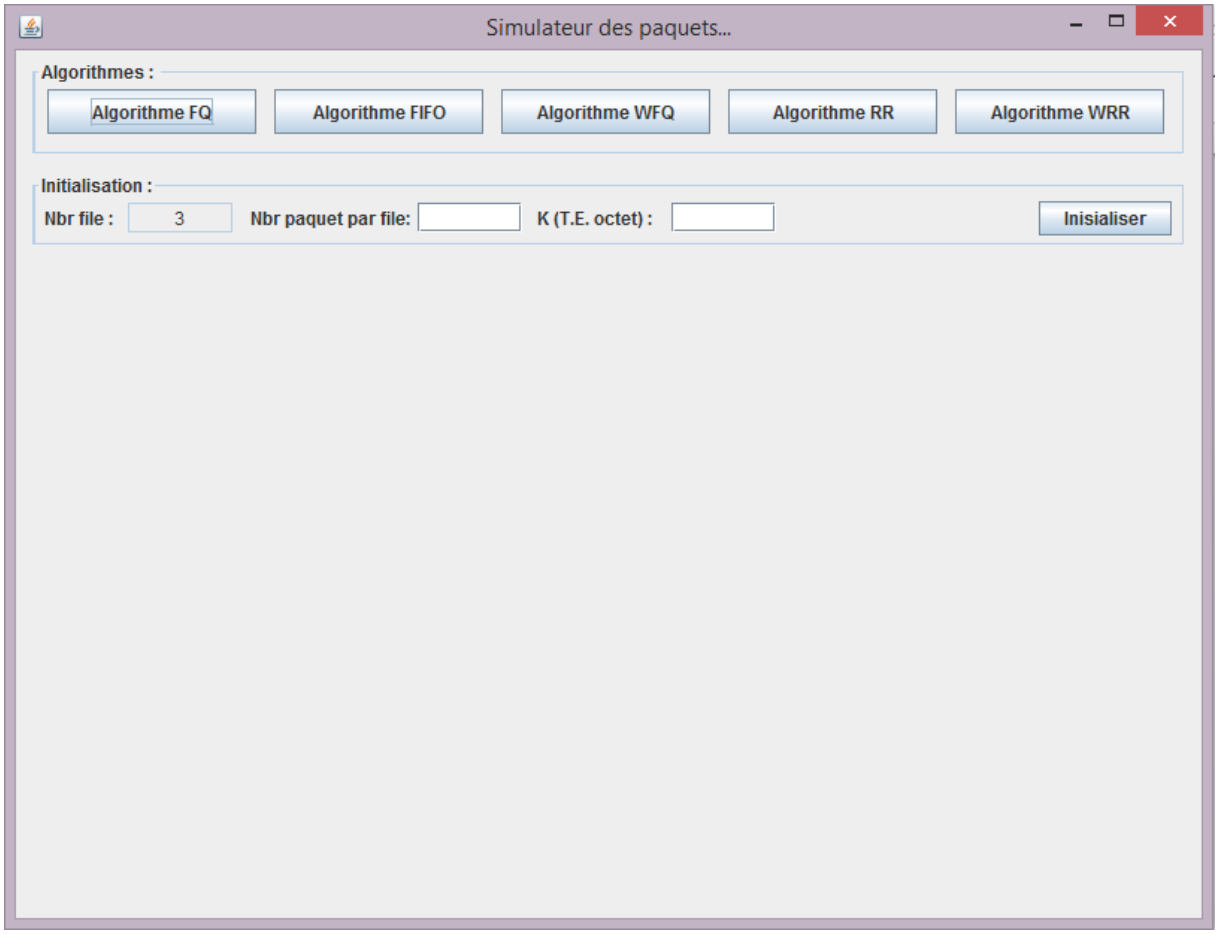


Figure 16 : Fenêtre Initialisation

II.4.3 Chargement des paquets

Apres l'initialisation, on passe à la partie de chargement des paquets qui permet de remplir les informations du paquet (nom, taille, temps d'arrivée).

Le butons Ajouter : pour ajouter les informations du paquet dans le tableau.

