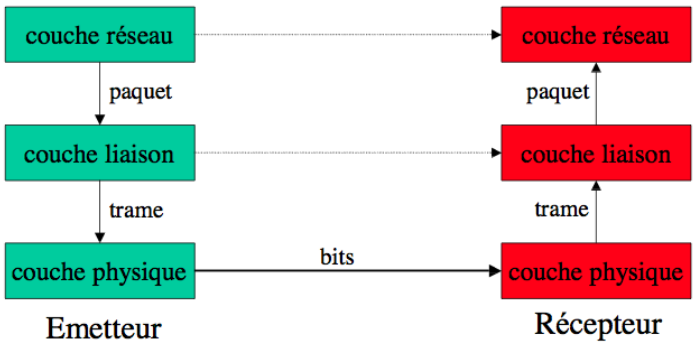
Chapitre 3

Fiabilisation de la transmission

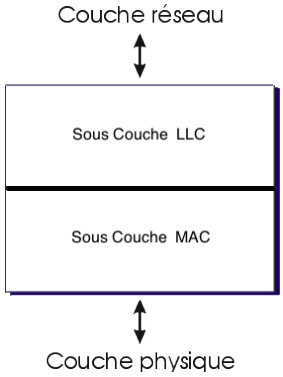
La couche physique permet de transmettre un flot de bits entre deux systèmes distants. La couche liaison (couche n° 2) doit s'assurer de la corrections des bits transmis et les rassembler en paquets pour les passer à la couche Réseaux. La couche liaison récupère des paquets de données de la couche réseau, les enveloppe en des trames qui les envoie une à une à la couche physique.



Pour rendre la transmission fiable, la couche liaison doit assurer :

- La délimitation des blocs de données échangées ;
- Le contrôle de l'intégrité des données reçues ;
- L'organisation et le contrôle de l'échange.
- Le contrôle d'accès à un canal partagé

En fait, la couche liaison se compose de deux sous-couches : LLC (Logical Link Control) et MAC (Media Access Control). La sous-couche LLC Assure les trois premières fonctions tandis que la sous-couche MAC assure la dernière.



3.1 Délimitation de trames

A l'instar des transmissions asynchrones où les bits de start et de stop encadrent les bits d'information, en transmission synchrone une information spéciale permet de repérer le début et la fin des données transmises. Il existe deux méthodes :

3.1.1 Comptage des caractères

On utilise un champ dans l'entête de la trame pour indiquer le nombre de caractères de la trame.

06	'S'	'U'	'P'	'E'	'R'	03	'L'	'E'	06	'C'	'O'	'U'	'R'	'S'

Un problème sérieux peut se poser avec cette méthode si la valeur du champ ajouté est modifiée au cours de la transmission. Généralement, cette méthode est rarement utilisée seule.

3.1.2 Utilisation des fanions

La trame est délimité par une séquence particulière de bits appelée **fanion** ou flag.

Fanion	DONNEES	Fanion

Pour garantir le bon fonctionnement de cette méthode, des bits de transparence sont nécessaires pour qu'une séquence binaire dans la trame ne corresponde accidentellement au fanion. Par exemple, si le fanion est l'octet 01111110, un bit de transparence "0" est inséré après toute séquence de cinq 1 successifs dans la trame.

Exemple :

Données :

01011001111110

Trame :

01111110 010110011111010 **01111110**

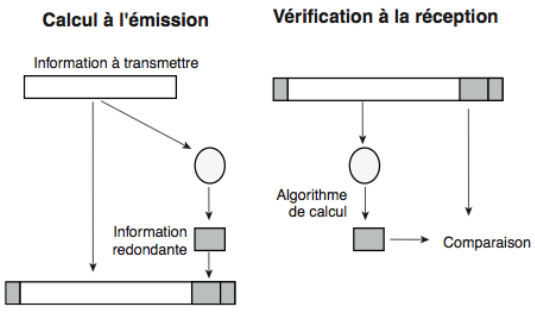
Cette technique est utilisée également en considérant des caractères de délimitation et des caractères de transparence.

L'avantages des fanions est qu'ils permettent de trouver toujours la synchronisation et d'envoyer des trames de tailles quelconques.

3.2 Détection et Correction d'erreurs

D'une manière générale on doit, lors d'une transmission de données, s'assurer que les données reçues n'ont pas été altérées durant la transmission. Plusieurs facteurs peuvent modifier le contenu des données tel que les interférences causées par des rayonnements électromagnétiques ou la distorsion des câbles de transmissions.

Les méthodes de protection exploitent la redondance de données en ajoutant des bits de contrôle aux bits de données. Les bits de contrôle sont calculés, au niveau de l'émetteur, par un algorithme spécifié dans le protocole à partir du bloc de données. À la réception, on exécute le même algorithme pour vérifier si la redondance est cohérente. Si c'est le cas, on considère qu'il n'y a pas d'erreur de transmission et l'information reçue est traitée ; sinon, on est certain que l'information est invalide.



Un exemple des bits de contrôle est la duplication des bits transmis (détection par répétition). Le message code est un double exemplaire du message initial, le récepteur sait qu'il y a eu erreur si les exemplaires ne sont pas identiques, il demande alors la

retransmission du message. Si la même erreur se passe sur les deux exemplaires, l'erreur ne sera pas détectée.

