

**Introduction à la
problématique des
Réseaux avec QoS
-Service Level Agreement,
Service Level Specification-**

Ingénierie des Réseaux d'Entreprise (Cycle C),
Compléments Réseaux de Transport et Application (Cycle B)

Janvier 2002

Eric Gressier-Soudan

Problématique à résoudre

Spécifier des contraintes de QoS (performances, mais on pourrait facilement ajouter sécurité et disponibilité) au niveau applicatif pour en déduire les contraintes sur l'architecture réseau !

Besoins :

Modèle de Description d'Application

Modèle de Spécification de QoS

Modèle d'Architecture

Spécification en langue naturelle :

"J'ai besoin d'une bande passante de 10Mb/s entre la caméra à Paris et l'écran vidéo à Toulouse"

Opérations de Gestion de la QoS

- **Spécification de la QoS:** création d'un contrat entre l'application, et l'environnement d'exécution (**SLS**)
Négociation de la QoS: en vue d'obtenir un accord entre utilisateur et fournisseur (**SLA**)
- **Contrôle d'Admission :** tests qui déterminent si le système est capable de supporter le contrat requis
Réservation de Ressources: pour garantir le contrat accepté
- **Surveillance de la QoS :** surveillance par l'utilisateur du respect des contraintes de QoS qui ont été garantie par le fournisseur, un grain de surveillance doit pouvoir être indiqué 100ms par exemple
Vérification de QoS : respect du contrat de QoS par l'utilisateur
Maintenance de la QoS : actions prises par le fournisseur en cas de défaut constaté sur la QoS garantie
- **Renégociation de QoS :** si la maintenance ne parvient pas à rétablir le niveau de service demandé, l'utilisateur doit pouvoir renégocier son contrat

Modèle de Spécification pour les applications : RM-ODP

RM-ODP: Reference Model of Open Distributed Processing

définit

une Architecture des Systèmes Répartis Ouverte
et Hétérogène

Modèle Générique

fondé sur la notion de **Points de vue** (aspects d'analyse et spécification) :

- **Entreprise** (besoins des applications, contraintes opérationnelles et organisationnelles)
- **Information** (modèle des données)
- **Traitements** (modèle des traitements sous forme d'Objets qui interagissent)
- **Ingénierie** (objets, OS, protocoles, liens réseau, contraintes sur l'infrastructure logicielle et matérielle ...)
- **Technologie** (implantation et conformité à la spécification...)

Approche Orientée Objet

Une plate-forme de type CORBA satisfait le Point de Vue de l'Ingénierie de RM-ODP. Elle serait aussi relativement conforme au Point de Vue des Traitements.

Rappels sur l'Orienté Objet

Objet :

- Etat + Données Internes (inaccessibles de l'extérieur)
- Opération = point d'accès à une fonction exécutable par un objet
- Interface regroupe des opérations, sert à la désignation des Opérations

Signature d'une opération : nom d'opération + paramètres
+ type du résultat

Un objet n'évolue que par l'utilisation des opérations de son interface.

Propriétés des objets :

- . Encapsulation
- . Héritage
- . Polymorphisme
- . Agrégation

Avantages :

- . Séparation entre spécification et implantation
- . Modularité et Extensibilité
- . Réutilisabilité des composants
- . Adéquation des abstractions : Analyse, Spécification, Conception, Programmation

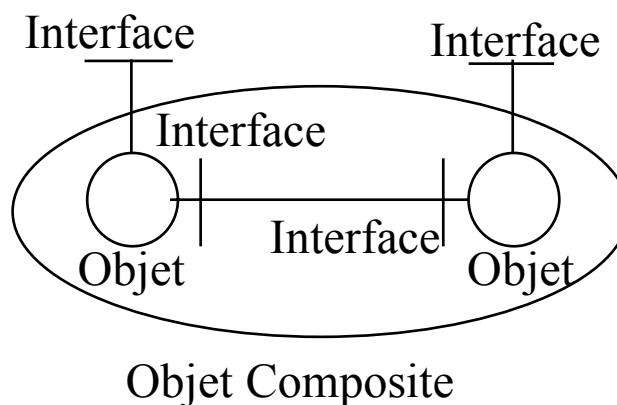
Modèle Objet d'ODP

Modèle objet des langages de programmation
+ plusieurs interfaces par objet
+ notion d'objet composite (réparti)

une interface = une vue abstraite de l'objet (concept de rôle)

exemple : une interface d'accès aux opérations, une interface pour les opérations de gestion

Schématisation d'un objet RM-ODP :

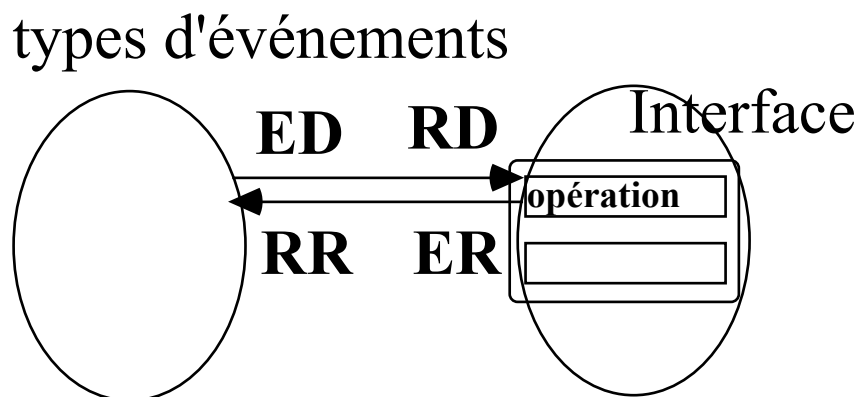


Modèle des Traitements de ODP

Repose sur la notion d'Objets en environnements répartis.

Application = ensemble d'objets

Interaction = Invocation d'Opération, ou Réaction d'Invocation décrite à l'aide d'événements



Chaque événement a :

- un Type,
- un Nom,
- une Valeur

Spécification de Contraintes de QoS

Modèle des Traitements :

=> Pouvoir Spécifier des Comportements et des Propriétés

=> Vérifier les Spécifications

Règles de Conformités exprimées sur les événements, donc sur les interfaces et sur les communications entre objets

Exemples :

- règle comportementale : ED et RR sur un objet Client
- règle temporelle : le délai entre RD et ER est Δ

Qualité de Service (QoS):

- taux d'erreur message,
- taux d'erreur bit,
- débit,
- garantie de délai

...

On a bien une notion de **Contrat** associée à une interface.

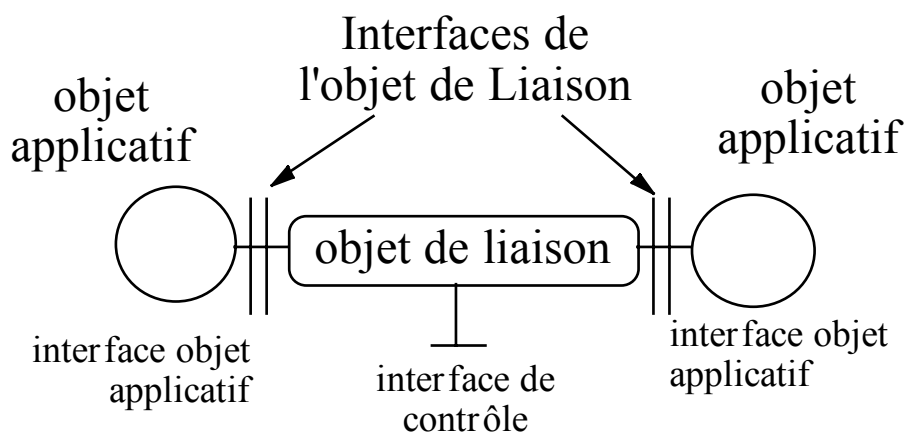
La QoS garantie par un objet implique la possibilité de supporter des propriétés de QoS par l'environnement d'exécution.