Le butons Résultat : afficher l'ordre de sortie des paquets.

Le butons nouveau : affiche le menu principale.

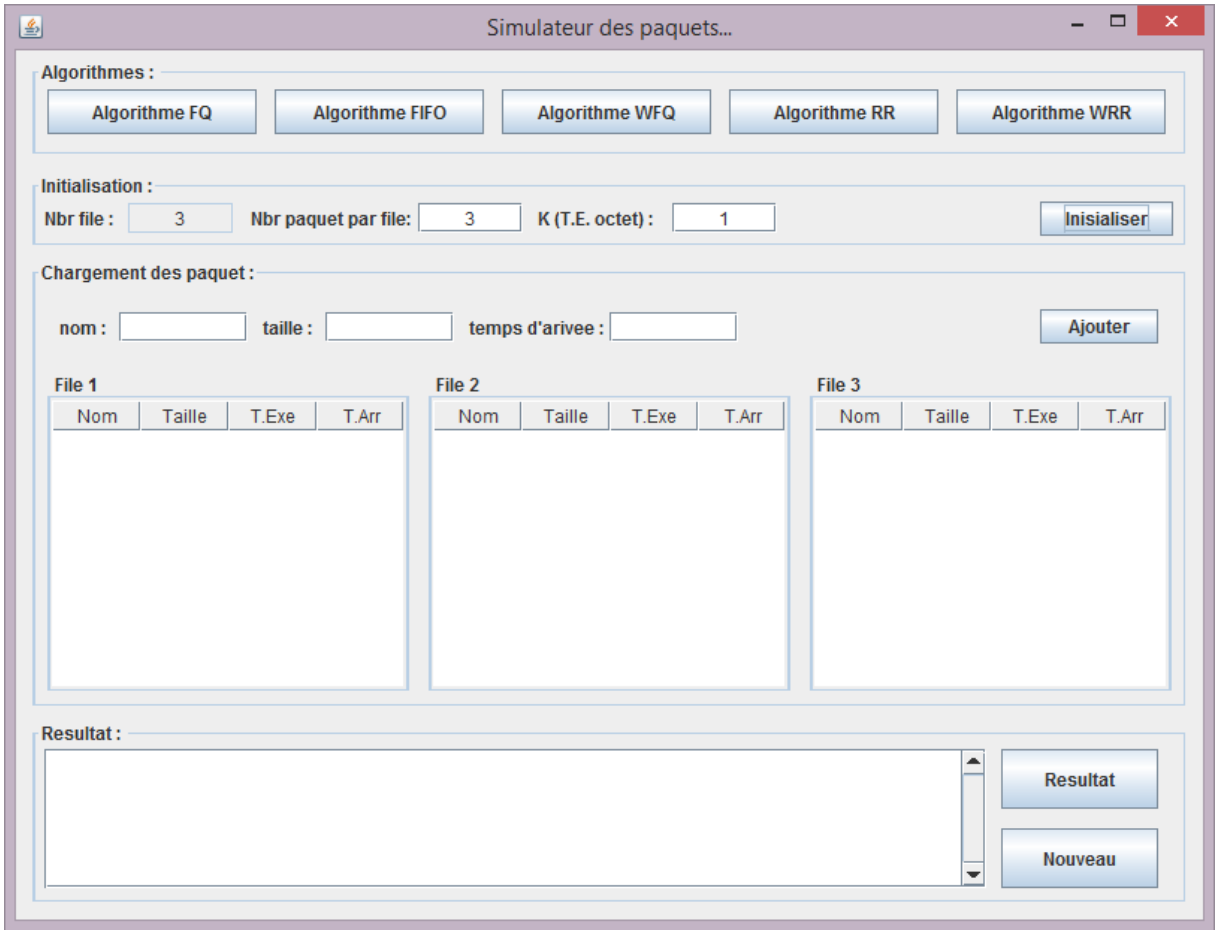


Figure 17 : Fenêtre chargement des paquets

II.4.4 Fenêtre Algorithme FQ

Cette fenêtre contient le résultat fourni par le programme lorsque nous choisissons l'algorithme d'ordonnancement Fair Queuing « **FQ** ».