Si on envoie le message en trois exemplaires, le récepteur pourra même corriger l'erreur en prenant les valeurs des deux copies identiques sans demander la retransmission de l'émetteur.

3.2.1 Définitions générales

Un code $C(k, n)$ transforme (code) tout bloc initial de k bits d'information en un bloc codé de n bits. Le code ajoute une redondance puisque $n \geq k$.

3.2.1.1 Code Systématique

Un code est dit systématique si les k premiers bits du bloc codé sont égaux aux bits du bloc initial. Alors les r $(r = n - k)$ derniers bits forment un champ de contrôle d'erreur.

3.2.1.2 Rendement d'un code

Le rendement R d'un code $C(k, n)$ est donné par le rapport entre le nombre de bits du bloc initial et le nombre de bits du bloc codé : $R = \frac{k}{n}$

3.2.1.3 Mot de code

On appelle mot du code, la suite de n bits obtenue après un codage $C(k, n)$. Le nombre n de bits qui composent un mot du code est appelé la longueur du code. La dimension k étant la longueur initiale des mots.

3.2.1.4 Poids de Hamming

Le poids de Hamming d'un mot est le nombre de bits à 1 qu'il contient. Par exemple le poids de Hamming de la séquence 01110110 est 5.

3.2.1.5 Distance de Hamming d'un code

La distance de Hamming entre deux mots de même longueur est définie par le nombre de positions binaires qui diffèrent entre ces deux mots. On l'obtient par le poids de Hamming de la somme binaire des 2 mots.

La distance de Hamming d'un code est la distance minimum entre tous les mots du code. Soit C un code ;

$$
\begin{array}{lll}
 & 1\,0\,0\,0\,1\,0\,0\,1 & \\
\oplus & 1\,0\,1\,1\,0\,0\,0\,1 & \\
\hline
= & 0\,0\,1\,1\,1\,0\,0\,0 & \text{Dist(x,y)} = 3
\end{array}
$$

$$Dist(C) = min\{Dist(x, y) \ / \ x \in C \wedge y \in C\}$$

3.2.1.6 Capacité d'un code

La capacité de détection (de correction) d'un code est définie par les configurations erronées qu'il est capable de détecter (corriger).

Une erreur simple (resp. double, ou d'ordre p) affecte une seule (resp. 2, ou p) position(s) binaire(s) d'un mot.

Pour qu'un code ait une capacité de détection (resp. correction) des erreurs d'ordre e, il faut que sa distance de Hamming soit supérieure ou égale à $(1 + e)$ (resp. $(1 + 2e)$).

Exemple : Si la distance de Hammind d'un code est égale à 3, sa capacité de détection est de 2, et sa capacité de correction est 1.

3.2.2 Code de contrôle de parité

C'est un code systématique $(k, k+1)$ dans lequel un bit (le bit de parité) est ajouté au mot initial pour assurer la parité. Son rendement est faible lorsque k est petit.

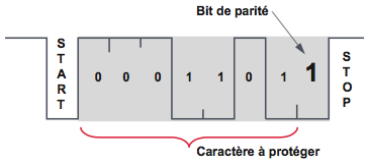
Exemple : Transmission de caractères utilisant un code de représentation (le code ASCII sur 7 bits).

Lettre	Code ASCII	Mot codé (parité paire)	Mode codé (parité impaire)
E	1010001	10100011	10100010
V	0110101	01101010	01101011
A	1000001	10000010	10000011

Ce code est capable de détecter toutes les erreurs en nombre impair mais il ne détecte pas les erreurs en nombre pair. Il permet de détecter une erreur de parité, mais ne permet pas de la localiser.

3.2.2.1 Parité verticale

À chaque caractère on rajoute un bit (bit de redondance verticale ou bit de parité, VRC :Vertical Redondancy Check)



3.2.2.2 Parité longitudinale

A chaque bloc de caractère, on ajoute un champ de contrôle supplémentaire (LRC : Longitudinal Redondancy Check)

Caractère à transmettre	bit de parité	Caractère à transmettre	bit de parité	...	Caractère LRC	bit de parité

La parité longitudinale était initialement utilisée pour les bandes magnétiques pour compléter la détection des erreurs de parité verticale.

3.2.2.3 Parité longitudinale et verticale

Le bloc de données est disposé sous une forme matricielle ($k = a \bullet b$). On applique la parité sur chaque ligne et chaque colonne. On obtient une matrice $(a+1, b+1)$. Un caractère le LRC est ajouté au bloc transmis. Chaque bit du caractère LRC correspond à la parité des bits de chaque caractère de même rang : le premier bit du LRC est la parité de tous les 1^{er} bits de chaque caractère, le second de tous les 2^e bits... Le caractère ainsi constitué est ajouté au message. Le LRC est lui-même protégé par un bit de parité (VRC).