Liaisons entre Objets

Une Liaison correspond à l'abstraction du mécanisme qui supporte une interaction entre Objets.

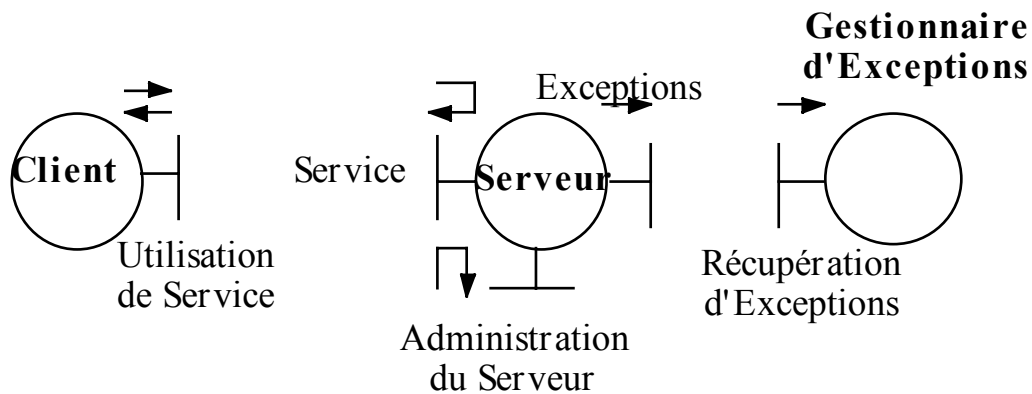
Les contraintes de QoS sur les interactions entre Objets imposent des contraintes de QoS sur les Liaisons.

Les Liaisons deviennent des Objets avec une Interface de Contrôle qui permet de les commander, de les superviser, de les configurer.



On voit ici l'extension de la notion de connexion ou d'association qu'on trouvait dans les couches du modèle ISO.

Modèle d'objets de base



correspond aux objets dans CORBA

Il existe des objets particuliers qui ont la charge d'ajouter des nouveaux objets à l'environnement. Ces objets particuliers s'appellent des **fabriques à objets** (object factory).

Les objets sont fabriqués à partir de modèles (template) complètement spécifiés y compris le contrat de QoS, c'est un processus d'instanciation.

Extensions Multimédia du Modèle Objet

Interface production et consommation de flux



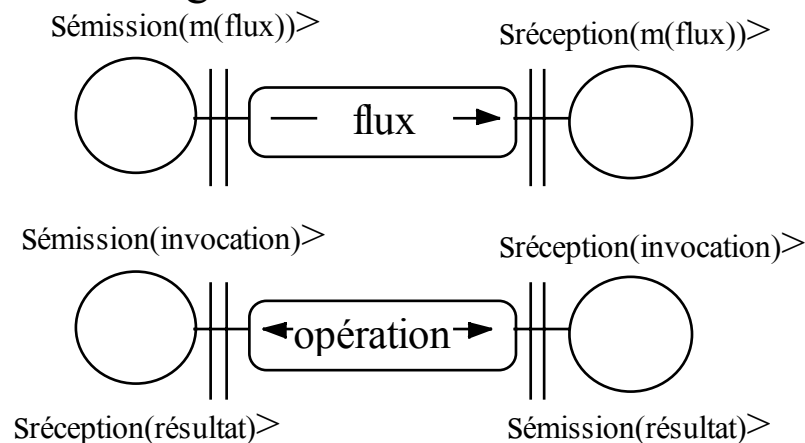
Une interface de type flux supporte des interactions multimédia de type continu, elle est définie en terme d'un ou plusieurs flots de données. Chaque flot est caractérisé par un nom, un type de media géré, et la direction du flot.

Interface signaux ou événements temps réel



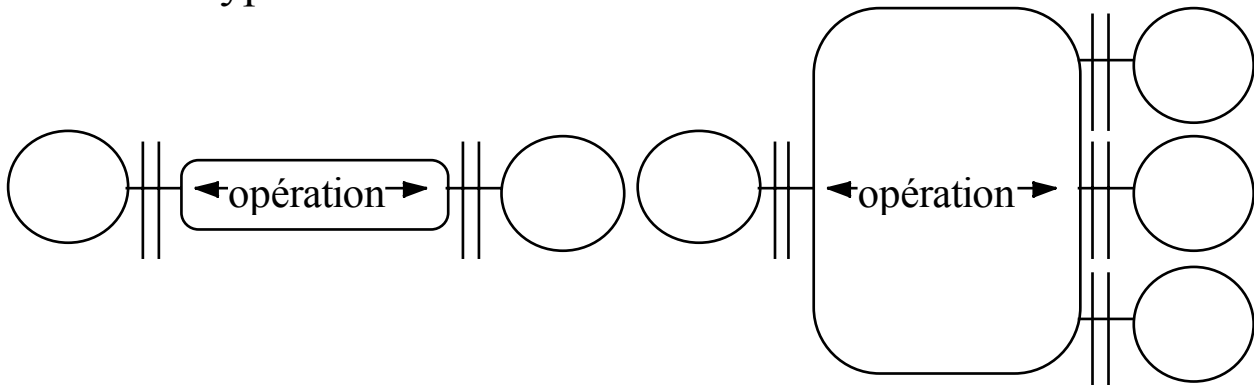
Les interfaces de type signal sont utilisées pour la gestion de la QoS et la synchronisation temps réel. Chaque signal dans un interface est représenté par un nom, le type, sa valeur, et la causalité induite (générateur, consommateur).

Les signaux peuvent être utilisés aussi bien pour des flux que pour des services, dans chaque cas ils représentent l'émission ou la réception d'un message.