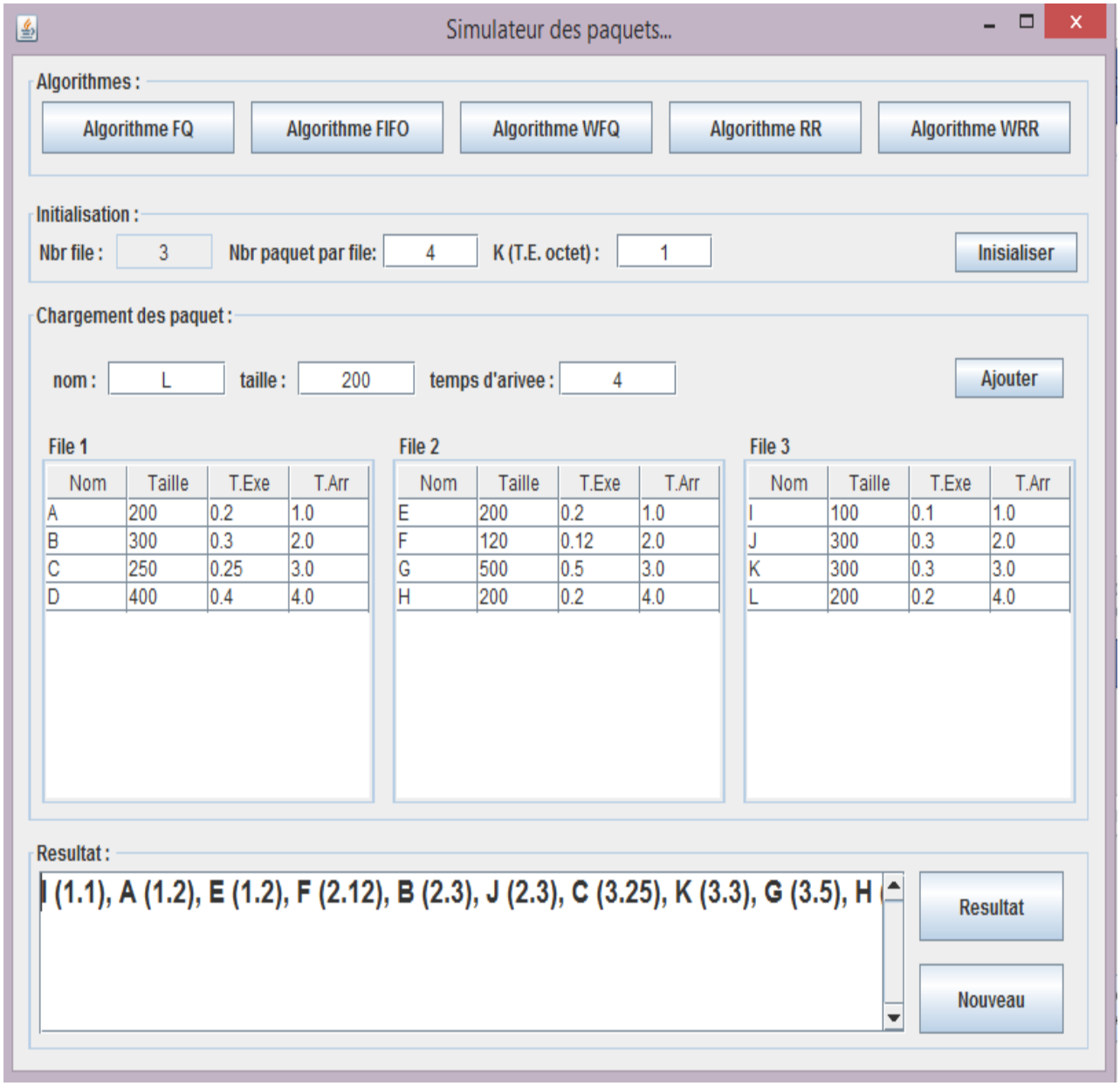


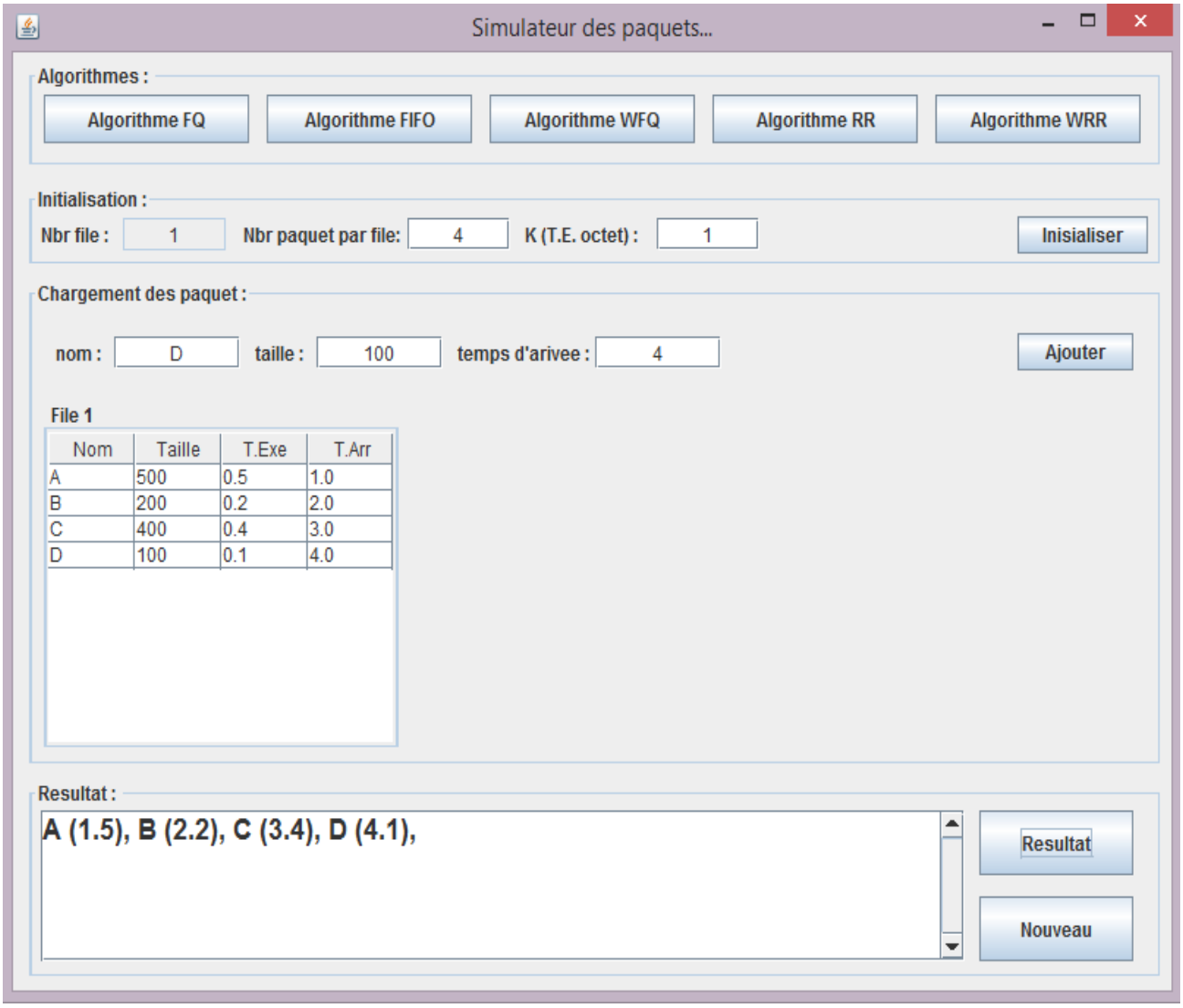
Figure 18 : Fenêtre Algorithme FQ

II.4.5 Fenêtre Algorithme FIFO

Cette fenêtre contient le résultat d'un algorithme first in first out « **FIFO** », on a

Qu'une seule fille d'attente.