	H	E	L	L	O	*LRC →*
bit 0	0	1	0	0	1	0
bit 1	0	0	0	0	1	1
bit 2	0	1	1	1	1	0
bit 3	1	0	1	1	1	0
bit 4	0	0	0	0	0	0
bit 5	0	0	0	0	0	0
bit 6	1	1	1	1	1	1
VRC ↓	0	1	1	1	1	0

1001000	0	1000101	1	1001100	1	1001100	1	1001111	1	1000010	0
H		E		L		L		O		LRC	

3.2.3 Code de Hamming

Le code de Hamming permet de détecter et de corriger une erreur survenue dans un bloc transmis. Aux d bits de données, on ajoute c bits de contrôle de parité. On a donc $d+c=n$ bits. Puisque les c bits de contrôle doivent indiquer les $n+1$ possibilités d'erreurs (dont l'absence d'erreur, ce qui explique le $+1$), il faut que $2^c \geq n+1$. Les 2^c possibilités

de codage sur les c bits servent à coder la position de l'erreur. Dès que cette position est calculée, on peut corriger le bit en erreur.

On numérote les bits de chaque mot de code de droite à gauche à partir de 1, les bits de contrôle sont placés aux positions représentant des puissances de 2 (1,2,4,8,...) et les bits de données sont intercalés.

Exemple :

					1	0	0	1	0	0	0
							+	c_3	c_2	c_1	c_0
=	1	0	0	c_3	1	0	0	c_2	0	c_1	c_0
	11	10	9	8	7	6	5	4	3	2	1

La liste des bits de parité contrôlant un bit donné est fourni par les positions à 1 dans sa représentation binaire. Par exemple :

- le bit numéro $11 = (1011)_2$ est contrôlé par les bits c_3, c_1 et c_0
- le bit numéro $7 = (0111)_2$ est contrôlé par les bits c_2, c_1 et c_0
- le bit numéro $4 = (0100)_2$ est contrôlé par le bits c_2
- ...

Ainsi, la liste des bits contrôlée par chaque bit de parité est :

- c_0 : {1,3,5,7,9,11}
- c_1 : {2,3,6,7,10,11}
- c_2 : {4,5,6,7}
- c_3 : {8,9,10,11}

Dans l'exemple précédent et pour une parité paire, les c_i peuvent être calculé comme suit :

- $c_0 = 0(0+0+0+1+0+1)$
- $c_1 = 0(0+0+0+1+0+1)$
- $c_2 = 1(1+0+0+1)$
- $c_3 = 1(1+0+0+1)$

Et le mot de code envoyé sera donc : 10011001000

À la réception, un mot binaire est construit de longueur c qui peut être déduite à partir de la longueur du mot de code reçu (pour coder 11 positions on a besoin de 4 bits). Chaque bit est égal à 1 si le contrôle de parité est OK et 0 sinon. Si le mot est égal à 0, on conclut qu'il n'y a pas eu d'erreur, sinon la valeur du mot représente le numéro du bit erroné.

Par exemple si on reçoit 10011001000, on calcule les c_i :

– $c_0 = 0+0+0+1+0+1 = 0$

– $c_1 = 0+0+0+1+0+1 = 0$

– $c_2 = 1+0+0+1 = 0$

– $c_3 = 1+0+0+1 = 0$

On conclut donc que le mot reçu est correct.

Si par contre on reçoit 1$\underline{1}$011001000 on trouve :

– $c_0 = 0+0+0+1+0+1 = 0$

– $c_1 = 0+0+0+1+1+1 = 1$

– $c_2 = 1+0+0+1 = 0$

– $c_3 = 1+0+1+1 = 1$

Le mot de contrôle est donc $(1010)_2 = (10)_{10}$, on conclut que le bit numéro 10 est erroné.

Exercice : Le codage de Hamming est utilisé . Un récepteur reçoit : 1110101. Que peut-on en conclure ?

Calcul simplifié du code de Hamming

Coder 10101011001 avec une parité paire : $d = 11$, donc $k = 4$. Le message à transmettre contient donc $n = 15$ bits et est égal à :

1	0	1	0	1	0	1	?	1	0	0	?	1	?	?
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Dans le message à transmettre, on a des bits à 1 dans les positions suivantes : 15, 13, 11, 9, 7, 3. On transforme ces positions en leur valeur binaire et on les additionne modulo 2 : On met 1 lorsque l'on a un nombre impair de 1 et 0 pour un nombre pair de 1.

15	=	1	1	1	1
13	=	1	1	0	1
11	=	1	0	1	1
9	=	1	0	0	1
7		0	1	1	1
3	=	0	0	1	1
c_i		0	1	0	0

Le message codé est donc 101010101001100.

Le même procédé peut être fait à la réception pour vérifier le mot reçu.

3.2.4 Codes polynomiaux

La méthode des codes polynômiaux (ou le CRC : Cyclic Redondant Coding) est la méthode la plus utilisée pour détecter des erreurs groupées. Avant la transmission, on ajoute des bits de contrôle. Si des erreurs sont détectées à la réception, il faut retransmettre le message.

Dans ce code, une information de n bits est considérée comme la liste des coefficients binaires d'un polynôme de n termes, donc de degré $n-1$.