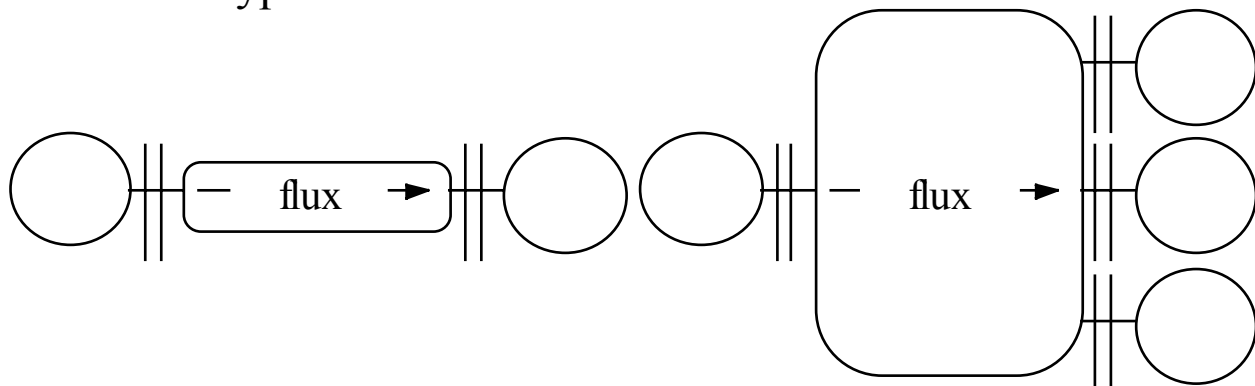


Modèles de Liaisons entre Objets

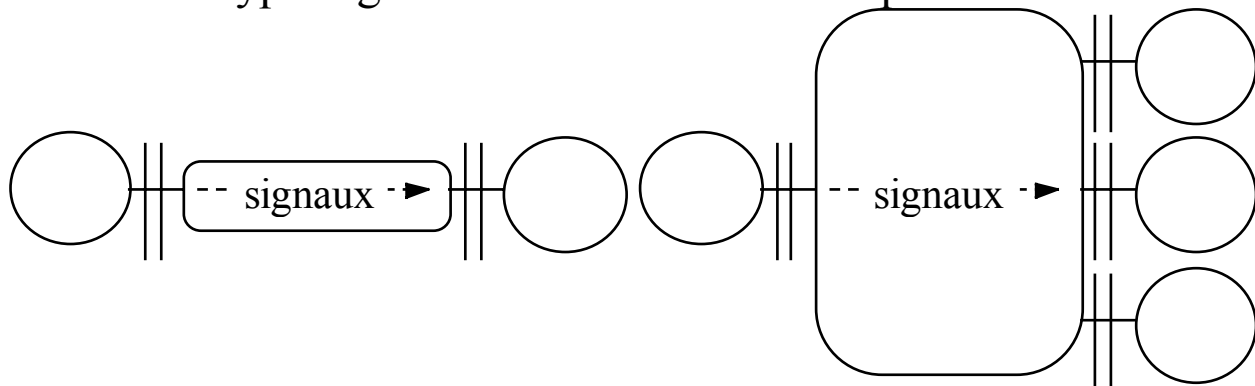
Liaison de type Service



Liaison de type Flux



Liaison de type Signaux ou Evénements Temps Réel



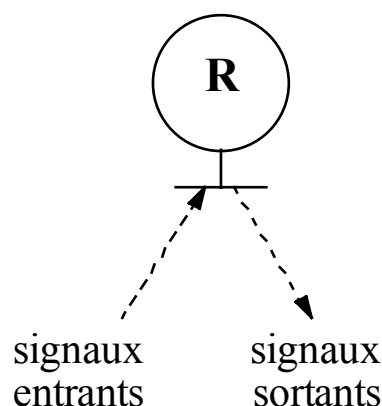
Objets réactifs

Les objets présentés sont des objets asynchrones. Objets ou objets de liaison, ils prennent un certain temps pour exécuter leur traitement.

La spécification des contraintes de QoS temporelle sert à contraindre le comportement des objets, et à édicter des contraintes environnementales sur l'architecture.

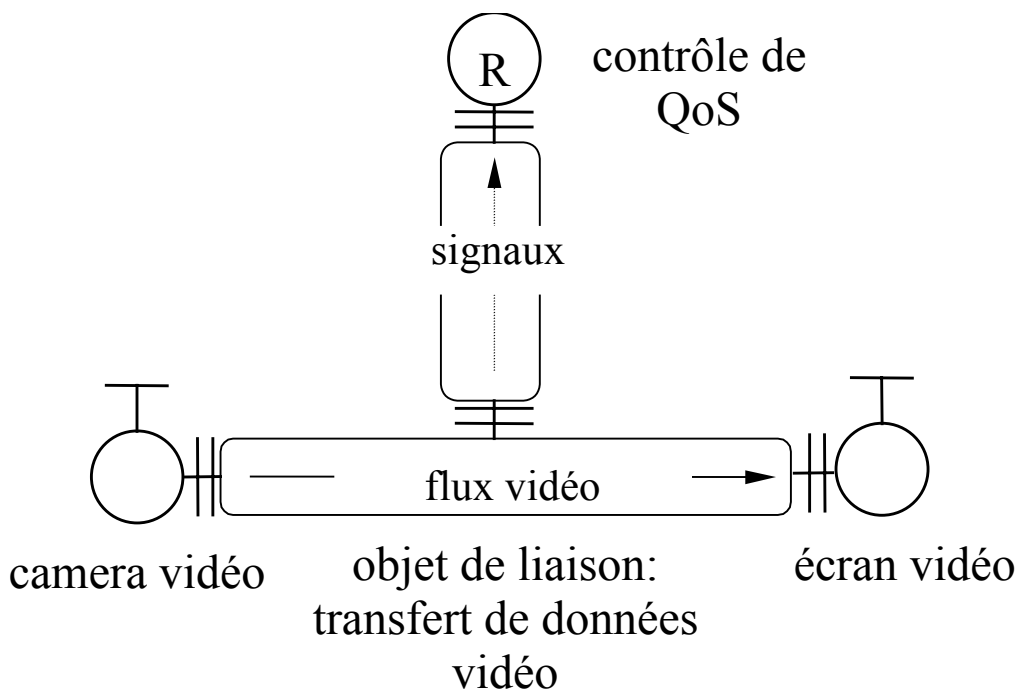
Les Objets Réactifs ont un rôle particulier, ils servent à contrôler le comportement d'objets, en particulier à observer l'évolution de la QoS ou à gérer la synchronisation temps réel. En ce sens, ils reçoivent des signaux, les traitent et envoient des signaux.

Les objets réactifs ont un comportement synchrone. Ils s'exécutent en un temps nul (hypothèse de modèle, en réalité c'est un temps borné inférieur à une période d'activation). Ils sont programmés en langage synchrone (LUSTRE, SIGNAL, ESTEREL).



Exemple de schématisation d'une application Multimédia

Exemple de modélisation :



Echange Vidéo avec Moniteur de QoS

Temps Réel Multimédia

$$\text{Temps Réel Multimédia} = \text{QoS} + \text{Contrôle}$$

Objets réactifs (à exécution immédiate) =>Contrôle

Objets applicatifs ou système (asynchrones contraints par une spec de QoS) => Traitements

Avantages :

- . Les contraintes temporelles sont formalisées explicitement par des équations de QoS
- . Claire séparation entre le contrôle et l'application
- . Portabilité, seule les équations de QoS doivent être recalculées avec un nouvel environnement

Equations de QoS

La spécification d'équations de QoS (ou plutôt d'inégalités) est fondée sur un modèle événementiel (suppose donc un temps discret). Exemples avec expression déterministe :

Latence bornée d'un transfert d'images vidéo :

pour tout n , (n représentant l'occurrence d'un événement) :

$$|\text{date}(\text{réception-image-video},n) - \text{date}(\text{émission-image-video},n)| \leq \text{latence}$$

Gigue d'un transfert d'images vidéo :

pour tout n :

$$\min \leq |\text{date}(\text{réception-image-video},n) - \text{date}(\text{émission-image-video},n)| \leq \max$$

$$\text{gigue} = \max - \min$$

Débit max d'un transfert d'images vidéo :

pour tout n :

$$|\text{date}(\text{réception-image-video},n+k) - \text{date}(\text{réception-image-video},n)| \leq \text{durée}$$

$$\text{débit} = k/\text{durée}, \text{ en images par seconde}$$

Taux de perte d'un transfert vidéo :

pour tout n :

$$(|\text{date}(\text{émission-image-video},n+k) - \text{date}(\text{émission-image-video},n)| \leq d) \text{ et } (|\text{date}(\text{réception-image-video},n+k') - \text{date}(\text{réception-image-video},n)| \leq d) \text{ et } (k' \leq k)$$

$$\text{taux de perte } 1 - (k'/k)$$

Exemple (1)

Dans la réalité un événement est repéré par :

nom-d'une-interface.nom-d'un-signal.causalité avec causalité pouvant prendre les valeurs (ED/ES¹,RD/RS,[ER,RR] conformément au modèle précédent)

Exemple du transfert vidéo entre une caméra et un écran

Objets :

Caméra avec l'interface vidéoOut, et écran avec l'interface vidéoIn

L'expression de la contrainte "latence bornée" devient :

pour tout n :

$$|\text{date}(\text{écran.vidéoIn.RS},n) - \text{date}(\text{caméra.vidéoOut.ES},n)| \leq 10\text{ms}$$

La spécification de paramètres de QoS est construite en indiquant, des clauses requises (Req) et des clauses fournies(Prov) , avec la relation Req(A) -> Prov(A).