Figure 19 : Fenêtre Algorithme FIFO



II.4.6 Fenêtre Algorithme WFQ

Cette fenêtre contient le résultat fourni par le programme lorsque nous choisissons l'algorithme d'ordonnancement Weighted Fair Queuing« **WFQ** ».

La fenêtre est diffèrent à des autres fenêtres car en ajoute le champ de « **poids file**» à la partie de l'initialisation, qui nous permet de choisie la file qui a un poids deux fois important que les autres files.

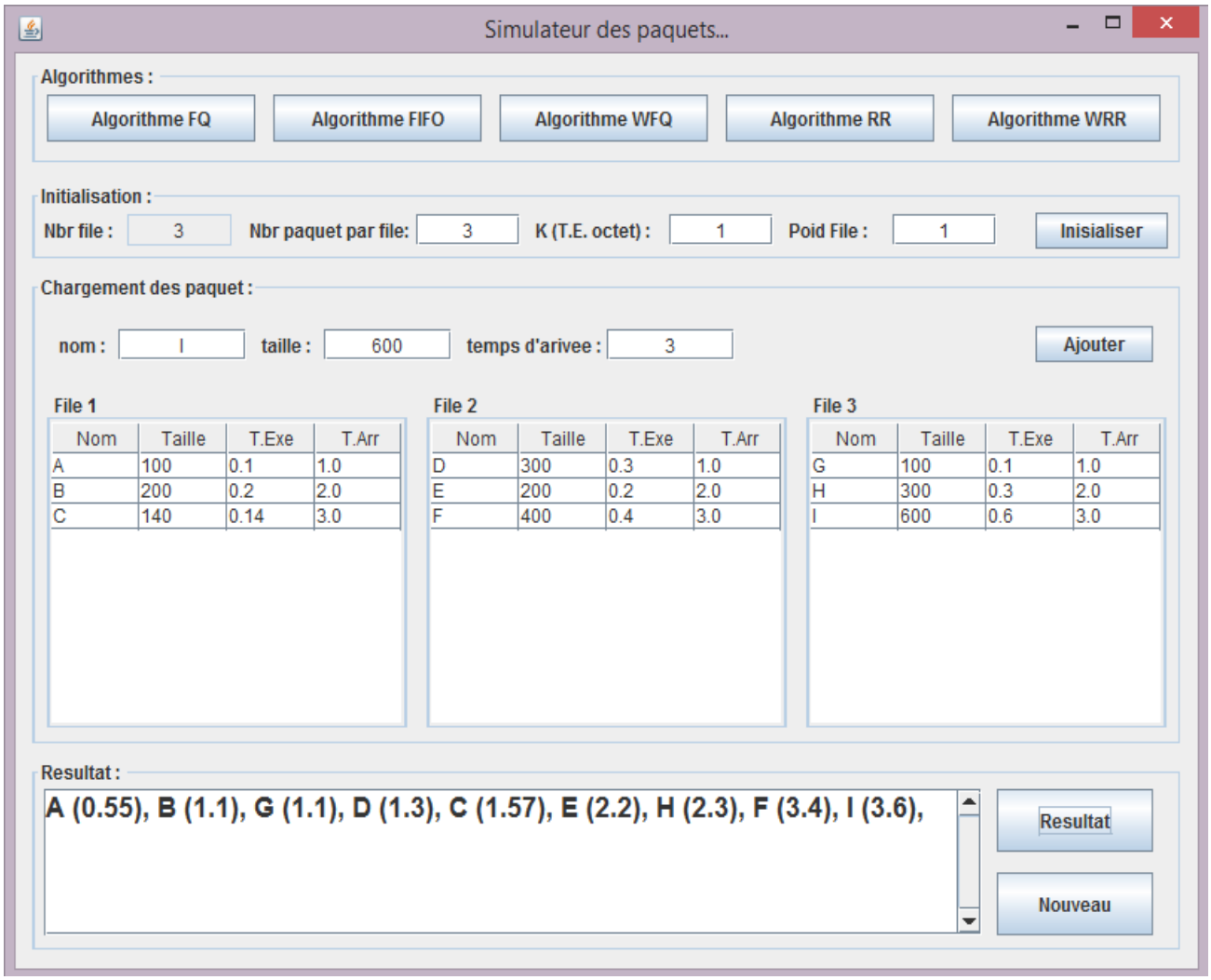


Figure 20 : Fenêtre Algorithme WFQ

II.4.7 Fenêtre Algorithme RR

Cette fenêtre contient le résultat fourni par le programme lorsque nous choisissons l'algorithme d'ordonnancement Round Robin«**RR** ».