Exemple :

- $1101 \rightarrow x^3 + x^2 + 1$
- $110001 \rightarrow x^5 + x^4 + 1$
- $11001011 \rightarrow x^7 + x^6 + x^3 + x + 1$

Pour calculer les bits de contrôle, on effectue un certain nombre d'opérations avec ces polynômes à coefficients binaires. Toutes ces opérations sont effectuées modulo 2. C'est ainsi que, dans les additions et dans les soustractions, on ne tient pas compte de la retenue : Toute addition et toute soustraction sont donc identiques à une opération XOR. Par exemple :

	1	0	0	1	1	0	1	1
+	1	1	0	0	1	0	1	0
=	0	1	0	1	0	0	0	1

	1	1	1	1	0	0	0	0
-	1	0	1	0	0	1	1	0
=	0	1	0	1	0	1	1	0

La source et la destination choisissent un même polynôme $G(x)$ dit générateur car il est utilisé pour générer les bits de contrôle (Checksum).

L'algorithme pour calculer le message à envoyer est le suivant. Soit $M(x)$ le polynôme correspondant au message original, et soit r le degré du polynôme générateur $G(x)$ choisi :

- Multiplier $M(x)$ par x^r, ce qui revient à ajouter r zéros à la fin du message original
- Effectuer la division suivante modulo 2 :

$$\frac{M(x)x^r}{G(x)} = Q(x) + R(x)$$

– Le quotient $Q(x)$ est ignoré. Le reste $R(x)$ (Checksum) contient r bits (degré du reste $r-1$). On effectue alors la soustraction modulo 2 :

$$M(x).x^r - R(x) = T(x)$$

Le polynôme $T(x)$ est le polynôme cyclique : c'est le message prêt à être envoyé. Le polynôme cyclique est un multiple du polynôme générateur $T(x) = Q(x).G(x)$

A la réception, on effectue la division suivante :

$$\frac{T(x)}{G(x)}$$

– Si le reste = 0, il n'y a pas d'erreur

– Si le reste $\neq 0$, il y a erreur, donc on doit retransmettre

En choisissant judicieusement $G(x)$, on peut détecter toute erreur sur 1 bit, 2 bits consécutifs, une séquence de n bits et au-delà de n bits avec une très grande probabilité.

Exemple

Soit à transmettre le message 1011011 en utilisant le polynôme générateur $G(x) = x^4 + x + 1$. On procède comme suit pour calculer le message à transmettre

1. message original = 1011011 $\Rightarrow M(x) = x^6 + x^4 + x^3 + x^1 + 1$
2. $G(x) = x^4 + x + 1$
3. $M(x).x^4 = x^10 + x^8 + x^7 + x^5 + x^4$
4. Calculer $R(x)$ par l'une des deux méthodes :
   (a) **Division polynomiale**

x^{10}	+	x^8	+	x^7	+	x^5	+	x^4	x^4	+	x	+	1
x^{10}	+	x^7	+	x^6					x^6	+	x^4	+	x^2
x^8	+	x^6	+	x^5	+	x^4							
x^8	+	x^5	+	x^4									
x^6													
x^6	+	x^3	+	x^2									
		x^3	+	x^2									

   (b) **Soustraction binaire**

$$
\begin{array}{cccccccccccc}
 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
- & 1 & 0 & 0 & 1 & 1 & & & & & & \\
\hline
 & 0 & 0 & 1 & 0 & 1 & \mathbf{1} & \mathbf{1} & & & & \\
 & & - & 1 & 0 & 0 & 1 & 1 & & & & \\
\hline
 & & & 0 & 0 & 1 & 0 & 0 & \mathbf{0} & \mathbf{0} & & \\
 & & & & - & 1 & 0 & 0 & 1 & 1 & & \\
\hline
 & & & & R= & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0}
\end{array}
$$

5. $R(x) = x^3 + x^2 = (1100)_2$
6. Le message à envoyer $T(x) = M(x).x^r - R(x) = x^10 + x^8 + x^7 + x^5 + x^4 - x^3 - x^2 = (10110111100)_2$

À la réception, un calcul semblable s'effectue sur le mot reçu, mais il faut, ici, que le reste soit nul. Dans le cas contraire, c'est qu'une erreur est survenue en cours de route.

Codes polynomiaux utilisés

Les principaux polynômes générateurs (diviseurs) sont :

- LRCC-8 : $x^8 + 1$
- LRCC-16 : $x^{16} + 1$
- CRC 12 : $x^{12} + x^{11} + x^3 + x^2 + x + 1$
- CRC 16 Forward : $x^{16} + x^{15} + x^2 + 1$
- CRC 16 Backward : $x^{16} + x^{14} + x + 1$
- CRC CITT Forward : $x^{16} + x^{12} + x^5 + 1$
- CRC CITT Backward : $x^{16} + x^{11} + x^4 + 1$

3.2.5 Codes auto-correcteurs

Un code de distance de Hamming α peut :

- détecter toute erreur portant sur $\alpha - 1$ bits
- corriger toute erreur portant sur $\frac{\alpha-1}{2}$ bits

Supposons le code suivant :

Mot naturel	Mot de code
00	10011
01	10100
10	01001
11	01110

La distance de Hamming du code est 3, il peut donc détecter des erreurs sur 2 bits et corriger des erreurs sur un bit.

Par exemple, si on veut transmettre 00(10011) et une erreur survient sur un bit, les possibilités sont :00011, 11011, 10111, 10001, 10010.

