USTHB-Info |2023

COURS
RÉSEAUX
L3 ACAD

Par
Dr. Khadidja CHAOUI

PLAN

CHAPITRE III

Protocoles de transmission

III.1 Codage et représentation de l'information textuelle

Un code est caractérisé par sa longueur, qui n'est autre que le nombre de bits représentant chaque caractère. Les codes les plus fréquents en téléinformatique ont des longueurs de 5, 6, 7 ou 8 bits permettant de coder respectivement 32, 64, 128 ou 256 caractères différents.

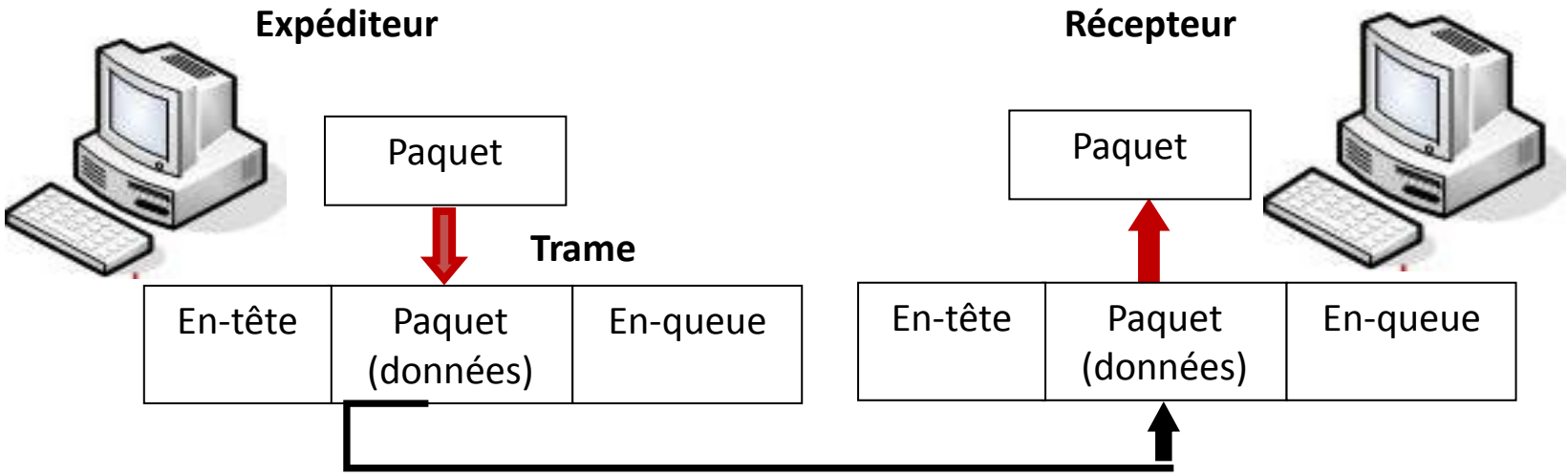
- Code **CCITT** (Comité Consultatif International Télégraphique et Téléphonique) Ce code, encore appelé code Baudot est utilisé pour la transmission sur le réseau télégraphique (TELEX). C'est un code à 5 bits qui ne permet donc que 32 combinaisons
- Code **DCB** (Décimal Code Binaire) ce code à 6 bits a été très utilisé pour la transmission avec des terminaux asynchrones.

- Code **CCITT** n°5 à 7 bits, appelé aussi code **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) ou encore alphabet international n°5 est très utilisé et permet de coder 128 caractères.
- Code **EBCDIC** ce code (**E**xtended **B**inary **C**oded **D**ecimal **I**nterchange **C**ode) à 8 bits et très utilisé du fait que chaque caractère est représenté par un octet.

La **transmission de données** entre machines distantes s'effectue en général en **série par bit** et la c**ommunication entre équipements informatiques** donne lieu à l'échange de messages qui comportent un nombre variables de bits. Les messages qui constituent la partie utile de l'information sont complétés par des informations de contrôle et de supervision telles que l'adresse du destinataire et une séquence de contrôle d'erreurs.

Entête (header) + texte (text) + terminaison (trailer)

La structure d'une **trame** est variable selon le protocole utilisé, mais souvent, elle contient un **entête** (header), **texte** (text) et une terminaison de trame ou **en-queue** (trailer).