¹ D pour Donnée et S pour Signal, dans le cas des signaux, il n'y a pas de réponse (R)

Exemple (2)

Objet Liaison transfertVideo :

```
// interface avec la caméra
interface flux transfertVideoIn {
    fluxEntrant videoIn (video) ;
}
//interface avec l'écran de restitution
interface flux transfertVideoOut {
    fluxSortant videoOut(video);
}
//interface contrôle de QoS
interface signal qosControle {
    signalOut videoEmis (date);
    signalIn videoDélivré (date) ;
}
```

Clause Fournie

//transfert soutient 25 images/s avec une latence entre 40 et 60 ms

pour tout n, $\text{date}(\text{transfertVideoOut.videoOut.ES}, n+25)$

$\leq \text{date}(\text{transfertVideoOut.videoOut.ES}, n) + 1000 \text{ ms}$

et pour tout n,

$40\text{ms} \leq$

$|\text{date}(\text{transfertVideoIn.videoIn.RS}, n) - \text{date}(\text{transfertVideoOut.videoOut.ES}, n)|$

$\leq 60\text{ms}$

et pour tout n, $\text{date}(\text{qosControle.videoEmis.ES}, n)$

$= \text{date}(\text{transfertVideoIn.videoIn.RS}, n)$

et pour tout n, $\text{date}(\text{qosControle.videoDélivré.RS}, n)$

$= \text{date}(\text{transfertVideoOut.videoOut.ES}, n)$

Clause Requisite

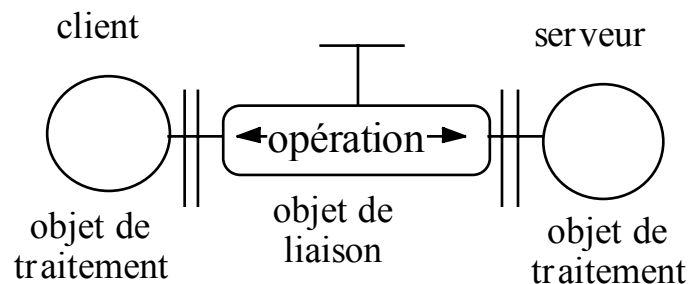
// si la caméra envoie 25 images/s

pour tout n, $\text{date}(\text{transfertVideoIn.videoIn.RS}, n+25)$

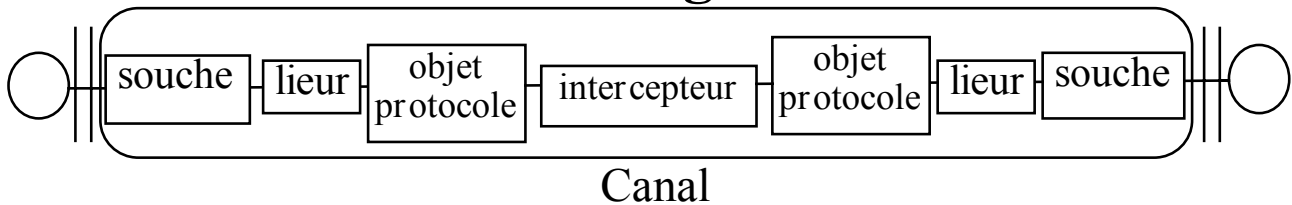
$\leq \text{date}(\text{transfertVideoIn.videoIn.RS}, n) + 1000 \text{ ms}$

Point de Vue de l'Ingénierie

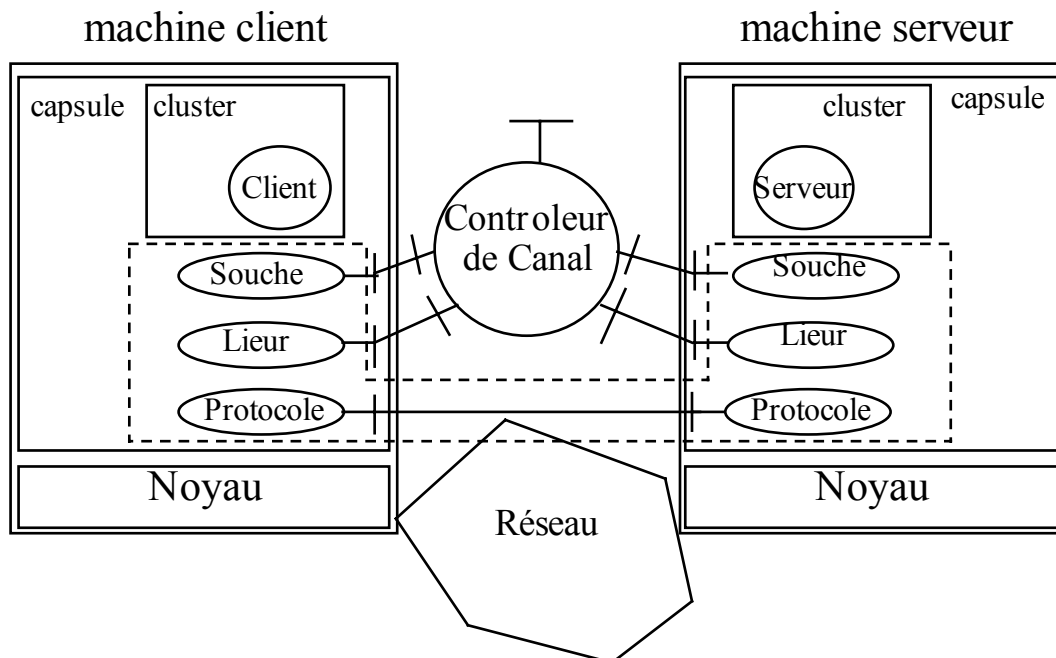
Modèle des traitements :



Modèle d'ingénierie :



intercepteur fonction équivalente à pont CORBA/passerelle
(conversion de protocole)



Les équations de QoS peuvent se projeter sur chacun des composants d'ingénierie, en particulier le réseau.

Autre Modèle de Spécification de QoS pour les applications : le modèle de l'API Winsock2

```
typedef struct _QualityOfService
{
    FLOWSPEC    SendingFlowspec; /*flow spec for data sending*/
    FLOWSPEC    ReceivingFlowspec; /*flow spec for data recvg*/
    WSABUF      ProviderSpecific; /*provider specific stuff*/
} QOS;
```

pour G711 (80 octets de voix numérisée suivant la μ -law toutes les 10ms, donne 120 octets dans un datagramme IP+UDP+RTP)

```
typedef struct _flowspec
{
    int32      TokenRate;          /* r, 12000 bytes/sec */
    int32      TokenBucketSize;   /* b, 120 bytes      */
    int32      PeakBandwidth;     /* p, 12000 bytes/sec */
    SERVICETYPE ServiceType;     /* GARANTEED        */
    int32      MaxSduSize;        /* M, 120 bytes      */
    int32      MinimumPolicedSize; /* m, 120 bytes      */
} FLOWSPEC;
```

Ca serait le même modèle si on utilise cette API avec un réseau ATM.