Supposons recevoir 11011.On Calcule la distance de Hamming avec tous les mots de code :

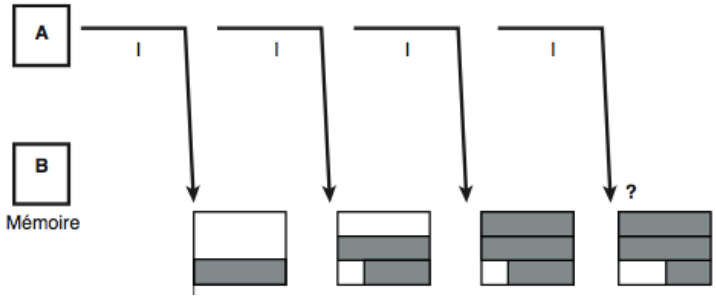
- $D(11011, 10011) = 1$
- $D(11011, 10100) = 4$
- $D(11011, 01001) = 2$
- $D(11011, 01110) = 3$

Le mot correcte est le mot plus proche 10011 c-à-d 00.

Remarquons que si une erreur survient sur 2 bits, le nombre de possibilités augmente et on tombe sur plusieurs mots proches et la décision devient impossible.

3.3 Contrôle de flux

Dans une transmission d'information d'un émetteur A vers un récepteur B, si l'émetteur produit les données à une vitesse nettement supérieure à la vitesse de consommation du récepteur, ce dernier sera engorgé (saturé ou surchargé) et les informations émises seront perdues. Pour résoudre ce problème, on peut penser à doter le récepteur d'une mémoire tampon lui permettant de stoker les messages en attendant leur traitement. On peut constater rapidement que quelque soit la taille de la mémoire utilisée, elle peut être saturée.



Le contrôle de flux sert à mettre en place un mécanisme de contrôle du rythme d'envoi des informations vers le récepteur. Il peut être réalisé par plusieurs méthodes :

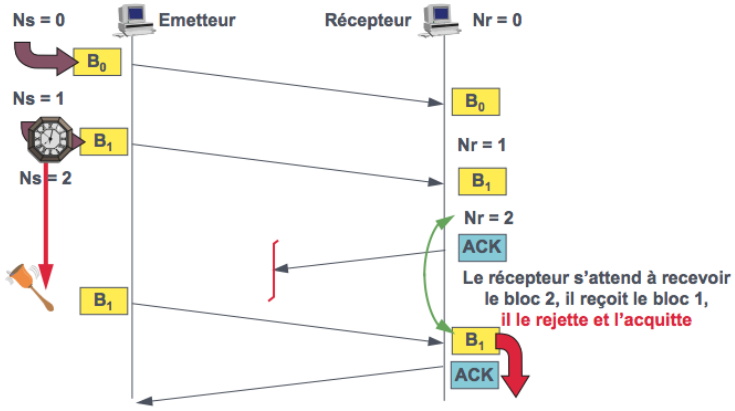
3.3.1 Envoyer et attendre (Sent and Wait)

Après l'envoi d'un bloc d'information, L'émetteur s'arrête dans l'attente d'un accusé de réception. À la réception de l'acquittement, noté ACK pour Acknowledge, l'émetteur envoie le bloc suivant.

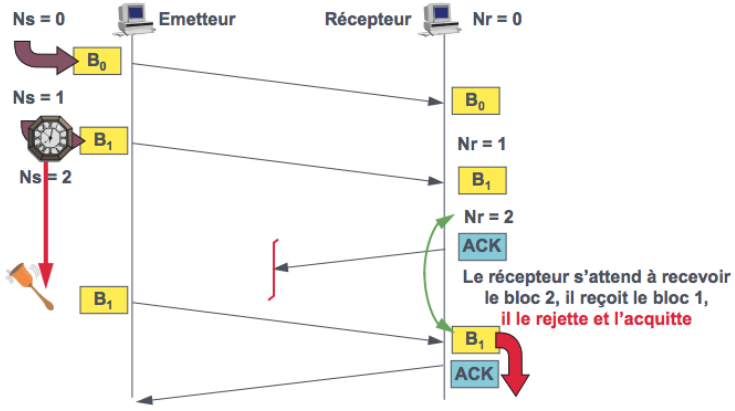
En cas d'erreur de transmission, le bloc reçu est rejeté. Le bloc est dit perdu, il n'est pas acquitté. L'émetteur reste alors en attente. Pour éviter un blocage de la transmission, à l'émission de chaque bloc de données, l'émetteur arme un temporisateur (Timer). À l'échéance du temps imparti (Time Out), si aucun accusé de réception (ACK) n'a été reçu, l'émetteur retransmet le bloc non acquitté.

Une difficulté survient si la perte concerne l'ACK. En effet, bien que les données aient été correctement reçues, l'émetteur les retransmet sur temporisation. Les informations sont ainsi reçues 2 fois. Pour éviter la duplication des données, il est nécessaire d'identifier les blocs. À cet effet, l'émetteur et le récepteur entretiennent des compteurs Ns (Ns, Numéro émis, s pour send) et Nr (Numéro du bloc à recevoir, r pour receive).Les deux compteurs sont initialisés à zéro. Le contenu du compteur Ns est transmis avec le bloc, le récepteur compare ce numéro avec le contenu de son compteur Nr. Si les deux valeurs sont identiques

le bloc est réputé valide et accepté. Si les valeurs diffèrent, le bloc reçu n'est pas celui attendu. Il est rejeté et acquitté s'il correspond à un bloc déjà reçu.