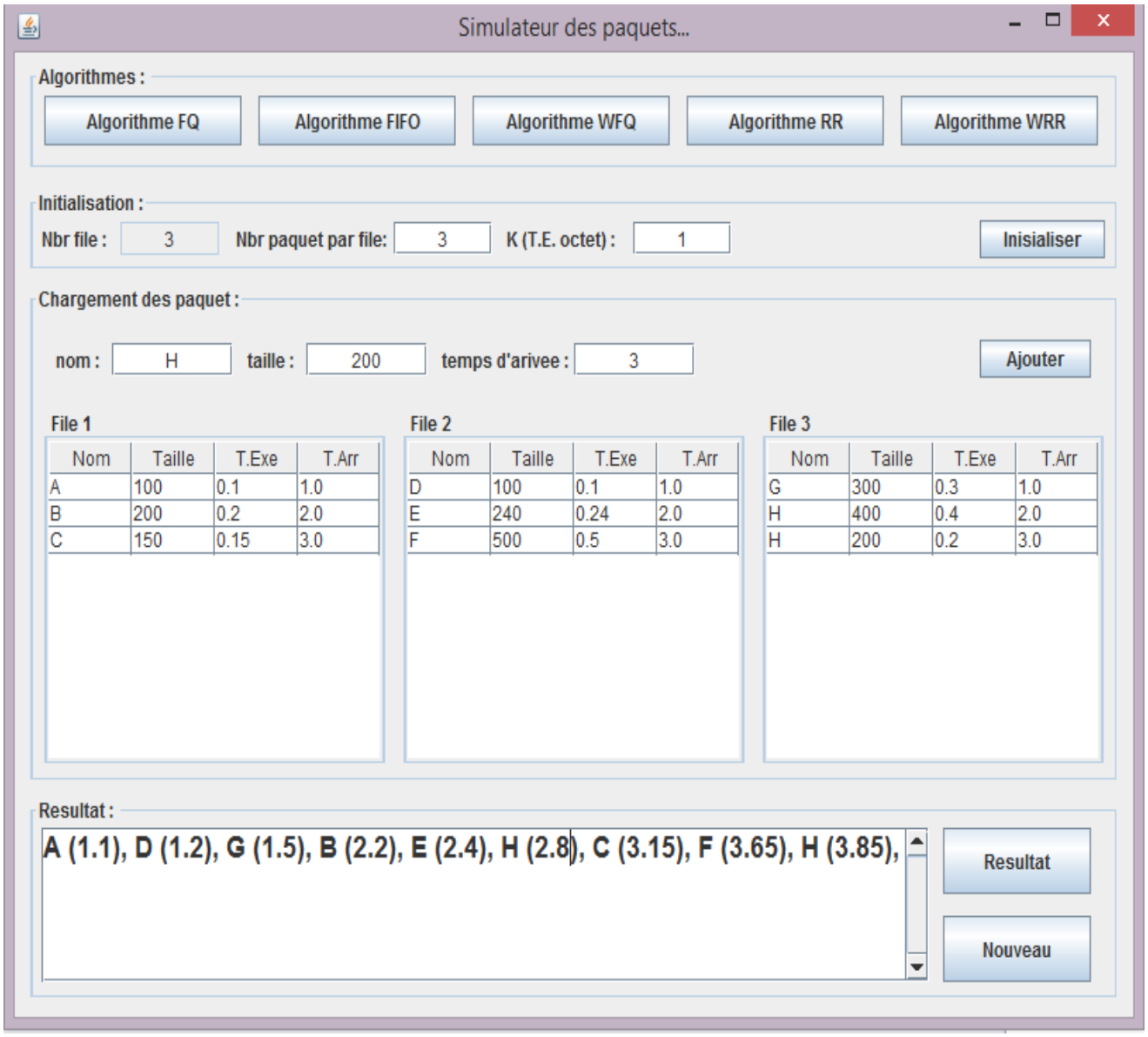


Figure 21 : Fenêtre Algorithme RR

II.4.8 Fenêtre Algorithme WRR

Cette fenêtre contient le résultat fourni par le programme lorsque nous choisissons l'algorithme d'ordonnancement Weighted Round Robin « **WRR** ».

