Niveau 2/OSI: Services LIAISON de DONNEES

1. Le service Liaison de données

1.1. Rôle

La couche Liaison de données assure, dans la mesure du possible, un transfert de données birectionnel fiable entre deux systèmes, c'est à dire :

- avec un taux d'erreur résiduel négligeable
- sans perte
- sans duplication

Ce service est fourni en tenat compte des temps de réponse (temps utile) souhaités et en tenat compte des contraintes économiques.

Les systèmes interconnectés par ce service sont soit des ETTD reliés par un même support physique, soit un ETTD et un point d'accès à un réseau étendu (ou métroplolitain).

L'impératif de fiabilité doit être mis en balance avec l'impératif de performance, la transmission devant se faire en temps utile. Il est donc parfois souhaitable de reporter le contrôle d'erreurs et le contrôle de flux à un niveau supérieur (souvent au niveau Transport).

Pour assurer cette fiabilité, le service Liaison de données détecte et corrige, si possible, les erreurs pouvant se produire dans la couche Physique. Si le taux d'erreurs offert par la couche Physique est plus faible que le taux requis (cas d'un réseau à fibre optique par exemple ou d'applications peu contraignantes) aucun traitement n'est nécessaire; par contre le contrôle de flux permet d'éviter la perte de blocs. Si le traitement des erreurs au niveau Liaison est insuffisant (ou nul ..) un traitement supplémentaire est réalisé au niveau Réseaux (rarement) ou Transport.

Si nécessaire, le service Liaison de données fournit aussi les moyens fonctionnels et procéduraux pour établir, maintenir et libérer les connexions de Liaison de données entre des entités de Réseau.

Enfin lorsque le service multipoint entraine le partage du support de communication physique entre plusieurs systèmes, le service Liaison de données régle les accès à cette ressource critique (contrôle centralisé ou distribué pour un réseau local). Pour ce faire il peut éventuellement être découpé en deux sous-couches :

- MAC : commande d'accès au médium (Method Acces Control)
- LLC : commande de liaison logique (Logical Link Control)

1.2. Service Physique requis

Une connexion de Liaison de données est réalisée à l'aide d'une ou de plusieurs connexions physiques qui lui fournissent un "Circuit de données".

Ce circuit de données assure le transfert d'un flux d'information binaire avec une certaine qualité de service (débit, mode de transmission, taux d'erreurs, commutation à l'appel, etc) et notifie les dérangements.

Très souvent le mode de transmission est syncrone (ou isochrone).

Le circuit de données peut être réalisé par une liaison spécialisée ou par une liaison physique d'un réseau de télécommunications en commutation de circuits (réseau téléphonique analogique ou numérique) ou en commutation de cellules (ATM).

1.3. Service fourni

- Connexion - Libération
- Transfert de (blocs de) données : DLPDU = trames
- Contrôle de flux
- Identification de la liaison de données
- Maintien en séquence des trames
- Notification des erreurs (anomalies) si elles ne sont pas récupérables.

2. PROTECTION CONTRE LES ERREURS

Pour des renseignements complémentaires, voir le cours de "Théorie de l'information" et le polycopié de "Théorie de l'Information" de J.F. Petit.

Le service "Liaison de données" de niveau 2/OSI a comme fonction principale, d'assurer dans la mesure du possible (si les performances le permettent ...) la fiabilité de l'information transférée entre deux systèmes OSI directement connectés.

Pour cela il met en oeuvre des services de contrôle d'erreurs et de contrôle de flux. Les autres activités servent à initialiser ces services (connexion), à rendre compte de leur insuffisance (signalisation d'anomalies) ou à les réaliser (séquencement par exemple).

Le contrôle d'erreurs peut aussi mettre en oeuvre des fonctions de niveau Physique (**1/OSI**) si on utilise la **détection de qualité de signal** dans les ETCD.

Au niveau **2/OSI**, comme au niveau **4/OSI** (Transport), le contrôle d'erreurs utilise une **redondance** de l'information transmise. Celle-ci est introduite par un système de **codage détecteur ou correcteur d'erreurs**.

2.1. Caractérisation des erreurs

Sur un système de transmission téléinformatique, le codage de l'information est binaire : l'information à transmettre est généralement groupée dans des PDU qui peuvent avoir une longueur de quelques centaines ou milliers de bits.