Dans les cas où les délais de consommation sont plus importants, les données peuvent ne pas être acquittées à temps. Par exemple, si A transmet un bloc B0 et B se tarde dans son traitement, A va retransmettre B0 avant de recevoir l'acquittement. Si A transmet un nouveau bloc B1 et se perd, il va considérer l'acquittement du deuxième B0 comme un acquittement de B1.



Pour éviter cette confusion d'interprétation, il est aussi nécessaire de numéroter les ACK.

Le temps d'attente des acquittements rend la méthode send and wait peu efficace. En plus, il est unidirectionnel.

3.3.2 Fenêtre d'anticipation

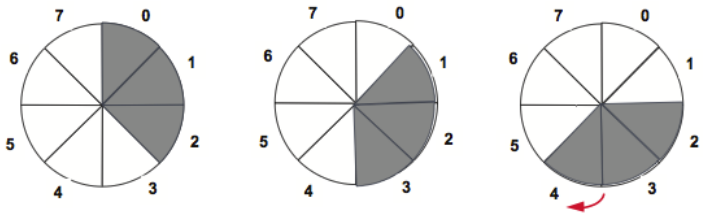
Pour améliorer le protocole précédent et réduire le temps d'attente des acquittements, on émet plusieurs blocs sans attendre les ACK, ce processus se nomme anticipation. Ainsi,

un acquittement n'acquitte plus une seule trame mais un ensemble de trames qui se suivent sans erreur. Le nombre de trames successives qu'on peut émettre sans réception d'acquittement est limité par une valeur notée W, appelée fenêtre (Window). Le principe est d'autoriser l'émetteur à envoyer les trames de numéro de séquence compris entre le numéro r de la prochaine trame attendue (communiqué par le récepteur) et $r + W - 1$:

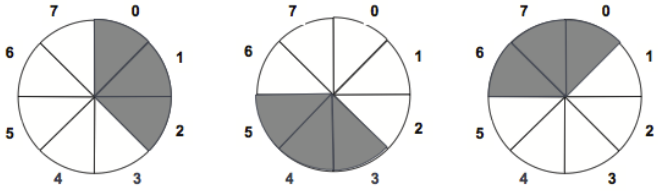
$$r \leq Ns \leq r + W - 1$$

Remarque : $W = 1$ dans le cas d'une procédure Send-and-Wait.

Pour pouvoir réenvoyer les trames en cas d'erreur, l'émetteur met les blocs non acquittés dans W mémoires tampons. À la réception d'un acquittement d'une trame, l'émetteur libère la mémoire correspondante et envoie une nouvelle trame. La fenêtre est dite, dans ce cas, *"glissante"*. Par exemple, si les trames sont numérotées de 0 à 7(sur 3 bits, numérotation modulo 8) et $W = 3$ les l'émission sera semblable à la figure suivante :



Le récepteur peut grouper les acquittements en répondant par un seul acquittement pour confirmer la bonne réception de plusieurs blocs. Dans ce cas, la fenêtre est dite sautante.



Problème : que se passe-t-il si, de façon temporaire, le récepteur n'est pas prêt à recevoir les W trames d'information de la fenêtre ? L'utilisation d'une fenêtre d'anticipation peut nécessiter la mise en œuvre d'un mécanisme de régulation supplémentaire de type tout-ou-rien. L'idée est d'utiliser une trame de contrôle particulière est utilisée pour indiquer que le récepteur est momentanément dans l'incapacité de continuer à recevoir. L'émetteur recevant cette trame de contrôle cesse immédiatement toute émission (même s'il n'avait pas utilisé toute sa "fenêtre" d'émission). Une autre trame de contrôle est alors nécessaire

pour indiquer à l'émetteur que le récepteur est revenu dans un état normal et qu'il est donc prêt à recevoir de nouvelles trames.

3.4 Procédures de gestion des données

Les procédures de gestion de données sont des protocoles de la couche liaison qui mettent en œuvre les techniques précédentes (délimitation des trames, correction des erreurs et contrôle de flux). Elles sont de deux familles :

1. Les procédures orientées caractère : qui fonctionnent généralement à l'alternat (de type send and wait).
2. Les procédures orientées bit : conçues pour les transmissions bidirectionnelles simultanées à hauts débits.

3.4.1 Procédure BSC

BSC (Binary Synchronous Communications) est une procédure orientée caractère développée par IBM pour la transmission synchrone des blocs de caractères. Elle permet d'exploiter principalement deux types de liaisons :

1. Les liaisons point à point : où les stations communiquent à l'alternat (half duplex) et en cas de conflits, une station privilégiée dite station primaire prend le contrôle de la station secondaire.
2. Les liaisons multipoints : où la station primaire représente le nœud central par rapport aux stations secondaires.
   - Pour la réception, on utilise le mode *polling* (interrogation). La station primaire invite les stations secondaires à émettre chacune à son tour. La station consultée répond par un message si elle a quelque chose à émettre, sinon elle répond négativement.
   - Pour l'émission, on utilise le mode *adressing* ou *selecting*. La station primaire envoie un message de sélection à la station secondaire pour l'informer qu'elle veut émettre un message ; la station secondaire répond positivement si elle est prête à recevoir, négativement si elle n'est pas prête.