La fenêtre est diffèrent à des autres fenêtres car en ajoute le champ de « **poids file**» à la partie de l'initialisation, qui nous permet de choisir la file qui a un poids deux fois important que les autres files.

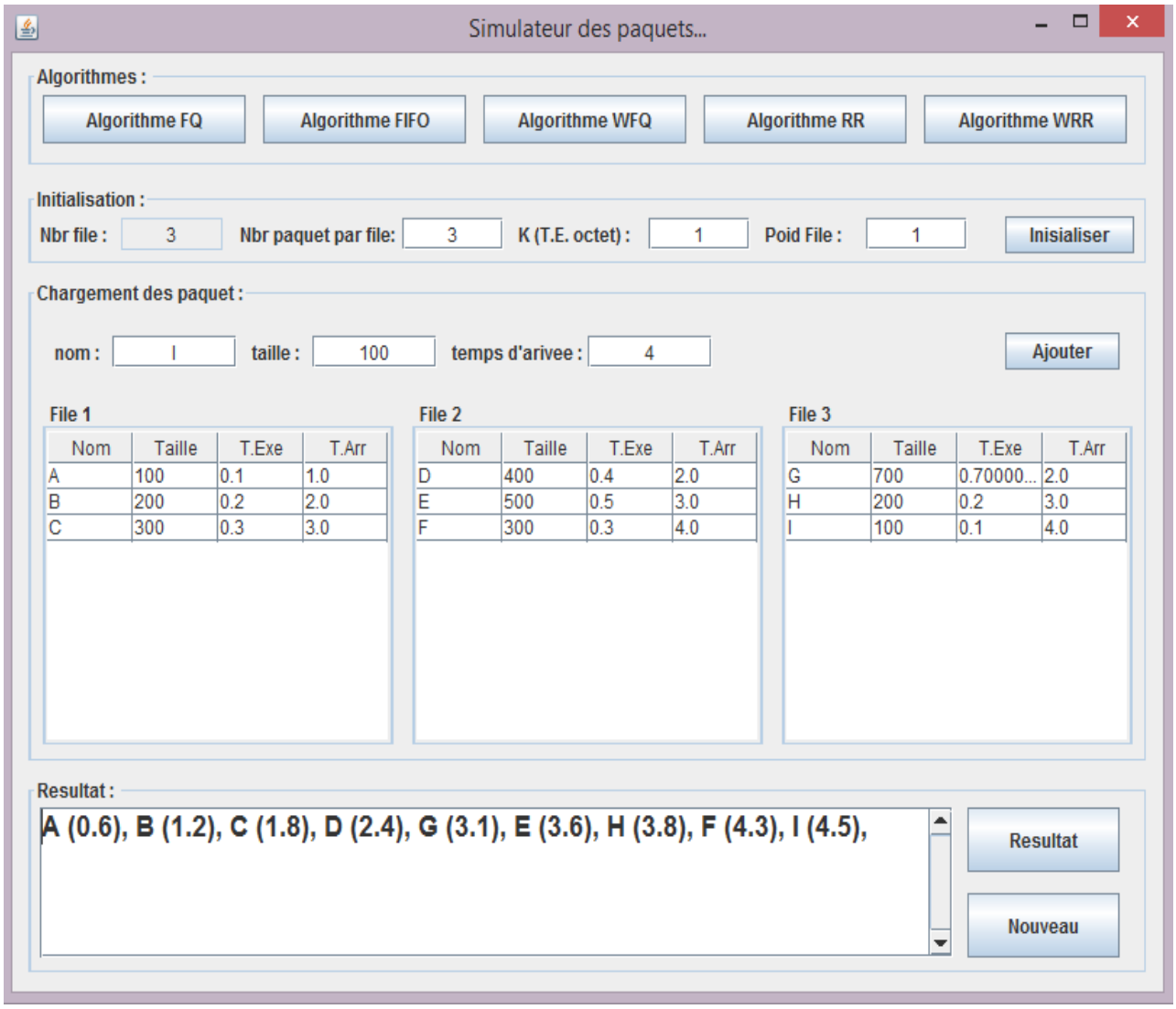


Figure 22 : Fenêtre Algorithme WRR

II.5 Conclusion

Nous avons présenté au cours de ce chapitre l'implémentation de notre système. Afin de compléter notre apprentissage théorique sur chaque algorithme d'ordonnancement, nous avons créé un programme codé en java qui nous affiche l'ordre et les temps de sortie des paquets.

Conclusion Générale

L'objectif de cette thèse a été l'étude de politiques d'ordonnancement de paquets IP pour un routeur d'internet.

Nous avons défini les trois types d'ordonnanceurs :

Ordonnanceurs à Priorité Fixe,

Ordonnanceurs basés sur le paradigme temps partagé généralisé

Ordonnanceurs basés sur la notion de trame temporelle

Nous avons ainsi présenté, les algorithmes d'ordonnancement sous un aspect général, en distinguant les mécanismes conçus pour les files d'attente simples, et ceux adaptés pour les files d'attente multiples.

Bibliographie

[1] : **Leila Toumi** . « Algorithmes et mécanismes pour la qualité de service dans des réseaux Hétérogènes Networking and Internet Architecture » Conférence. Institut National Polytechnique de Grenoble - INPG, 2002.

[2] : Ye-Qiong Song . « Garantir la qualité de service temps réel : ordonnancement et gestion de files d'attente ». La 5_eme Ecole d'été Temps Réel - ETR 2007, Sep 2007, Nantes, France. pp.257-275.

[3] : Mohamed Anouar Rachdi. « Optimisation des ressources de réseaux hétérogènes avec cœur de réseau MPLS ». INSA de Toulouse, 2007.