Au niveau Physique, le signal support de cette information est perturbé par un bruit additif; des erreurs de transmission peuvent se produire qui transforment les "1" en "0" ou les "0" en "1" ... . Ces erreurs peuvent être isolées (un bit erroné dans une longue séquence de bits corrects) ou groupés en paquets (un nombre important de bits erronés regroupés dans une courte séquence).

exemple :

chaîne à transmettre : 1 **0** 1 1 0 0 0 1 1 1 0 1 0 1 0 1 1 **1** 0 **0 0** 0 1 **0** 1 0 1 0
chaîne reçue : 1 **1** 1 1 0 0 0 1 1 1 0 1 0 1 0 1 1 **0** 0 **1 1** 0 1 **1** 1 0 1 0
syndrome d'erreurs : 0 **1** 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 **1 1** 0 0 **1** 0 0 0 0
erreur isolée ----------**
paquet d'erreurs de taille 7 ----------**** ** **

Des statistiques faites sur les séquences les plus courtes (Télex, 5 bits) montrent qu'environ 50 % seulement des erreurs sont isolées et que l'on peut avoir 4 bits erronés consécutifs dans 2 à 6% des cas. D'autres études faites sur un réseau téléphonique, sur des séquences d'une centaine d'octets, montrent des erreurs isolées dans 36 % des cas seulement, des paquets de tailles ≤ 10 dans 15% des cas et que des paquets d'erreurs très longs peuvent être observés dans 10% des cas. Ceci prouve qu'il est quasi inutile de ne détecter que les erreurs simples (un seul bit erroné, par exemple par un contrôle de parité, voir ci-dessous) mais que le système de protection doit prendre en compte les paquets d'erreurs pour donner une sécurité suffisante. Ceci nous conduit à utiliser des **codes cycliques**.

2.2. Puissance d'un code . Distance de Hamming.

La puissance de détection ou de correction d'un code correcteur d'erreurs est donné par sa distance de Hamming. Sans rentrer dans des considérations mathématiques dépassant le cadre de cet exposé, cette distance de Hamming caractérise la faculté de séparer aisément les symboles constituant le code lorsqu'ils sont plus ou moins erronés.

Prenons un exemple très imagé et très approximatif.

Un groupe d'enfants dans le cadre d'un jeu de piste décide de coder les distances par un code "secret" dans lequel les valeurs 1, 2 et 3 mètres sont codées par les symboles C, E et F. Les valeurs 1, 2 et 3 représentent l'information à transmettre. Les trois symboles C,E,F, constituent notre code. Par beau temps, la lecture du symbole E donne aisément la valeur 2. A la suite d'une légère averse une branche du E s'efface et on lit C (ou F) : il y a erreur de transmission due au "bruit" créé par l'averse qui a modifié le symbole E. Le choix des symboles étant peu judicieux, cette erreur est indécelable et les enfants cherchent vainement le "trésor" à 1 (ou 3) mètres.

Lors d'une autre journée les symboles retenus sont A, E et J. Le même incident (C ou F reçu) donne un symbole non reconnu, hors code. Une erreur de transmission est détectée et on peut demander la répétition du message. Il est même (presque) possible de corriger directement cette erreur : un symbole F reçu ne peut très probablement être issu que du symbole E erroné (beaucoup plus probable qu'une déformation de A ou J). Ce nouveau code a une distance de Hamming beaucoup plus élevée que le précédent, ses symboles sont beaucoup plus différents les uns des autres.

Pour avoir une vue plus précise nous pouvons considérer, par exemple, plusieurs codes bâtis avec un alphabet à 4 bits, selon le tableau ci-dessous.

La distance de Hamming du code 1 vaut 1. Chaque symbole ne diffère de ses plus proches voisins que de 1 bit.

Dans le code 2, les symboles différent par au moins 2 bits : la distance de Hamming est 2. Le nombre de bits à 1 dans chaque symbole est pair (code de parité paire). Ce code peut détecter toutes les erreurs simples (un bit erroné) ou toutes les erreurs d'ordre impair (nombre impair de bits erronés). Les huit valeurs exclues (par rapport au code 1) constituent un autre code de distance de Hamming égale à 2 (code de parité impaire) aux performances identiques.

code 1	code 2	code 3
0 0 0 0	0 0 0 0	0 0 0 0
0 0 0 1	0 0 1 1	1 0 1 1
0 0 1 0	0 1 0 1	
0 0 1 1	0 1 1 0	
0 1 0 0	1 0 0 1	
"	1 0 1 0	
"	1 1 0 0	
1 1 1 1	1 1 1 1	

Le code 3 a une distance de Hamming de 3. Il permet de détecter à coup sûr deux bits erronés ou de corriger une erreur simple, c'est à dire de retrouver directement le symbole émis si le symbole reçu n'a qu'un bit erroné : si on reçoit 0 0 1 0, le symbole émis est très probablement 0 0 0 0.