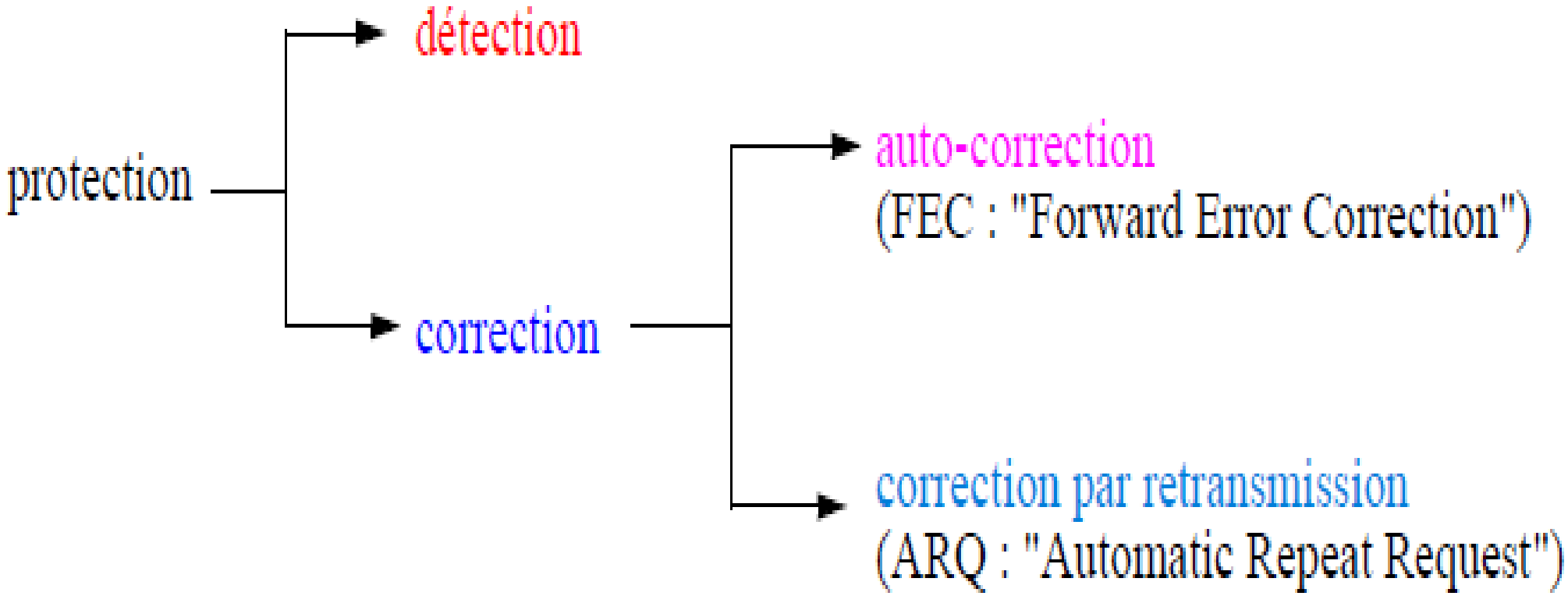


Encapsulation de paquets dans des trames.

Problématique?

Lors des échanges de données, les canaux de transmission imparfaits entraînent souvent des erreurs. Les ***erreurs*** de transmission sont due aux ***perturbations électromagnétiques*** (le démarrage de moteurs, le passage prés d'un circuit à courant fort, les orages, …etc.), au ***câblage mal isolé*** et aux effets de ***distorsion***. Le taux d'erreur sur un canal de transmission est égale au ***nombre de bits erronés / le nombre de bits émis***. Il est de l'ordre de $\mathbf{10^{-9}}$ pour les réseaux locaux filaires ; ce taux est nettement plus élevé pour les réseaux sans fil.

Il existe deux stratégies possibles pour détecter les erreurs, la destination peut :



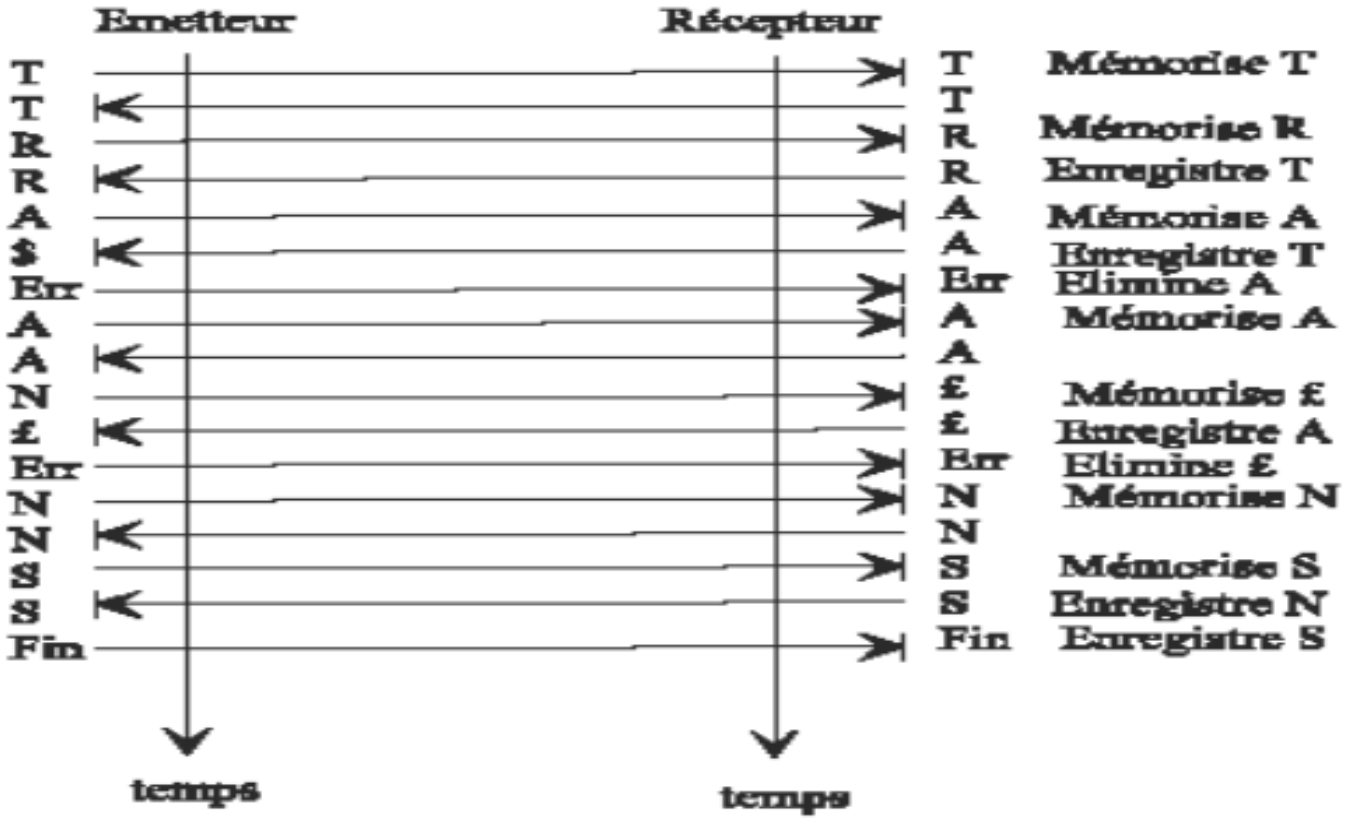
III.2. Détection et correction d'erreurs

III. 2.1 Méthodes simples de détection des erreurs

a. Vérification par répétition

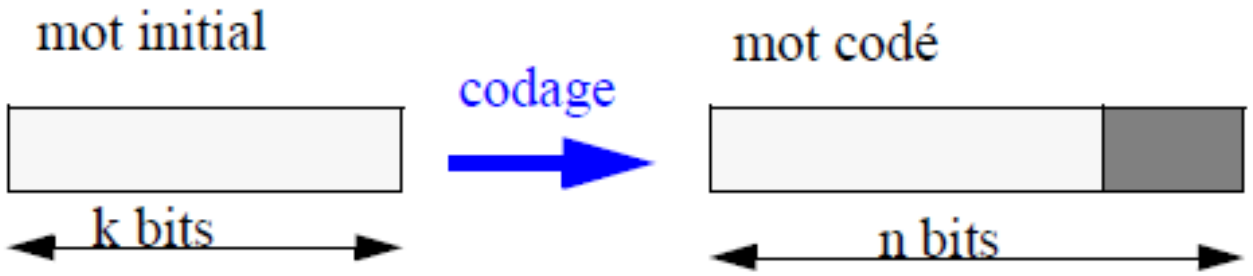
- Elle consiste à faire suivre le message d'une réplique du message d'origine, à la réception.
- Le récepteur compare les deux copies, et il est accepté que s'il y a identité. Dans le cas contraire, il est demandé à l'émetteur de retransmettre le message.
- On peut améliorer cette méthode en procédant à la correction, en envoyant un nombre impair de copies au récepteur, le récepteur détermine le nombre majoritaire de copies identiques et les considère comme étant le message d'origine ou au pire des cas renvoie une copie à l'émetteur pour vérification.