Rappel du Modèle de contrat ATM :

Paramètres de Trafic :

- *Peak Cell Rate (PCR)*, cells/s (*débit max*)
- *Substainable Cell Rate (SCR)*, cells/s \leq PCR (*débit moyen*)
- *Maximum Burst Size (MBS)*, cells, (*nombre max de cellules envoyées au débit PCR*)
- *Minimum Cell Rate (MCR)*, cells/s, pour service ABR(*débit minimal garanti*).

Paramètres de QoS demandée :

- *maximum Cell Transfer Delay (maxCTD)*, sec (*délai max*)
- *peak-to-peak Cell Delay Variation (peak-to-peak CDV)*, sec (*gigue max*)
- *Cell Loss Ratio (CLR)*, cells (*taux de perte de cellules max*)

services ATM

attributs	CBR	rt-VBR	nrt-VBR	UBR	ABR
paramètres de Trafic:					
PCR	√	√	√		√
SCR, MBS	n/a	√	√	n/a	n/a
MCR	n/a	n/a	n/a	n/a	√
paramètres de QoS:					
ppCDV	√	√			
maxCTD	√	√	√		
CLR	√	√	√	√	

√ = spécifié , n/a = non applicable

ne pas oublier le nouveau GFR !!! similaire à un service de type Frame Relay dans un contexte ATM.

Autre Modèle de Spécification de QoS pour les la partie Réseau : approche TENET

Pile de protocoles :

TCP	UDP	RMTP	CMTP	R	R
IP		RTIP		C	T
Liaison de Données				A	C
				P	M
					P

La couche Liaison de Données est supposée capable d'offrir un service d'acheminement à caractère déterministe : FDDI, ATM.

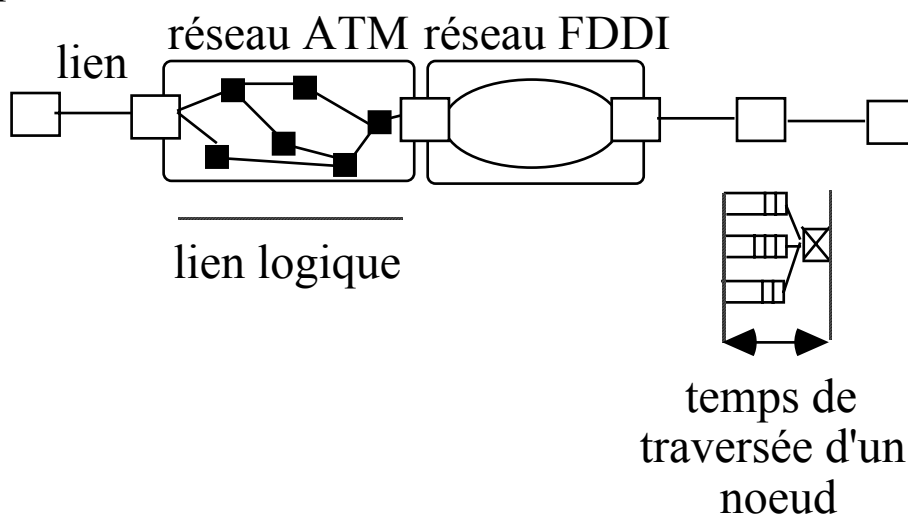
- **RTIP** : Extension de IP pour gérer la notion d'urgence des Datagrammes, et la surveillance du trafic réel ... en cas de pb, il réduit le trafic, il est orienté connexion unidirectionnelle
- **RMTP** : Real time Message Transport Protocol, orienté message
- **CMTP** : Continuous Media Transport Protocol, orienté flot d'octet
- **RCAP** : Real time Chanel Administration Protocol, Réserve de ressources (ancêtre de RSVP)
- **RTCMP** : Real Time Control Management Protocol, Gestion de Ressources

TENET - Hypothèses du modèle

Le réseau est composé d'un ensemble de sous-réseaux interconnectés par des noeuds de routage.

Les sites traversés ont des horloges synchronisées, ou re-synchronisées périodiquement.

Les liens de communication entre noeuds sont déterministes. Un lien logique est : un lien physique ou un sous-réseau entier. Le lien logique est donc lui aussi déterministe.



Les files d'attente dans les noeuds intermédiaires gèrent les messages en fonction de leur priorité. Une politique Earliest Deadline First peut être adoptée pour gérer les messages en fonction de leur priorité.

Les liens logiques ont un taux d'erreur négligeable.

TENET – Modèle de Gestion de ressources - Types de Canaux

Un canal entre deux communicants est matérialisé par un chemin fixe entre ces deux entités. C'est une sorte de circuit virtuel.

Le modèle TENET suppose deux types de Canaux Temps Réel :

- Un canal TR est dit déterministe si le délai d'acheminement de bout en bout est garanti dans tous les cas.
- Un canal TR est dit statistique si le délai d'acheminement de bout en bout est assuré avec une probabilité supérieure à une valeur seuil

TENET - Spécification QoS

Paramètres de QoS spécifiés par l'utilisateur :

- Délai d'Acheminement
- Gigue
- Débit
- Fiabilité (taux d'erreur)

TENET - Délai d'acheminement

Délai d'acheminement (émission->délivrance au récepteur), la fiabilité de l'information n'est pas garantie ... une donnée peut arriver à l'heure mais être erronée, attention !

Soit $D_{c,m}$ le délai associé à un message m sur un canal c

Délai Déterministe :

$D_{c,Max}$ délai max, $D_{c,Max}$ est spécifié par l'utilisateur

pour tout m : $D_{c,m} \leq D_{c,Max}$

Délai Statistique :

$Z_{c,min}$ seuil de respect du délai max $D_{c,Max}$ spécifié par l'utilisateur

pour tout m : $\text{Prob}(D_{c,m} \leq D_{c,Max}) \geq Z_{c,min}$

Certaines applications de transmission d'image, de son ou d'animation tolèrent des erreurs mais nécessitent le respect des délais d'acheminement pour les messages corrects.

TENET - Gigue

Gigue, écart toléré par rapport à un temps de référence

Soit D_c un délai idéal pour l'acheminement de données

Gigue Déterministe :

$J_{c,Max}$ écart max p/r à D_c , spécifié par l'utilisateur

$$\text{pour tout } m : J_{c,m} = |D_{c,m} - D_c| \leq J_{c,Max}$$

Gigue Statistique :

$U_{c,min}$ seuil de respect de $J_{c,Max}$, spécifié par l'utilisateur

$$\text{pour tout } m : \text{Prob}(J_{c,m} \leq J_{c,Max}) \geq U_{c,min}$$

TENET - Débit

Le Débit est contraint par la charge du service de communication, et le taux d'erreur de transmission²

soit TH_c le débit courant du canal de communication c

Débit Déterministe :

soit $TH_{c,min}$ le débit min spécifié par l'utilisateur

pour tout c : $TH_c \geq TH_{c,min}$

Débit Probabiliste :

$V_{c,min}$ seuil de respect de TH_{min} spécifié par l'utilisateur

pour tout c : $\text{Prob}(TH_c \geq TH_{c,min}) \geq V_{c,min}$

² Les erreurs de transmission impliquent des retransmissions qui pénalisent le réseau

TENET - Fiabilité

Fiabilité, elle concerne le taux d'erreur admis, certaines applications ont des contraintes d'échéance et de fiabilité absolue : embarqué militaire par ex.

$W_{c,min}$ seuil de succès min souhaité par l'utilisateur

$$\text{Prob}(\text{message délivré correct}) \geq W_{c,min}$$

Dans la fiabilité on ne comptabilise pas les messages qui sont hors délai et hors gigue (pourtant éliminés), ils sont comptés ailleurs.