D'une manière plus générale, si on veut respectivement

- détecter p erreurs (p bits erronés dans une séquence)
- corriger q erreurs simples
- corriger q erreurs et en détecter q

le code de Hamming à utiliser doit avoir une distance d telle que :

$$p + 1 \leq d$$
$$2*q + 1 \leq d$$
$$2*q + p + 1 \leq d$$

Si de plus un code est cyclique (voir ci-dessous) et a un champ de redondance de longueur r, il peut détecter les paquets d'erreurs de taille ≤ r (ou corriger les paquets de taille ≤ r/2).

2.3. Codes cycliques

2.3.1. Représentation polynomiale de l'information

Pour traiter ces problèmes de codage correcteur d'erreurs, il est possible d'utiliser une notation matricielle. Toutefois une notation polynomiale, même si elle ne montre plus le codage binaire des valeurs, fournit des outils commodes et puissants pour traiter les codes cycliques.

Dans cette notation, le **codage binaire de l'information** donne les **coefficients de polynômes en x** (coefficient à valeur dans le Corps de Galois d'ordre 2); Ces polynômes présentent une structure d'anneau. Certains d'entre eux ne sont pas divisibles (analogie avec les nombres premiers sur le corps des entiers); ils sont dits **irréductibles**.

Ainsi le mot binaire 1 1 0 1 0 1 1
donne le polynôme $1.x^6+1.x^5+0.x^4+1.x^3+0.x^2+1.x^1+1.x^0$
soit $x^6+x^5+x^3+x+1$.

Dans le corps CG2 où sont pris ces coefficients l'opération notée additivement est le "ou exclusif" (0+0=1+1=0 ; 1+0=0+1=1).

Un **code cyclique** est un code dont tous les symboles sont **multiples d'un polynôme générateur G(x)** irréductible ou produit de polynômes irréductibles, à une constante près (en particulier 0 ...).

Les polynômes irréductibles sont donnés dans les ouvrages de W. Peterson (Error Correcting Codes) ou W.W. Peterson et E.J. Weldon (Error Correcting Codes, MIT Press) par les "**tables de Peterson**" (voir polycopié de Théorie de l'information).

Pour les plus bas degrés x+1, x^2+x+1,x3+x+1 et x^3+x^2+1 sont irréductibles.

Le polynôme normalisé CRC16 : $\mathbf{x^{16}+x^{15}+x^2+1}$ est le produit $(x+1)*(x^{15}+x+1)$
Le polynôme normalisé CCITT : $\mathbf{x^{16}+x^{12}+x^5+1}$ est le produit
$(x+1)*(x^{15}+x14+x13+x12+x4+x3+x2+x+1)$

Nota : Le polynôme x+1 génère un code de parité paire. Son emploi garantit de détecter toutes les erreurs simples et les erreurs d'ordre impair.

2.3.2. Codage par multiplication (principe)

Ce système n'est pas utilisé pour le contrôle d'erreurs (mais par exemple dans les brouilleurs des modems), mais il est le plus simple à exposer.

Soit un message M(x) à transmettre d'une longueur éventuellement élevée, dont le maximum est fonction du polynôme générateur utilisé.

Le message transmis est C(x) = M(x)*G(x)

Si la transmission est erronée un syndrome d'erreur E(x) s'ajoute a C(x) (bloc de même longueur avec des 1 aux endroits où s'est produite une erreur : voir exemple ci-dessus).

Le message reçu est $C^*(x) = C(x) + E(x)$.

A la réception,

C*(x) est divisé par G(x).

Si le reste R(x) de la division n'est pas nul, $C^*(x)$ n'est pas un mot du code et une erreur de transmission s'est produite.

Sinon le quotient rend le message M(x).

Si le code a une distance de Hamming suffisante, R(x) permet de retrouver E(x) (généralement par une table); alors C(x) = C*(x) + E(R(x)). Il suffit alors de refaire une division pour avoir une correction directe d'erreur.

Ce codage n'est pas utilisé car il nécessite deux outils ou algorithmes différents (multiplication et division polynomiales). D'autre part, il "brouille" les bits transmis comme le montre l'exemple ci-dessous. On ne peut retrouver l'information sans décodage (en prenant un risque d'erreur ..)