b. Echoplex : Cette méthode permet théoriquement un taux d'erreur nul. Lors d'une communication, les caractères transmis d'une machine vers une autre sont retransmis à l'arrivée vers la machine émettrice. De ce fait, si une erreur se produit dans la transmission ou la retransmission le caractère en erreur est automatiquement détecté et retransmis



III.2.2 Méthodes de correction par acquittement

- Dans ces méthodes, chaque trame à transmettre est augmentée par une autre suite de bits dite *bits de redondance* ou *somme de contrôle (checksum)* **FCS** ou encore *total de contrôle*. Ainsi, pour chaque suite de *k* bits transmise, on ajoute *r* bits.
- On dit alors que l'on utilise un code *C(n,k)* avec *n=k+r* , où *n* est la taille du code et *k* la taille de *l'information utile*.
- À la réception, les bits ajoutés permettent d'effectuer des contrôles à l'arrivée. Dans le cas d'une réception sans anomalie, il suffira d'extraire l'information utile.



III.2.2 Méthodes de correction par acquittement

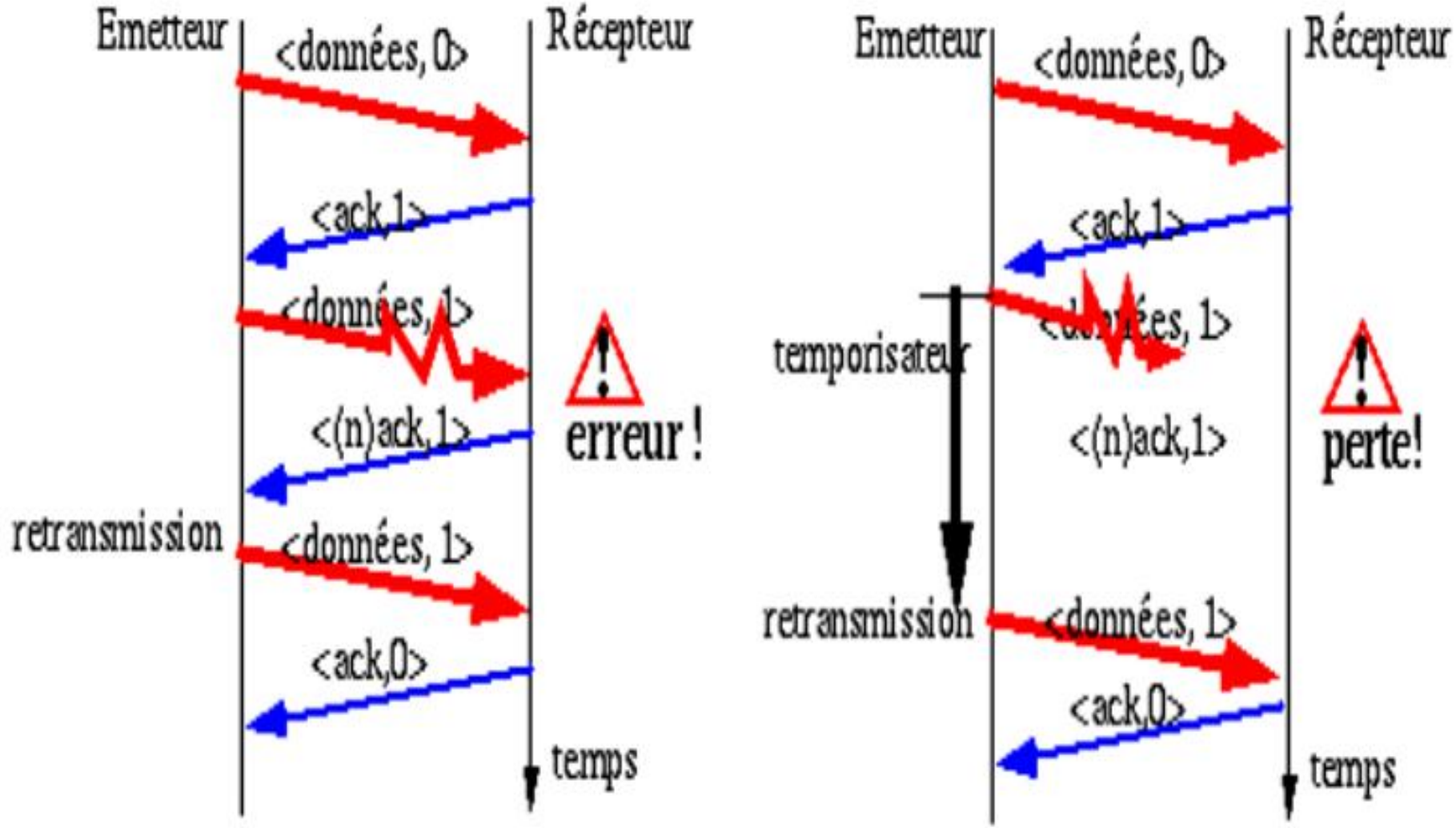
Les concepteurs de réseau ont développé deux stratégies dans le domaine de correction des erreurs de transmission :

- La première consiste à inclure dans les blocs de données transmis ***suffisamment de redondance*** pour que le récepteur soit capable de restituer les données originales à partir des données reçues. Cette stratégie utilise des *codes correcteurs d'erreur* (*Error Correcting Codes, ECC*).

- La seconde consiste à ajouter *juste* ***assez de redondance*** dans les données à transmettre pour que récepteur puisse **détecter les erreurs** et demander alors la **retransmission des trames erronées**. Cette stratégie utilise des *codes détecteurs d'erreur (Error Detecting Codes, EDC)*.