TENET - paramètres de calcul pour la réservation de ressources

Sur un intervalle d'observation I :

- $X_{\min,c}$ délai d'inter-arrivée min des messages sur le canal c (taux d'arrivée $1/X_{\min,c}$ correspond au Lambda des files d'attentes)
- $X_{ave,c}$ délai moyen d'inter-arrivée des messages sur le canal c
- $t_{c,n}$ temps maximum pour servir un message au noeud n sur le canal c (taux $1/t_{c,n}$ correspond au Mu des files d'attentes)
- $d_{c,n}$ délai limite pour traverser le noeud n sur le canal c (correspond à une échéance globale)
- $d_{c,n}^m$ délai de traversée du noeud n sur le canal c par le message m (correspond à une échéance message)
- $S_{\max,c}$ taille max d'un message sur le canal c

TENET - Tests des Ressources (1)

Un canal ne peut exister que s'il y a assez de ressources, ceci est fait à l'aide de tests au moment de l'établissement du canal, à la traversée de chaque noeud.

Tests Intermédiaires (à l'établissement ou en mode établi) :

Puissance CPU pour un Canal Déterministe :

pour un canal c , le taux d'utilisation de la CPU d'un noeud est $t_{c,n}/X_{\min,c}$, (règle classique du Lambda/Mu en files d'attentes), ce qui doit être vrai pour chaque noeud d'où la contrainte à vérifier

$$\sum_n (t_{c,n}/X_{\min,c}) < 1$$

autres Tests :

Puissance CPU pour un Canal Statistique

Test de Délai Limite

Test d'allocation de tampons pour un Canal Déterministe

Test d'allocation de tampons pour un Canal Statistique

Test de Contrôle de Gigue

TENET - Tests des Ressources (2)

Tests Destinataires :

Test D pour un canal quelconque :

$$D_{c,Max} \geq \sum_{n=1}^N d_{c,n}^m$$

pour m = demande d'ouverture lors de la réservation de ressources à la création d'un canal

On calcule le $d_{c,n}$ de chaque nœud par :

$$d_{c,n} = 1/N * \left[D_{c,Max} - \sum_{i=1}^N d_{c,i}^m \right] + d_{c,n}^m$$

où N = tous les nœuds traversés sauf le nœud courant de numéro n (ici le destinataire pour ce test)

Test Z pour un canal statistique : Calcul de la probabilité de dépassement du délai d'acheminement $D_{c,Max}$ ou D_c

TENET - RCAP

Service de Réserveation de ressources ou de Contrôle d'Admission.

Chaque site dispose d'un daemon RCAP, le demandeur fournit la suite des noeuds intermédiaires que doit traverser le canal temps réel à établir.

Cette réserveation de ressources peut emprunter des connexions TCP entre daemon RCAP.

La réserveation de ressources faite par RCAP sert pour RTIP

Format d'un message de réserveation de ressources :



HR : Header Record

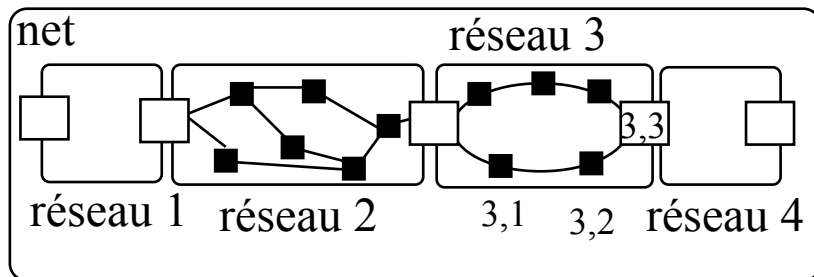
NSR : Network Sub-header Record

ER : Establishment Record

NSR -> début d'une description de ressources pour un sous-réseau

ER -> paramètres de chaque nœud du sous réseau qui représentent les ressources locales affectées au canal, les ressources sont réservées si elles sont disponibles au passage du message

TENET - Exemple de réservation RCAP



Message "establish_request" :

réseau 3, nœud 3,2 (ajoute $ER_{3,2}$):

HR	NSR	ER	ER	NSR	ER	ER
	net	réseau 1	réseau 2	réseau 3	3,1	3,2

réseau 3, nœud 3,3 (ajoute $ER_{3,3}$):

HR	NSR	ER	ER	NSR	ER	ER	ER
	net	réseau 1	réseau 2	réseau 3	3,1	3,2	3,3

$ER_{réseau 1}$ et $ER_{réseau 2}$ récapitulent les ressources déjà utilisées pour le canal temps réel dans les réseaux 1 et 2.

réseau 3, nœud 3,3 - fin du réseau 3 (synthèse) :

HR	NSR	ER	ER	ER
	net	réseau 1	réseau 2	réseau 3

réseau 3, nœud 3,3 - début du réseau 4 :

HR	NSR	ER	ER	ER	NSR
	net	réseau 1	réseau 2	réseau 3	réseau 4

TENET - PDU RCAP

établissement par **establish_request** : **réponse positive** par **establish_accept** les paramètres liés aux tests D et Z sont connus

réponse négative par **establish_denied** qui provoque la libération des ressources réservées

fermeture par **close_request** (initiateur), propagation par **close_request_forward** (source) ou **close_request_reverse** (destinataire)

interrogation sur l'état des ressources de tous les noeuds d'un canal temps réel par **status_request** et réponse par **status_report**

TENET – CMTP et RMTP à titre indicatif

CMTP : Transfert de Données temps réel isochrone

Trafic de type continu : vidéoconférence, son haute fidélité... ex :
une caméra transmet vers une carte vidéo sur un PC

Deux types de transferts :

- flot de messages : l'utilisateur produit un message de longueur variable pdt un intervalle de longueur fixe (qui donne la période)
- flot d'octets : l'utilisateur spécifie la quantité d'octets émise par période sous la forme d'un min et d'un max

Les données peuvent contenir des informations de synchronisation.

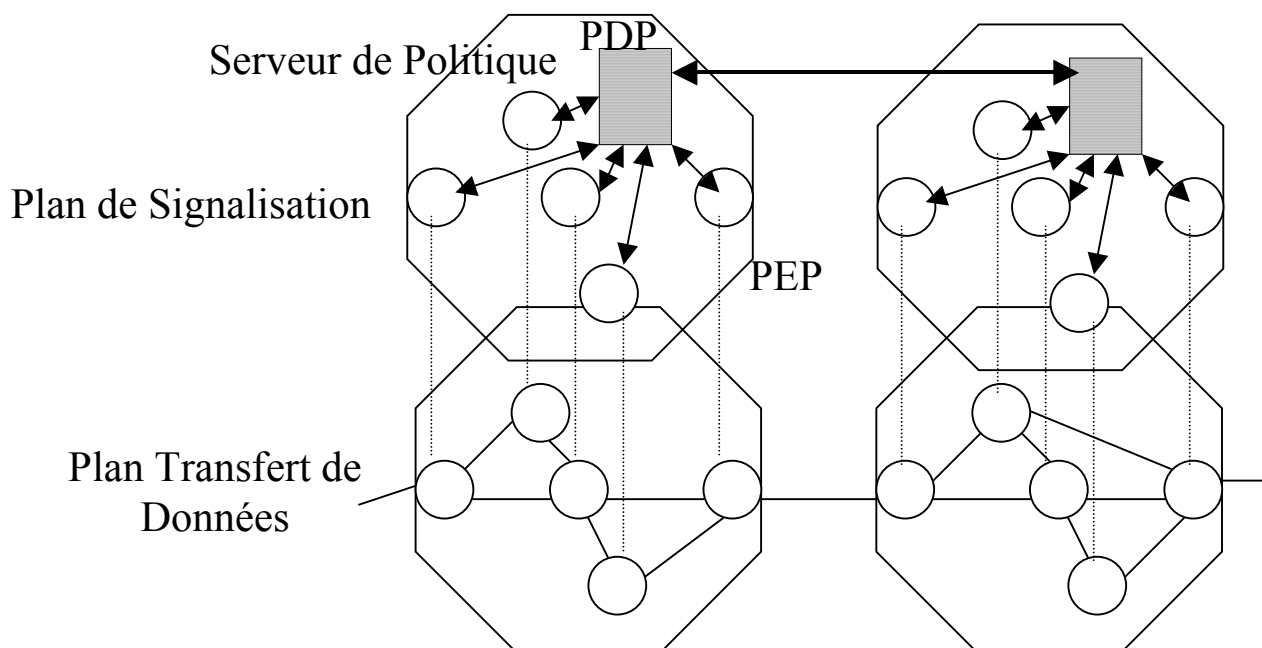
RMTP : Transfert de Données temps réel asynchrone (Trafic de type sporadique)