Exemple :

message : **1 0 1 1 0** --> $M(x) = x^4+x^2+x$
polynôme générateur $G(x) = x^3+x+1$

code transmis $C(x) = x^7+x^3+x$ soit **1 0 0 0 1 01 0**

si une erreur intervient sur le bit 5 : syndrome 0 0 1 0 0 0 0 0 soit $E(x) = x^5$

code reçu $C^*(x) = x^7+x^5+x^3+x$
décodage $R(x) = x^2+x+1$ ($Q(x) = x^4+x+1$ soit 1 0 0 1 1)
une erreur est décelée et on demande la répétition du message.

2.3.3. Codage par division

Ce type de codage n'utilise qu'un outil (diviseur polynomial); il concatène la redondance à l'information utile à transmettre et permet de la "lire" éventuellement sans décodage.

Soit **k** le degré du polynôme générateur G(x) et M(x) le message à transmettre.

On calcule $I(x) = x^k * M(x)$.
puis **r(x)** reste de la division de I(x) par G(x) : $I(x) = Q(x)*G(x) + \mathbf{r(x)}$
Le message à transmettre est $C(x) = I(x) + r(x)$
(en pratique I(x) est constitué de M(x) suivi de r(x))

C(x) est bien un mot du code, multiple de G(x) (on ne doit pas oublier en effet,que dans cette algèbre, l'addition est le "ou exclusif" et que 1 + 1 = 1 - 1 = 0).

Le message reçu est $C^*(x) = C(x) + E(x)$
A la réception, $C^*(x)$ est divisé par G(x) : $C^*(x) = Q^*(x)* G(x) + R(x)$

Si R(x) est nul, il suffit d'**éliminer les k derniers bits** correspondant à r(x) pour **retrouver M(x) .** Sinon on répétera le message.
R(x) permet, comme ci-dessus de retrouver le syndrome d'erreur et de le corriger.

D'un point de vue **algorithme informatique**, il suffit d'une fonction de division polynomiale. Le message **M** à coder est placé dans un buffer, suivi de 2 (ou 4) octets à 0 pour un polynôme générateur **G** de degré 16 (ou 32) pour obtenir **I.**

On calcule alors **r** que l'on place dans ces 2 (ou 4) octets de fin de buffer.

A la réception, après division par G, il suffit d'éliminer ces 2 (ou 4) octets pour retrouver **M** lorsqu'il n'y pas d'erreur.

Nota :

On peut ajouter une constante au codage en initialisant les octets de fin à une valeur différents de 0 et en leur ajoutant (ou exclusif) r. Le reste R, en cas de transfert sans erreur n'est pas nul. (voir protocole HDLC)

2.4. Autre codage

Les codages précédents sont faciles à mettre en oeuvre par des moyens matériels et la fonction codage/décodage par code cyclique est intégrée aux circuits de communication synchrones.

D'autres systèmes peuvent être utilisées, en particulier le code suivant, normalisé pour la détection d'erreurs au niveau 4/OSI (Transport).

Le champ de contrôle placé en fin de PCI est composé de deux octets que nous noterons X et Y . Si i est le numéro d'un octet de la PDU, a_i la valeur de cet octet et L la longueur en octets de la PDU, les valeurs de X et Y sont telles que :

$$\sum_{i=1}^{L} a_i = 0 \quad \text{(modulo 255)}$$

$$\sum_{i=1}^{L} i*a_i = 0 \quad \text{(modulo 255)}$$

On utilise en émission et réception deux variables intermédiaires C_0 et C_1 . On note aussi n la position du premier octet de contrôle (X) dans la PDU (numéro d'octet) .

à l'émission :

Initialiser X, Y, C_0 et C_1 à 0
Pour chaque octet i de 1 à L :
  ajouter a_i à C_0
  ajouter C_0 à C_1
Calculer $X = - C_1 + (L - n) * C_0$
$Y = C_1 - (L - n + 1) * C_0$
Placer X et Y dans les octets n et n + 1

à la réception :

Initialiser C_0 et C_1 à 0
Pour chaque octet i de 1 à L :
ajouter a_i à C_0
ajouter C_0 à C_1
Si C_0 et C_1 ne sont pas nuls, la PDU est incorrecte .