- La **correction par retransmission** est préférée dans ***les réseaux où le taux de perte est faible*** et le délai de retransmission tolérable, car son surcoût est généralement plus faible que celui induit par les codes auto-correcteurs.

- Sur les ***canaux fiables***, comme la **FO**, on utilise un *CDE* d'erreurs et on retransmet un éventuel bloc de données défectueux. Sur des réseaux sans fil où les erreurs sont courantes, on utilise suffisamment de redondance pour permettre au récepteur de trouver le bloc d'origine.
- Dans les deux stratégies, un message d'acquittement appelé « *ACK* » est systématiquement envoyé à la réception d'un bloc ou de plusieurs blocs (*acquittement groupé*). L'ACK est dit *positif* si le ou les blocs ne comportent pas d'erreurs et *négatif* sinon ; dans ce cas l'émetteur réémet les blocs erronés.
- Pour le cas où l'ACK ou le message est perdu, on implémente *un temporisateur* qui s'enclenche dès que le message est envoyé : si après « *tmax* » secondes l'ACK n'a pas été reçu, on conclut que soit le message ou l'ACK a été perdu. On procède par conséquent au renvoi du message. Toute la difficulté réside en l'estimation du temps maximal d'un allée-retour *RTT* (*Round trip Time*) déterminant la valeur « *tmax* ».



Acquittement de trames

III.3. Généralités sur les codes

III.3.1 Principe général

- Chaque suite de bits à transmettre est augmentée par une autre suite de bits dite de **redondance** ou de **contrôle**.

- Pour chaque suite de **k** bits transmise, on ajoute **r** bits. On dit alors que l'on utilise un code **C(n,k)** avec **n = k +r**.

n: taille du code;

k: taille de l'information utile.

- À la réception, les bits ajoutés permettent d'effectuer des contrôles à l'arrivée.

  Dans le cas d'une réception sans anomalie, il suffira d'extraire l'information utile.

III.3. 2 Définitions Générales

- Rappel sur les opérations binaires

Somme modulo 2	$0 \oplus 0=0$	Multiplication	0.0=0
	$0 \oplus 1=1$		0.1=0
	$1 \oplus 0=1$		1.0=0
	$1 \oplus 1=0$		1.1=1

Définition 1: On appelle un code de longueur *n*, noté *C(n,k)* telle que *n* est la taille du code et *k* est la taille de l'information, un ensemble *C* de séquences de *n bits* (mots distincts), construits sur l'ensemble {0,1}.

- Une séquence de *n* bits est dite un ***mot de code***.
- Un mot de code de *n* bits n'appartenant pas à *C* sera dit *invalide*.
- Un code *C(n k)* contient 2^k mots de codes valides.

Exemple : Soit un code *C(4,2)* : { 00**10** 10**00** 01**11** 11**10** };

la séquence 10**01** définit un mot de code invalide

A la réception d'une séquence de *n bits,* deux cas sont possibles :

- La séquence correspond à un mot du code et la transmission sera considérée comme étant correcte.
- La séquence n'est pas valide. Dans ce cas, on est en présence d'une erreur *(c'est une certitude),* et le récepteur peut alors soit corriger l'erreur *(CCD)*, soit demander une retransmission *(CDE)*.

Example : Soit le code *C(4,2)* : { 00**10** 10**00** 01**11** 11**10** }.

- A l'émission : On veut transmettre 00**,** on récupère son code 00**10.**
- Erreur de transmission : 1er cas 00**10** 00**0**0 **(erreur simple)**

2ième cas 00**10** 11**10 (erreur double)**

- A la réception, on vérifie si le mot est valide (si appartient au code)

1er cas : erreur mot non valide (erreur détectable)

2ième cas : Pas d'erreur mot valide (erreur non détectable)

Définition 2

On appelle:

- ***Distance de Hamming*** : C'est le nombre de bits en lesquels 2 mots d'un code diffèrent.

- ***Distance de Hamming minimal d'un code :*** On appelle *distance de Hamming minimale*, notée Dh_{min}, d'un code ***C***, le minimum des distances entre 2 mots quelconques de ce code c.a.d c'est le nombre de bits à 1 dans le résultat du XOR.

Exemple : M = 10001001 et M' = 10110001 => Dh_{min}(M, M') = 3.

$$\begin{array}{r} 10001001 \\ \underline{\ \ 10110001} \\ 00111000 \end{array}$$

- La Dh_{min} d'un code $C(n,k)$, est obtenu en comparant ses 2^k mots de code valides. Exemple:
  { 00**10** 10**00** 01**11** 11**10** } Dh_{min} =**2**.

- Dh_{min} d'un code permet d'évaluer son pouvoir détecteur d'erreurs, ainsi que son pouvoir correcteur. En effet, si la Dh_{min} entre deux mots de code est *d*, il faut *d* erreurs pour transformer un mot en un autre.

Théorème : Soit un code *C(n,k)* tel que si sa Dh_{min} est égale à :

- *2d+1* alors la *détection* est d'ordre *2d* et la *correction* est d'ordre *d*.
- *2d* alors la *détection* est d'ordre *2d – 1* et la *correction* est d'ordre *d –1*.

Exemple : {00**10** 10**00** 01**11** 11**10**} Dh_{min}=**2** détecte une erreur et corrige 0.

Considérons le code suivant :

*{0→ **0**000 ; 1→**1**111}, k = 1, n= 4*. Ce code possède 3 bits de contrôle. On a D_{min}*= 4*. La détection est d'ordre **3** et la correction est d'ordre **1**.

- On envoie: 0000 erreur de transmission simple → 1000
- A l'arrivée: 1000 on peut corriger en → 0000

  1000 (mot invalide) → (0000 ou 1111) **?**
- On envoie: 1111 erreur de transmission double → 0011

  A l'arrivée: 0011 on ne peut pas corriger en → 1111

  0011 (mot invalide) → (0000 ou 1111) **?**

Définition 3 (*Efficacité d'un code*) : L'*efficacité* d'un code est d'autant meilleure que les mots du code sont plus distincts les uns des autres. Elle dépend de la distance minimale de *Hamming* entre les différents mots de codes.

$$L'efficacité \text{ du code} = \frac{\text{nombre de messages reconnus faux}}{\text{nombre de messages faux}}$$

Plus l'efficacité est proche de 1, plus le code est performant

- **Définition 4 :** Soit *X* le bloc émis et *X'* le bloc reçu. On appelle ***vecteur d'erreur*** . Si ***E = 0***, alors il n y pas d'erreur. Si ***E ≠ 0,*** il y a une erreur, localisé sur les bits 1 de *E*.

$$E = X \oplus X'$$

III.4. Codes de blocs

L'information transmise est découpée en *blocs* de *k* bits, puis on leur rajoute *r* bits de redondance. On crée alors un *code de bloc* de longueur $n=k+r$. Sur 2^n combinaisons possibles, seules 2^k combinaisons sont valides. Les bits de redondance sont calculés de différentes méthodes.