SLA/SLS pour Internet à QoS

SLA/SLS

On a vu que les routeurs étaient les éléments essentiels d'une architecture à QoS aussi bien pour l'approche IntServ que pour l'approche DiffServ.

Pb à résoudre : paramétrer les routeurs au fur et à mesure du fonctionnement du réseau, problème similaire à la supervision.



Travaux de l'IETF : WG RAP

RAP = Resource Allocation Protocol

Definition des entités du plan de signalisation :

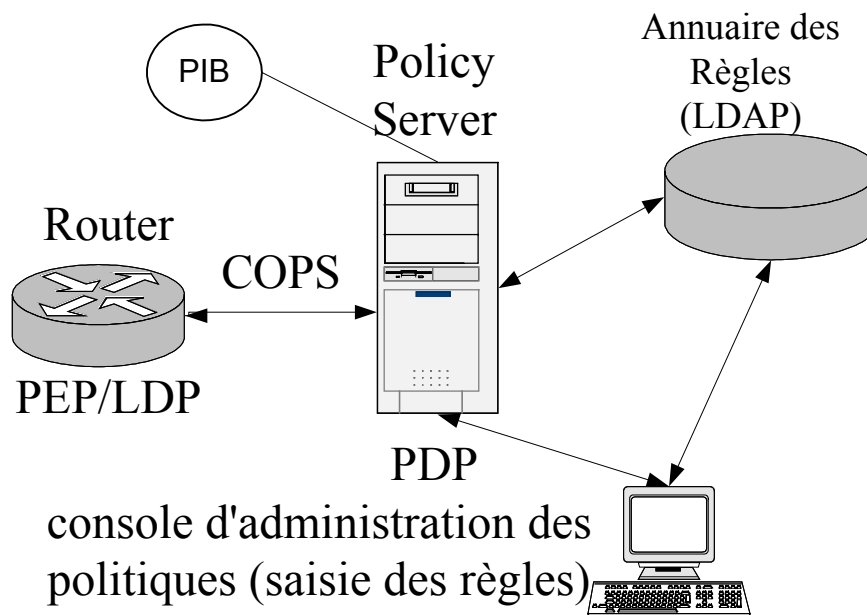
- PDP (Policy Decision Point) : encore appelé Policy Server, Point Central, en particulier :
 - Il est capable de localiser les règles applicables aux PEP dont il a la responsabilité.
 - Il transforme les règles du format annuaire vers un format compréhensible par les routeurs.
 - Il vérifie les conditions d'application des règles.
- PEP (Policy Enforcement Point) : Routeur
Applique les décisions de stratégie :
Classification, Ordonnancement, Marquage ...
- LDP (Local PDP) : Routeur de bordure, contient aussi un PEP
 - Décider du marquage des datagrammes (DSCP)
 - Effectuer le contrôle d'admission : tests de conformité du Trafic, filtrage
 - Lissage du Trafic

Protocole de Signalisation entre PEP et PDP :

- COPS (Common Open Policy Service) RFC2748
 - COPS-RSVP, COPS avec RSVP (RFC2749)
 - COPS-PR, pour COPS Policy Provisioning (RFC3084)

Architecture distribuée à l'image de l'architecture de supervision (COPS est équivalent de SNMP)

Entités de gestion de la QoS



PIB = Policy Information Base, équivalent de la MIB pour la gestion de QoS, la convergence MIB/PIB est à l'étude

La console d'administration sert à saisir les règles de gestion des flots pour les différents routeurs. Elle contient des outils de configuration, visualisation, parfois de dimensionnement, et de suivi... Définition de la stratégie à suivre pour assurer le niveau de QoS requis par les utilisateurs. En général il existe un outil qui intègre une validation syntaxique et sémantique des règles, et vérifie leur cohérence entre elles.

L'annuaire contient les règles telles qu'elles ont été validées, elles sont accédées par les PDP. LDAP est la solution la plus répandue, mais on peut s'appuyer sur une architecture WEB, CORBA ou propriétaire.

COPS

Protocole Requête-Réponse pour l'échange d'information PEP-PDP :

- Règles de marquage
- Stratégies à adopter
- Configuration
- Règle de sécurité
- ...

Utilise une connexion TCP

Possibilité de notification du PEP vers PDP

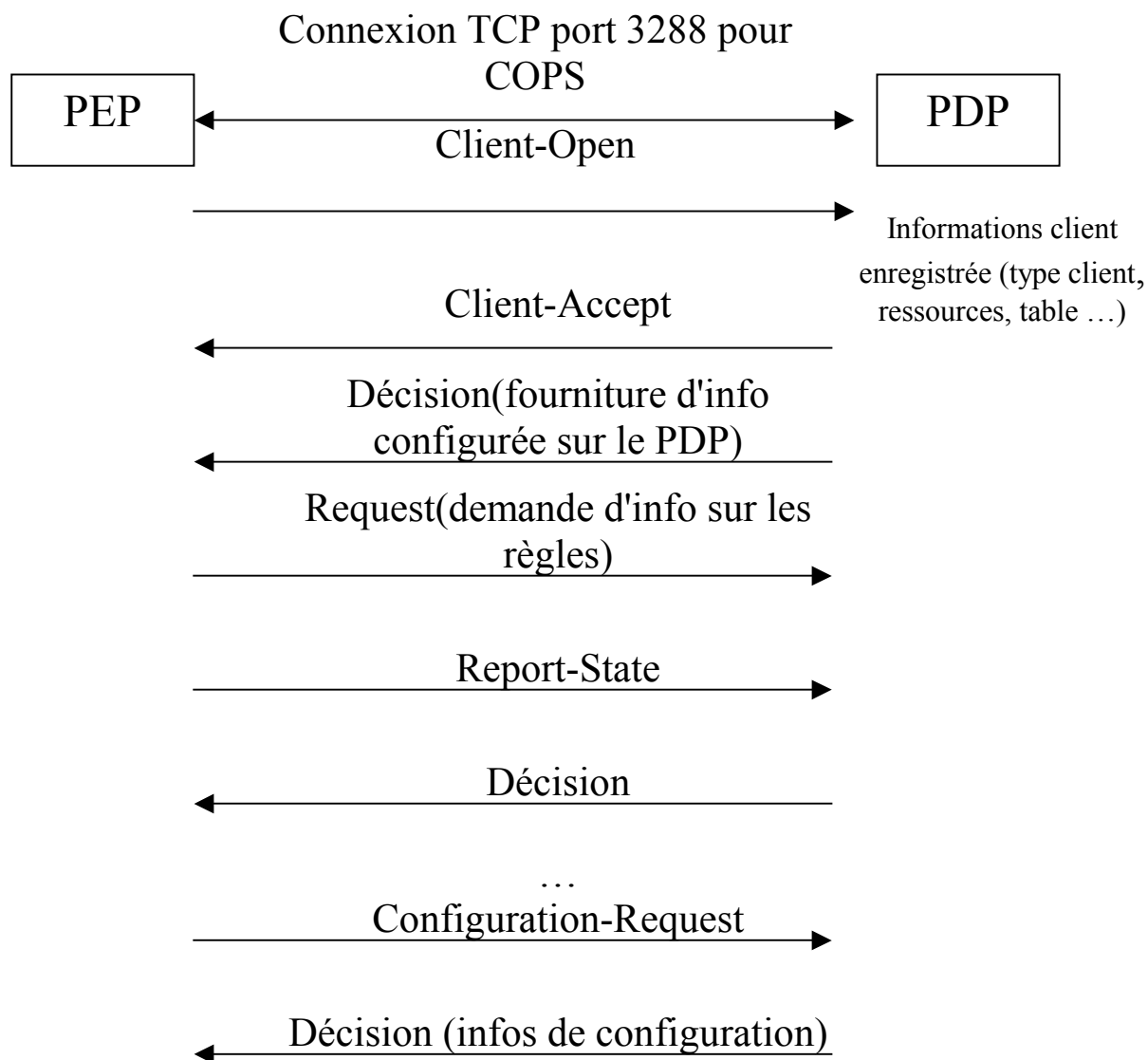
Certain conflit d'intérêt entre COPS et SNMP.