Cet algorithme calcule :

$$C1 * \sum_{i=1}^{L} (L - i + 1)\, a_i$$

qui doit être égal à zéro puisque :

$$\sum_{i=1}^{L} (L - i + 1)\, a_i = (L + 1) * \sum_{i=1}^{L} a_i - \sum_{i=1}^{L} i * a_i = 0$$

2.5. Mise en oeuvre

2.5.1. Choix du code

En pratique on utilise des codes normalisés de degré 16 (CRC16 et CCITT) ou 32 (voir réseaux locaux).

Ces codes sont mis en oeuvre ("au fil de l'eau") par les circuits de communication synchrones utilisés. Ces codes protègent une taille maximale de données (par exemple 32 ko pour un polynôme de degré 16 de type $(x+1)*(X^{15}+...)$. Ces codes détectent toutes les erreurs simples, doubles et d'ordre impair.

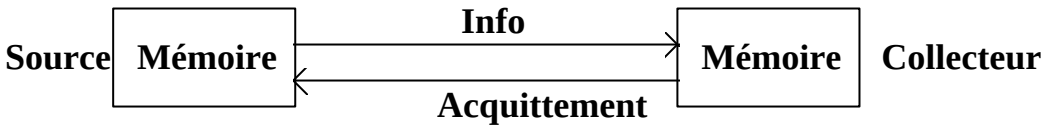
Pour des applications spécifiques, on peut utiliser d'autres codes cycliques (ou linéaires). Les codes de distance de Hamming 3 sont dits codes de Hamming. Si k est le degré de G(x), ils sont optimaux pour une taille de C(x) de 2^n - 1 bits. Pour des séquences plus courtes, ils sont dits "tronqués".

Pour obtenir une puissance plus élevée, on utilise en général des codes BCH : Bose-Chaudhuri-Hockengheim. Ces auteurs ont donné un algorithme qui fournit une condition suffisante pour construire un code de distance de Hamming donnée.

2.5.2. modes d'utilisation

Les systèmes de transmission permettant de mémoriser les données coté source et destination, un système **de correction par détection et répétition** des blocs erroné peut être mis en oeuvre; un tel système est beaucoup moins coûteux ou plus performant qu'une correction directe d'erreurs.

Dans les plus ancien systèmes, la mémoire étant très onéreuse, les mêmes buffers étaient utilisés pour l'émission et la réception des données. Un petit buffer supplémentaire est seul nécessaire pour préparer ou recevoir les accusés de réception positifs (pas d'erreur de transmission) ou négatif en retour.

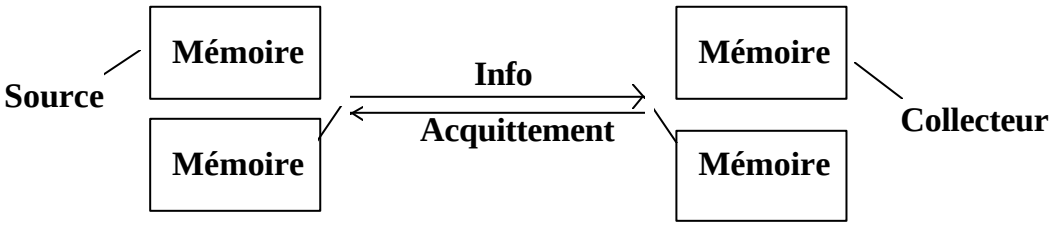


Les protocoles utilisés fonctionnent alors à l'alternat avec les phases successives suivantes :

Préparation du bloc à émettre dans le buffer d'émission
Transfert du bloc
Analyse du bloc reçu dans le buffer de réception
Préparation de l'accusé de réception
Transmission de l'accusé de réception
Analyse de l'accusé de réception

La somme des temps nécessaires à ces 6 phases constitue le "temps de transmission en boucle" **t**.

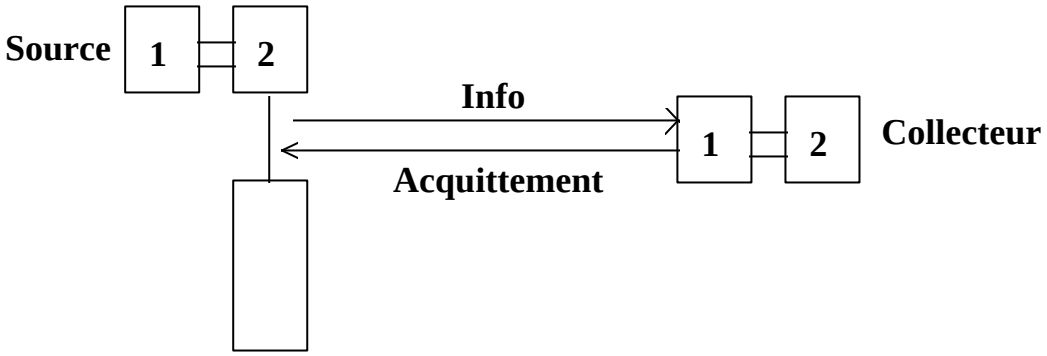
Pour améliorer les performances on peut utiliser 2 buffers en bascule. Le bloc n+1 est préparé dans un buffer durant la transmission de l'autre buffer. Si l'accusé de réception est positif on transmet le bloc n+1 et on prépare le suivant; sinon on réémet le bloc n.