III.4.1 Méthode basée sur la parité

a. Parité transversale (ou verticale) VRC

- L'information est sectionnée en blocs de *k* bits qui sont généralement des caractères, puis on ajoute à chaque bloc un bit de parité *(r=1)* de telle sorte que la somme des *k*+1 bits modulo 2 soit 0 (*parité paire*) ou égale à 1 (*parité impaire)*.

Exemple : Envoi d'un bloc de 4 caractères de longueur 3 (*k*=3) :

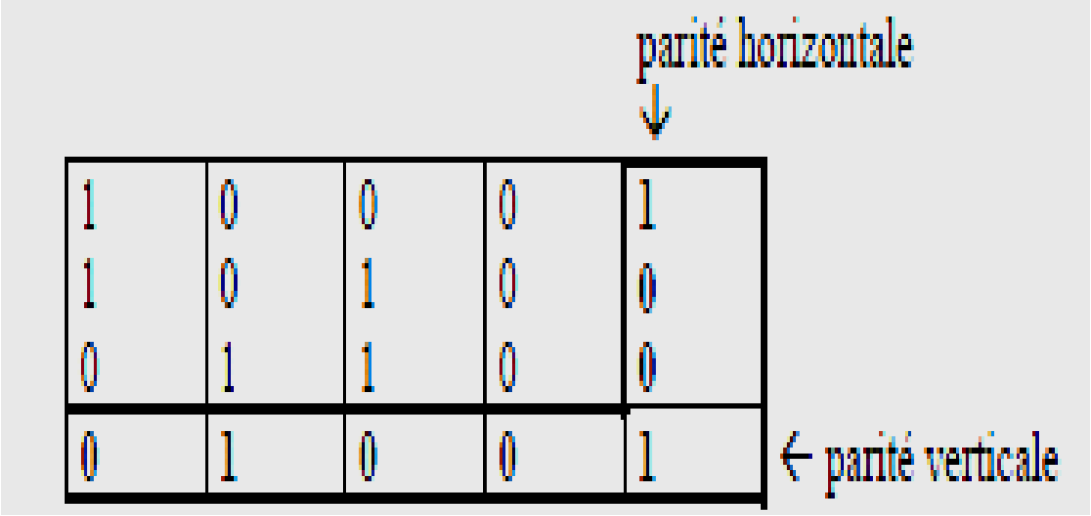
- Information utile : 110 001 011 000.
- Information envoyée : 110**0** 001**1** 011**0** 000**0**.

VRC permet de détecter une erreur simple sur chacun des mots transmis.

b. Parité longitudinale (ou horizontale) LRC/VRC*:* On combine généralement la parité transversale et la parité longitudinale de la façon suivante : les caractères munis de leur bit de parité transversale sont regroupés en blocs, et on ajoute à la fin de chaque bloc un caractère supplémentaire pour la parité longitudinale. Ce contrôle est appelé *Vertical Redundancy Checking / Longitudinal Redundancy Checking, LRC/VRC.*

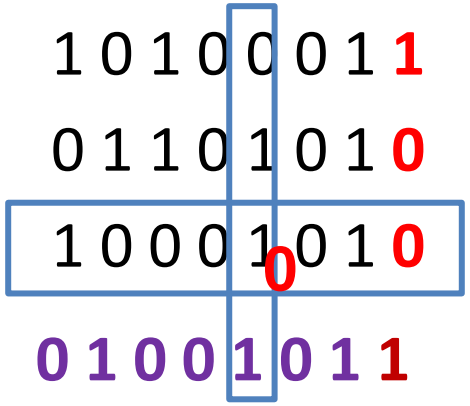
Exemple : Envoi d'un bloc de 4 caractères avec contrôle LRC/VRC :

- Information utile : 110 001 011 000.
- Information envoyée : 1100 0011 0110 0000 **1001.**



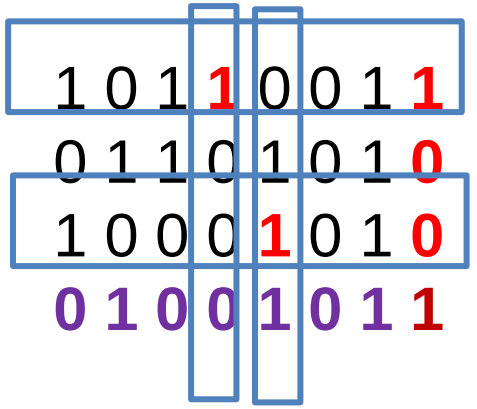
- Comme une erreur simple modifie simultanément la parité d'une ligne et d'une colonne, la *correction* est possible en inversant le bit situé à l'intersection de la ligne et de la colonne ayant une parité incorrecte.

Exemple



On peut corriger

(Erreur simple)



Impossible de corriger

(Erreur double)

III.4.2 Codes polynomiaux

Ce sont des codes de blocs très utilisés dans la pratique car facilement implantables et qui donnent d'excellents résultats. On considère que les bits d'une séquence sont les coefficients d'un polynôme. Ces coefficients ne prennent que les valeurs 0 ou 1. Un bloc de *n* bits est vu comme la série de coefficients d'un polynôme de *n termes*, allant de x^{n-1} à x^0. Un tel polynôme est dit de degré *n-1*. Le bit le plus à gauche (fort) est le coefficient de x^{n-1}, son voisin est le coefficient de x^{n-2}, ainsi de suite:

$$P(x) = p_{n-1}\, x^{n-1} + p_{n-2}\, x^{n-2} + \ldots + p_1\, x + p_0\, x^0 \qquad (p_i \in \{0,1\})$$

Par exemple, la séquence "**001101**" comprend 6 bits. Elle peut être représentée par un polynôme à 6 termes dont les coefficients sont 0, 0, 1, 1, 0 et 1, ce qui donne le polynôme :

$$x^5.0+x^4.0+x^3.1+x^2.1+x.0+x^0.1= \mathbf{x^3+x^2+1}$$

Remarque : *Sur les polynômes, nous utilisons les opérations d'addition et de multiplication modulo 2 (une addition ou une soustraction équivaut à un OU exclusif). 1+1=1-1 =0 ; 1-0=1+0=1.*

- Formellement, supposons un code *C(n, k)*. ***L'information utile est représentée par le polynôme*** *Z(x)*.