Le codage des messages utilisé par BSC repose essentiellement sur les principaux alphabets internationaux tel que le code ASCII. La procédures réserve des caractères du code (caractères de contrôle) pour la gestion de la liaison de données. Certains caractères ont une fonction universellement reconnue, tel que :

- SYN (SYNchronous idle) : utilisé pour assurer la synchronisation caractère. L'émetteur commence par envoyer une séquence de caractères SYN (séquence préfixe) avant tout message ou séquence de supervision.
- **ENQ** (ENQuiry) : placé dans une séquence de supervision. Il invite une station à émettre ou à recevoir.
- **SOH** (Start Of Heading) : signale un début d'en-tête.
- **STX** (Start of TeXt) : délimite à la fois la fin de l'en-tête (lorsqu'il existe) et le début du texte.
- **ETB** (End of Transmission Block) : peut être employé pour distinguer une fin de bloc de données d'une fin de message.
- **ETX** (End of TeXt) : indique la fin d'un texte et le début des caractères de contrôle servant à la détection des erreurs.
- **ACK** (ACKnowledgement) : figure dans une séquence envoyée par le récepteur pour accuser positivement le message qu'il vient de recevoir.
- **NAK** (Négative AcKnowledgement) : figure dans une séquence envoyée par le récepteur pour refuser le message qu'il vient de recevoir. Ce message doit alors être réémis.
- **DLE** (Data Link Escape) : utilisé lors des transmissions en mode transparent (lorsque toutes les configurations de l'alphabet de transmission, y compris les caractères de procédure, sont susceptibles de se retrouver dans le champ de données). Il Signale que le caractère suivant doit être interprété comme un caractère de procédure et non comme un caractère de données (STX, ETB, ETX,...). Lorsque le caractère à transférer est lui-même DLE, il faut émettre deux DLE consécutifs (technique dite de doublement du DLE).
- **EOT** (End Of Transmission) : signale la fin d'un transfert de données.

Les formats des messages utilisés par BSC reposent sur la notion de bloc. Un message est un champ de données d'une taille quelconque, susceptible d'être découpé en blocs de taille optimale. La procédure BSC permet la transmission de nombreux types de messages :

- en-tête seul : SYN, SOH, ...{l'entête}..., ETX, BCC, PAD ;
  - BCC : Block Check Character - PAD (padding) : séquence de remplissage.
- un seul message sans en-tête : SYN, STX,...,ETX, BCC, PAD,
- un seul message avec en-tête : SYN, SOH, ..., STX,...,ETX, BCC, PAD
- plusieurs blocs pour un même message, sans en-tête :
  - SYN, STX,... ,ETB, BCC, PAD pour les premiers blocs,

  - SYN , STX,..., ETX, BCC, PAD pour le dernier bloc du message ;
- plusieurs blocs pour un même message, avec en-tête :
  - SYN, SOH, ..., STX,...,ETB, BCC, PAD pour les premiers blocs,
  - SYN, SOH, ..., STX, ..., ETX, BCC, PAD pour le dernier bloc.

Pour la détection des erreurs, la procédure BSC utilise un CRC 16 avec un polynôme générateur $G(x) = x^{16} + x^{12} + x^5 + 1$

3.4.2 Procédure HDLC

3.4.2.1 Généralités

La procédure HDLC (High-level Data Link Control) est une procédure orientée bit, développée par IBM et normalisée par l'UIT en 1976. HDLC est une procédure point à point et multipoints en full duplex, utilisant des trames séparées par des fanions de valeur 01111110 (7E). Trois modes peuvent être exploités par HDLC :

1. le mode de réponse normal (Normal Response Mode ou NRM) : la station secondaire doit attendre un ordre explicite du primaire avant de pouvoir émettre.
2. le mode de réponse asynchrone (Asynchronous Response Mode ou ARM) :la station secondaire a le droit d'émettre des données sans attendre l'invitation du primaire. Ce mode de fonctionnement est également connu sous le nom protocole LAP (Link Access Protocol). Il suppose que les deux stations possèdent à la fois le statut primaire et le statut secondaire.
3. le mode de réponse asynchrone équilibré ou symétrique (Asynchronous Balanced Mode ou ABM) : la liaison est obligatoirement point à point ; comme pour LAP, les deux stations possèdent à la fois le statut primaire et le statut secondaire. Ce mode de fonctionnement est connu aussi sous le nom de protocole LAP-B (Link Access Protocol-Balanced). De nos jours, c'est le seul mode utilisé.

3.4.2.2 Types de trames HDLC

HDLC utilise trois types de trames :

1. les trames diinformation ou trames **I** : assurent le transfert de données ;
2. les trames de supervision ou trames **S** (Supervisor) : assurent la transmission des commandes de supervision (accusé de réception...),
3. les trames non numérotées ou trames **U** (Unnumbered) : supervisent la liaison (connexion, déconnexion).