COPS-RSVP : les clients (PEP sont des routeurs qui supportent RSVP, en particulier, COPS peut transporter des messages PATH, RESV, PATHERR, RESVERR

COPS-PR : définit de nouveaux objets dans COPS, un PEP peut dépendre de plusieurs PDP, un part type de politique (performance, sécurité, tolérance aux pannes ...)

COPS : Exemple de fonctionnement

D'après G. Pujolle "Tutorial : Un état de l'art sur la nouvelle génération Internet". DNAC' 2000.



Spécification de règles de QoS

Spécification en langue naturelle

=> Spécification formelle

=> Génération de règles :

"if (dest(datagramme)=A.B.C.0/n) and
(port_source=x..z) alors appliquer traitement
Classe 1"

=> Formalisation sous forme de tables:

Nom Politiq	IPSrc	Port Src	prot ³	IPDst	Port Dst	Cl/ Pr	PHB	Ovrflow PHB
P1	12.0.0.3	256	6	*	*	1	EF	BE
P2	*	*	6	12.0.0.3	256	1	EF	BE
P3	9.2.0.0/24	*	6	*	80	2	AF	BE
P4	*	*	6	9.2.0.0/24	*	2	AF	BE
P5	*	4000- 6000	17	*	*	3	BE	Drop
P6	*	*	17	*	4000- 6000	3	BE	Drop

Remarque : La recherche dans la table doit être efficace, il faut organiser le parcours des politiques suivant un arbre, la politique la plus fréquemment appliquée à la racine de l'arbre.

Remarque : Le champ DSCP peut lui aussi être prise en compte dans la table. Les champs ci-dessus ne sont pas obligatoires.

³ 6 = TCP, 17 = UDP

PIB

C'est le pendant de la MIB SNMP dans le domaine de la gestion de QoS.

Les données sont représentées sous la forme d'un arbre suivant des conventions identiques à la MIB SNMP :

```

PIB-----+-----+-----+-----PRC--+-PRI
          |         |         |         +--PRI
          |         |         |         +---PRC---PRI
          |         |         |         +---PRC--+-PRI
          |         |         |         +--PRI
          |         |         |         +--PRI
          |         |         |         +--PRI
          |         |         |         +--PRI
          +---PRC---PRI

```

PRC = Provisioning Class

PRI = Provisioning Instances

Il y a une certaine concurrence entre MIB et PIB, et d'ailleurs se pose le pb de leur intégration. Toutefois, la structure PIB est plus adaptée aux besoins de gestion de la QoS

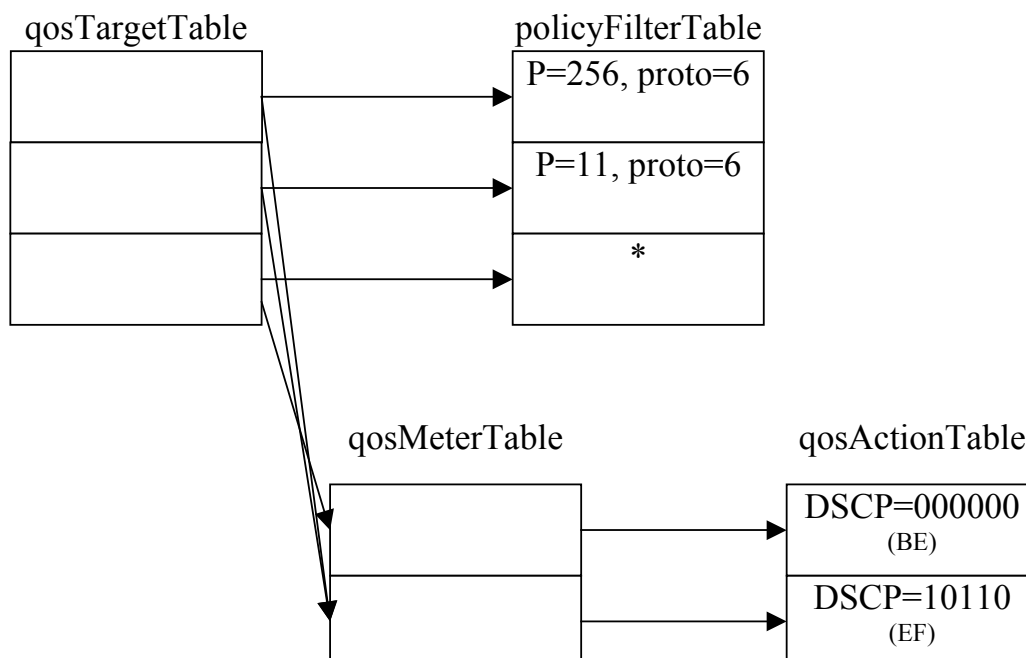
Remarque: Il existe une PIB-RSVP

PIB pour DiffServ

La PIB associée à DiffServ contient 4 tables :

- **policyFilterTable** : 5-tuple d'identification de flot (@IP source, @IP destination, port source, port destination, protocole), on peut avoir en plus le champ DSCP
- **qosActionTable** : les actions qui peuvent être faite par le routeur par entrée
- **qosMeterTable** : seuils sur la bande passante max, 3 types : sans seuil, un débit constant, un débit moyen avec un débit max, et à chaque entrée est associé une action
- **qosTargetTable** : table qui regroupe les descripteurs précédents

Vue globale :



Points Délicats d'une architecture à QoS

La spécification de contrat de QoS et le SLA qui en découle correspondent au premier pas de la vie d'un réseau devant être conforme à une architecture à QoS.

Ce qui devient clef, c'est le suivi minute après minute de la QoS offerte par le réseau :

- Outils de mesure,
- Aide à l'analyse,
- Supervision et maintenance de la QoS

Il existe un besoin d'outils déterminant.