  Z(x) est au maximum de degré *(k-1)* puisque l'information comporte *k* bits :

  $$Z(x) = u_{k-1} x^{k-1} + \ldots + u_1 x + u_0 x^0. \qquad (u_i \in \{0,1\})$$

- Pour utiliser un code polynomial, l'émetteur et le récepteur doivent d'abord se mettre d'accord sur le choix d'un *polynôme générateur*. Pour des raisons pratiques, les coefficients de poids fort et faible du générateur doivent être égaux à 1. Soit *G(x)* un polynôme de degré *r=n-k* appelé polynôme *générateur.*

  $$G(x) = g_r x^r + \ldots + g_1 x + g_0 x^0. \qquad (g_i \in \{0,1\})\ g_{r-1} = g_0 = 1$$

Codage à l'émission

- Il faut ***multiplier*** $Z(x)$ par x^r pour créer un décalage à gauche de r bits.

Exemple : Soit un code $C(11,7)$. Nous considérons, $Z(x)$ 1011011, on multiplie par x^4 et on obtient : 10110110000.

- On ***divise*** le ***produit obtenu*** par ***G(x)*** ; on obtient:

$Z(x)\, x^r = Q(x)\, G(x) + A(x)$ Où :

$Q(x)$: polynôme quotient.

$A(x)$: polynôme reste de la division, au maximum de degré : $r-1$.

- Donc on ***obtient le mot de code à envoyer***, représenté par le polynôme $Y(x)$ de degré $n-1$ suivant :

$$Y(x) = Z(x)\, x^r + A(x) = Q(x)\, G(x)$$

- On envoie la séquence de bits de longueur $n=k+r$ associée au polynôme $Y(x)$. La séquence envoyée est construite en collant (rajoutant) à l'information utile, le total de contrôle représenté par $A(x)$. Le polynôme $Y(x)$ obtenu est aussi divisible par $G(x)$. Il en résulte que les mots valides du code polynomial $C(n,k)$ sont donc les polynômes multiples de $G(x)$.

Exemple

Soit le code *C(9,6)* avec ***G(x)= x^3+1.***

- On veut transférer l'information "001101".

Z(x) = x^3+x^2+1.

$Z(x).x^3 = x^6+x^5+x^3$

$$\begin{array}{r|l} X^6+X^5+X^3 & X^3+1 \\ \hline -(X^6+X^3) & X^3+X^2 \\ X^5 & \\ -(X^5+X^2) & \\ X^2 & \end{array}$$

- On divise $Z(x).x^3$ par *G(x)*, on obtient :

  $Z(x) \cdot x^3 = G(x) \cdot Q(x) + A(x)$, avec ***$A(x)=x^2$*** et $Q(x)=x^3+x^2$.

- D'où : $Y(x) = Z(x) \cdot x^3 + A(x) = x^6+x^5+x^3+x^2 =$ ***$x^6+x^5+x^3+x^2$.***

Le polynôme *Y(x)* correspond à la séquence : 001101***100***

A(x) doit être écrit sur *r bits*. Si A(x)=x on l'écrit 010 et non 10.

Décodage à la réception

- A 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 transmission.

- Soit *Ÿ(x)* le polynôme de degré *n-1* dénotant le ***mot de code reçu***.

$$S(x) = \ddot{Y}(x) \ MOD[\ G(x)]$$

- Aucune erreur n'est détectée si le syndrome $S(x) = 0$. Sinon elle est détectée si $S(x) \neq 0$. La détection d'erreur consiste à vérifier que le mot reçu est bien un mot du code $C(n,k)$ c'est-à-dire que $\ddot{Y}(x)$, est divisible par $G(x)$.
- Si le polynôme est divisible par $G(x)$, alors il suffit alors d'extraire l'information utile en supprimant les r derniers bits de la séquence reçue. Si le polynôme n'est pas divisible par $G(x)$, alors une erreur a eu lieu pendant la transmission et le récepteur demandera une retransmission du message.

Exemple : On envoie la séquence de bits"001101**100**". On remarque bien que cette séquence est construite en concaténant à l'information utile "001101", la séquence "**100**" qui correspond à la redondance.

Cas 1 : On reçoit "**0**11101100" (Noter l'erreur de transmission sur le deuxième bit).

$\ddot{Y}(x) = x^7 + x^6 + x^5 + x^3 + x^2.$

On calcule $S(x)$:

$$\begin{array}{r|l} X^7+X^6+X^5+X^3+X^2 & X^3+1 \\ \hline -(X^7+X^4) & X^4+X^3+X^2+X \\ X^6+X^5+X^4+X^3+X^2 & \\ -(X^6+X^3) & \\ X^5+X^4+X^2 & \\ -(X^5+X^2) & \\ X^4 & \\ -(X^4+X) & \\ X & \end{array}$$

Le reste étant non nul, une erreur de transmission est détectée.

Cas 2 : On reçoit "001101100" (Noter l'absence d'erreur de transmission).

$\ddot{Y}(x) = x^6 + x^5 + x^3 + x^2.$

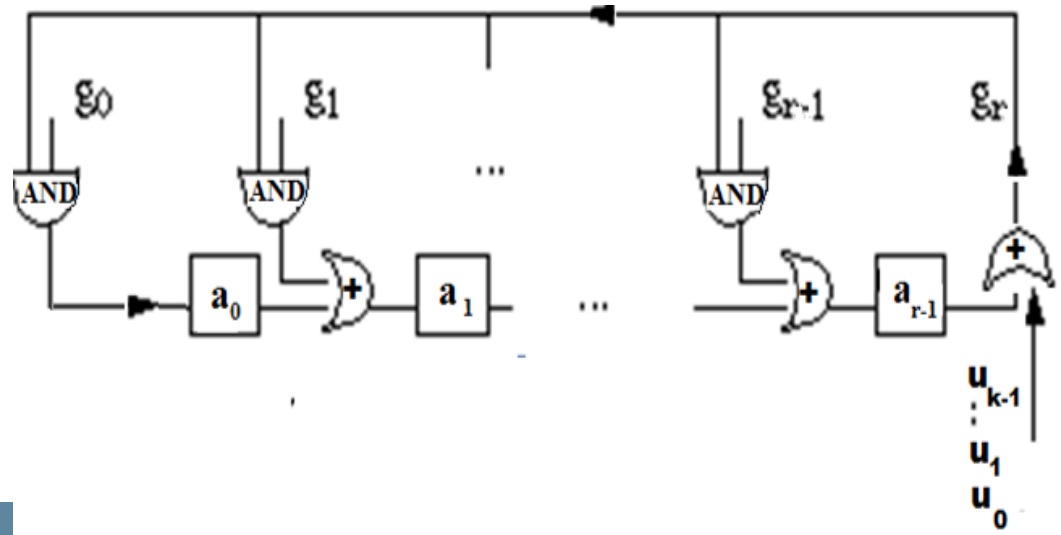
On calcule $S(x)$:

$$\begin{array}{r|l} X^6+X^5+X^3+X^2 & X^3+1 \\ \hline -(X^6+X^3) & X^3+X^2 \\ X^5+X^2 & \\ -(X^5+X^2) & \\ 0 & \end{array}$$

Aucune erreur de transmission n'est détectée et l'information utile est la séquence de bits obtenue en supprimant les 3 derniers bits (r=3) de la séquence "001101**100**".

Circuit logique d'un codeur polynomial

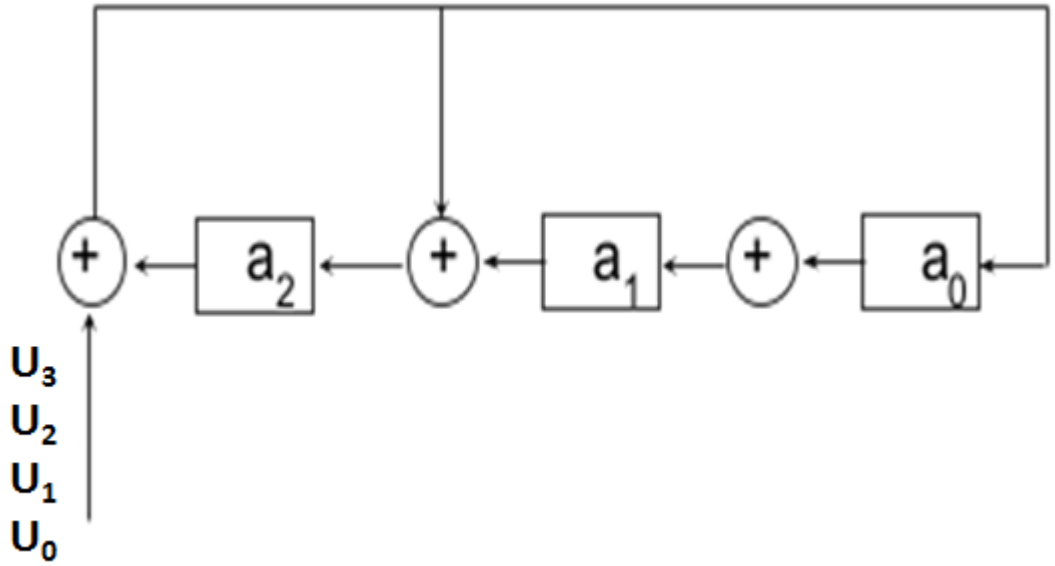
- La division se fait à l'aide d'un **circuit logique** appelé ***diviseur bâti*** autour d'un registre à décalage. Le registre est constitué de ***r*** bascules a_{r-1}, …, a_0, représentant les ***bits de contrôle***, liés par des opérateurs de ***OU exclusif*** $\oplus$. Les coefficients du polynôme *Z(x)* sont injectés dans le circuit un à un, à chaque cycle d'horloge, commençant du coefficient le plus fort au plus faible.
- Initialement, les registres a_i sont à zéro. Chaque coefficient en entrée est sommé avec le bit a_{r-1}. La sortie de cet opérateur va en entrée vers a_0. Si dans le polynôme *G(x)*, le coefficient de x^i est égal à 1, une branche de la sortie $x_i \oplus a_{r-1}$ est créée en entrée de l'opérateur *OU exclusif* mis avant la bascule du bit a_{i-1}. Après le passage des *k* bits de *Z(x)* en *k* cycle horloge, le calcul est achevé et les registres a_i contiennent les coefficients du polynôme *A(x)*.



Exemple

Soit un code C(7,4) avec G(x) = x3 + x2 +1

u_i	u_i+a_2	a_0	a_1	a_2
		0	0	0
1	1	1	0	1
1	0	0	1	0
0	0	0	0	1
1	0	0	0	0



Propriétés des codes polynomiaux

Soit Y(x) un mot envoyé, et Ÿ(x) le mot reçu correspondant tel que Ÿ(x)=Y(x)+E(x).
On a alors les propriétés suivantes :

- Toute erreur simple est détectée si G(x) comporte plus d'un coefficient non nul (C'est pour cela que g0=gr=1).
- Les erreurs doubles sont toutes détectées si G(x) ne divise pas xi+1 où i appartient à {r, n-1}, n étant la taille du code.
- L'erreur sur un message comportant un nombre impair d'erreurs est toujours détectée si le polynôme générateur G(x) est divisible par (x+1).
- Un code polynomial détecte toutes les salves (suite d'erreurs), de longueur inférieure ou égale à r avec r le degré de G(x).

Polynômes générateurs utilisés : Le choix du polynôme générateur est très important : de lui dépendra le pouvoir de détection de certains types d'erreur de transmission. Les principaux polynômes utilisés :

- 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 : $x^{16} + x^{11} + x^4 +1$
- CRC-32= $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x +1$.