Ce système, pour être utilisé de manière optimale, sans perte de temps, lie la taille des blocs au temps de préparation, de transfert et d'analyse des accusés de réception.

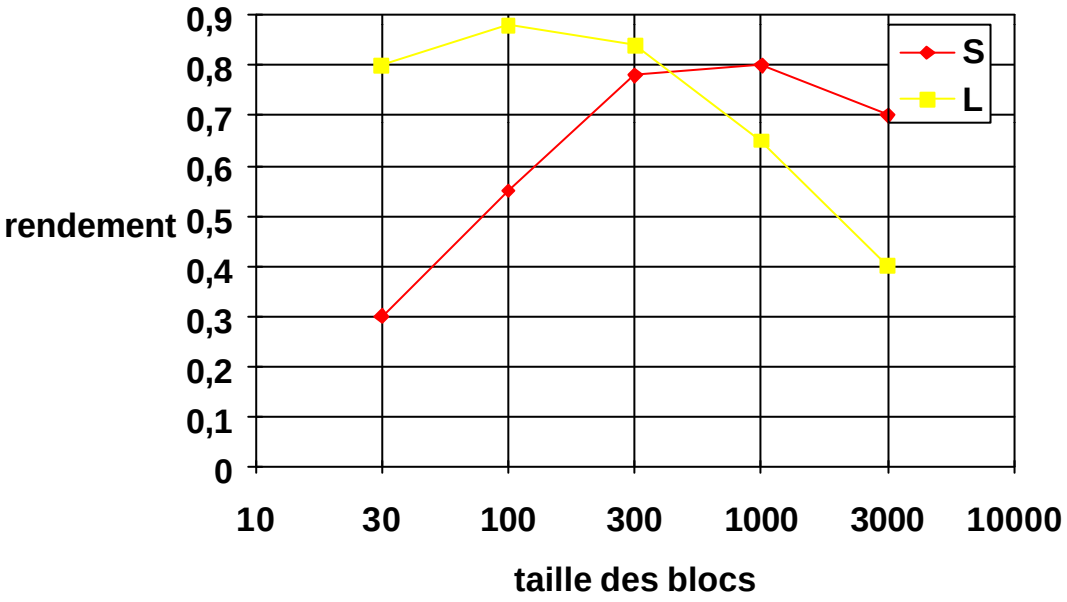
Pour lever cette contrainte, on peut transmettre des blocs plus courts qui sont stockés en attente d'acquittement coté émetteur. Les buffers correspondants sont libérés par les accusés de réception qui peuvent être groupés. Le système fonctionne avec anticipation. En cas

d'erreur sur le bloc n, on sera amené à répéter les blocs n+1, n+2, ... transmis, probablement sans erreur, avant l'arrivée de l'accusé de réception négatif du bloc n.



Le choix d'un taux d'anticipation optimal est lié au taux d'erreur observé sur la liaison. En cas de taux d'erreur élevé, un taux d'anticipation nul, avec attente de l'accusé de réception du bloc n avant transmission du bloc n+1 peut s'avérer profitable (en ne répétant pas inutilement les blocs suivants).

Le diagramme ci-dessous montre que si le système adapté au temps de transmission en boucle peut offrir un rendement voisin de 90%, il travaille avec des blocs plus cours et se dégrade plus vite qu'un système plus simple.



Si le taux d'erreurs naturel est faible on aura intérêt à travailler avec anticipation en n'acquittant pas tous les blocs et en prenat le risque de retransmettre plusieurs blocs, éventuellement correct. Si il est très élevé, il est préférable d'attendre l'acquittement d'un bloc avant de transmettre le suivant. En période très perturbée on réduit parfois ainsi le rendement mais aussi le risque d'erreurs non décelées (en diminuant le nombre de trames transmises dans ces périodes).