3.4.2.3 Structure de la trame HDLC

La trame HDLC est organisée comme suit :

8 bits	8 bits	8 bits	n bits	16 bits	8 bits
fanion	adresse	contrôle	information	FCS	fanion

- Fanion (flag) : 01111110
  . délimite les trames : toutes les trames doivent commencer et finir par un fanion.
  . permet la synchronisation des trames : toutes les stations rattachées à la liaison doivent rechercher en permanence cette séquence ;
  . un même fanion peut servir de fanion de fermeture pour une trame et de fanion d'ouverture pour la trame suivante ;
  . mécanisme de transparence au fanion par bits de bourrage : en émission, un 0 est inséré dès que cinq 1 consécutifs apparaissent en dehors des champs F ; ces 0 sont enlevés en réception. Si sept 1 apparaissent n'importe où dans une trame, elle est déclarée en erreur.
- Champ d'adresse : permet d'identifier la trame comme étant une commande ou une réponse. En mode ABM, les valeurs que peut prendre ce champ sont prédéfinies. Quatre valeurs sont suffisantes pour distinguer les commandes et les réponses dans les deux sens de transmission (ex : 11000000, 10000000, 11110000 et 1110000).
- Champ de contrôle : il indique le type de trame avec les paramètres nécessaires.
- FCS (Frame Check Sequence) : calculé sur les champs d'adresse, de commande et d'information, à partir du code polynômial V.41 ($x^{16} + x^{12} + x^5 + 1$).

3.4.2.4 Champ de contrôle et formats de trame

Il existe trois formats de trame qui correspondent à des codages différents du champ de contrôle :

	1	2	3	4	5	6	7	8
Information	0	Ns			P/F	Nr		
Supervision	1	0	S	S	P/F	Nr		
Non numérotée	1	1	M	M	P/F	M	M	M

- Ns : numéro de séquence de la trame émise,
- Nr : numéro de la prochaine trame attendue (acquittement dans les données),

– P/F (Poll/Final) : Ce bit est positionné à 1 par le primaire lorsque celui-ci sollicite une réponse immédiate du secondaire.

La signification des trames de type S dépend des deux bits SS selon le tableau suivant :

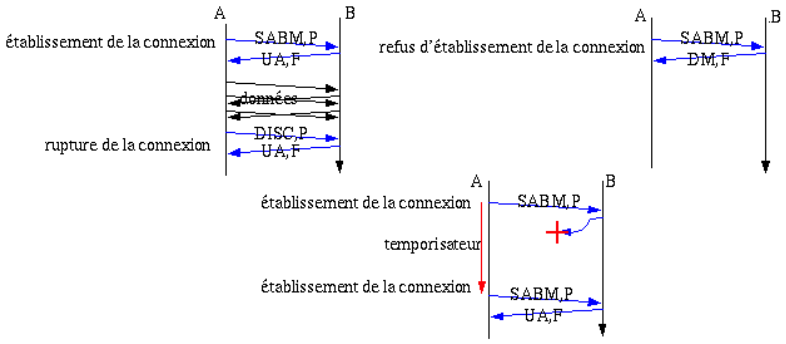
S	S	Commande	Signification
0	0	RR (Receiver Ready)	La station est prête à recevoir la trame numéro Nr et accuse positivement la réception des trames jusqu'à (Nr - 1)
0	1	RNR (Receiver not Ready)	La station n'est pas prête à recevoir des trames mais et accuse positivement la réception des trames jusqu'à (Nr - 1)
1	0	REJ (Reject)	La station rejette les trames à partir du numéro Nr. L'émetteur est obligé de retransmettre (P/F = 1)
1	11	SREJ (Reject)	= REJ mais uniquement pour la trame numéro Nr.

La signification des trames de type U dépend des deux bits M selon le tableau suivant :

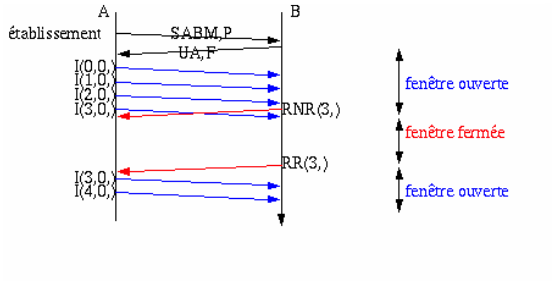
Trame	Commande	Signification
11111100	SABM	Set ABM demande l'établissement en mode ABM
11110000	DM	Disconnect Mode indique que la station se trouve en mode déconnecté
11001010	DISC	Disconnect libère la liaison
11000110	UA	Unnumbered Acknowledge indique la réception et l'acceptation d'une commande non numérotée

3.4.2.5 Exemples des échange

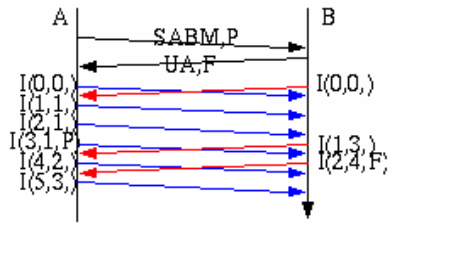
La figure suivante illustre des exemples des échanges entre deux stations utilisant la procédure HDLC pour établir la connexion.



L'exemple suivant, illustre un cas d'échange unidirectionnel.



L'exemple suivant, illustre un cas d'échange bidirectionnel.



L'exemple suivant, illustre un cas d'échange unidirectionnel avec perte de trames.