[4] : David GAUCHARD. « Simulation hybride des réseaux IP-DiffServ-MPLS multi-services sur environnement d'exécution distribuée. Université Paul Sabatier » - Toulouse III, 2003.

[5] : Sasa Klampfer, AmorChowdhury, JozeMohorko and ZarkoCucej. « Influences of Classical and Hybrid Queuing Mechanisms on VoIP's QoS Properties, VoIP Technologies Conference ». (2011)

[6] : Marc-André Breton. « Développement D'un Système De Surveillance Des Mécanismes De Qualité De Service Dans Le Contexte Des Réseaux De Prochaine Génération Montréal », École De Technologie supérieure Université Du Québec. Le 06 juillet 2006.

[7] : ARTAUD Franck| EYMARD Louis GASPARD Anthony. « Implémentation d'algorithmes de qualité de service dans un routeur ». 2013.

[8] : Alexandre SIMON. « Principes de mécanismes de Qualité de Service, mesures de performances et mises en œuvre sur Lothaire » .Centre Interuniversitaire de Ressources Informatiques de Lorraine.

[9]:Octavio Napoleón MEDINA CARVAJAL . « Etude des algorithmes d'attribution de priorités dans un Internet à Différentiation de Services ».

[10] :Chuck Semeria. « WhitePaper Queue Scheduling .Supporting Differentiated Service Classes »

[11] :Hazar Aouad. « Transport de Flux Temps Réels dans un Réseau IPMobile.l'Ecole Nationale Supérieuredes Télécommunications ». 20 Janvier 2005.

Webographie

[SIT01] :http://blog.globalknowledge.com/technology/unified-communications/qos-11-cbwfq/

[SIT02] : http://projets-gmi.univ-avignon.fr/projets/proj1213/M1/p01/data/documents/qos1.pdf.

[SIT03] : http://slideplayer.fr/slide/442758.

[SIT04] :http://mynetworkingwiki.com/index.php/Low_Latency_Queuing_%28LLQ%29.