Nom	Générateur	Factorisation	Exemples d'utilisation
TCH/FS -HS-EFS	$X^3 + X + 1$	irréductible	GSM transmission de voix
GSM TCH/EFS	$X^8 + X^4 + X^3 + X^2 + 1$	irréductible	GSM pré-codage canal à plein taux
CRC-8	$X^8 + X^7 + X^4 + X^3 + X + 1$	$(X + 1)(X^7 + X^3 + 1)$	GSM 3ème génération
CRC-16 X25-CCITT	$X^{16} + X^{12} + X^5 + 1$	$(X + 1)(X^{15} + X^{14} + X^{13} + X^{12} + X^4 + X^3 + X^2 + X + 1)$	Protocole X25-CCITT ; contrôle trames PPP FCS-16 (RFC-1662)
CRC-24	$X^{24} + X^{23} + X^{18} + X^{17} + X^{14} + X^{11} + X^{10} + X^7 + X^6 + X^5 + X^4 + X^3 + X + 1$	$(X + 1)(X^{23} + X^{17} + X^{13} + X^{12} + X^{11} + X^9 + X^8 + X^7 + X^5 + X^3 + 1)$	communications UHF et satellites (SATCOM) ; messages OpenPGP (RFC-2440)
CRC-24 (3GPP)	$X^{24} + X^{23} + X^6 + X^5 + X + 1$	$(X + 1)(X^{23} + X^5 + 1)$	GSM 3ème génération
CRC-32 AUTODIN-II	$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$	irréductible	IEEE-802.3, ATM AAL5, trames PPP FCS-32 (RFC-1662) ; contrôle d'intégrité des fichiers ZIP et RAR ;

Pour corriger, on utilise des polynômes avec des propriétés particulières.

Exemple: Codes cycliques, codes **BCH** (**B**ose, Ray-**C**haudhuri et **H**ocquenghem); Codes de Reed Solomon.

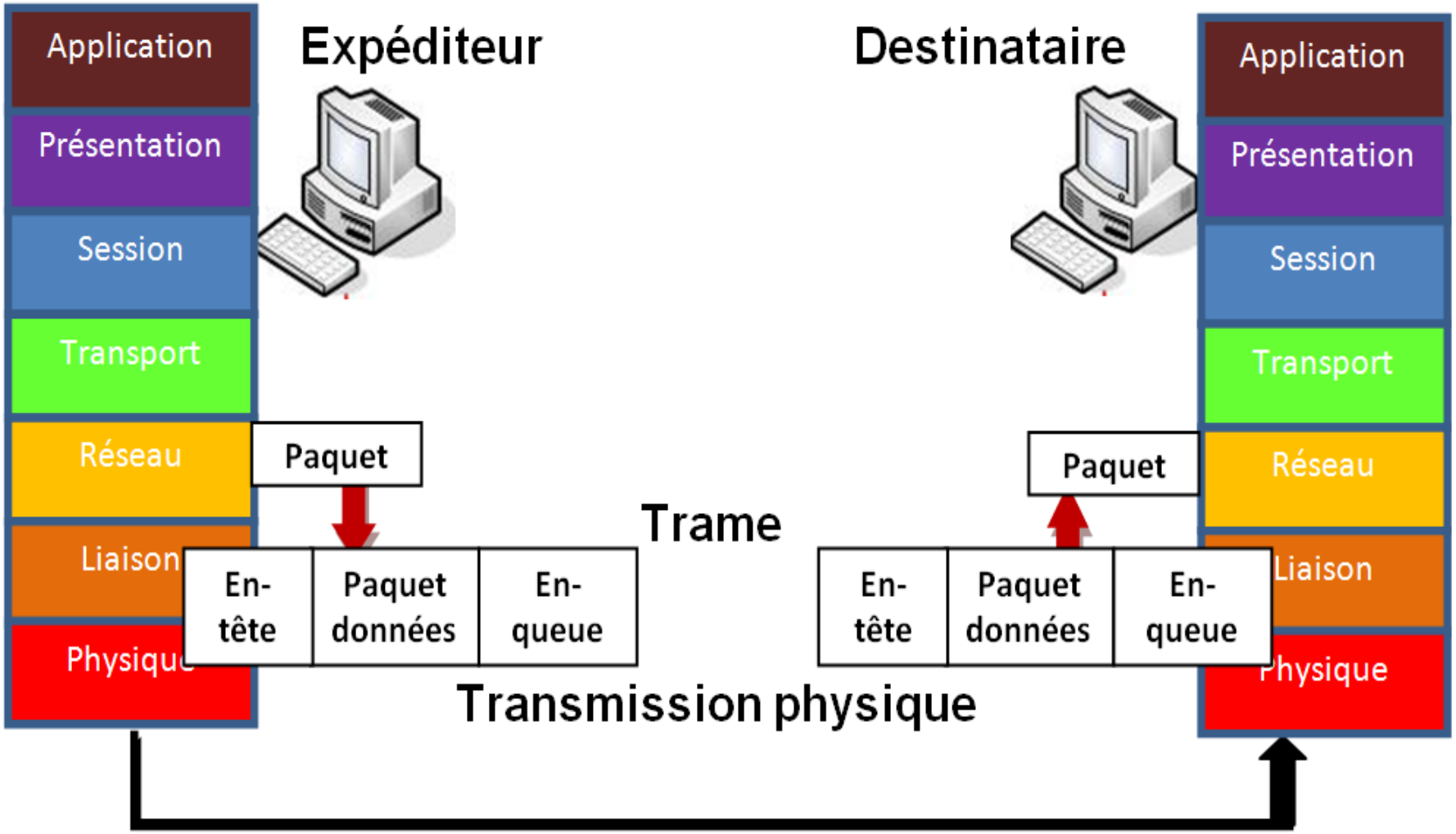
III.5. Protocoles de liaison

Les processus correspondant aux couches physiques et liaison s'exécutent sur un processeur situé sur une carte d'*E/S* (carte réseau). Le contrôle d'erreur est effectué par des circuits électroniques (hardware).

III.5.1 Principe général d'un protocole de liaison

- Une connexion de liaison de données est réalisée entre deux machines adjacentes dans le réseau, donc sans nœuds intermédiaires entre elles.
- La couche liaison reçoit de la couche réseau des données sous forme de paquets, et son rôle est de délivrer les paquets à la couche réseau voisine.
- Chaque paquet reçu, est *encapsulé* dans une trame en lui ajoutant un entête et un en-queue. L'en-queue contient des informations de contrôle destinées à la détection d'erreur. La trame est ensuite envoyée vers la machine voisine.
- Lorsque le récepteur reçoit une trame, le matériel calcule le total de contrôle. S'il est incorrect (erreur de transmission), la couche liaison en est informée avec une *erreur de checksum*; sinon elle reçoit la trame. Puis après vérification qu'elle est bien le destinataire (adresses physiques de l'entête), elle *décapsule* le paquet et le transmet à la couche réseau.

Algorithme
Début
Tant que vrai Faire
Capturer(evt);
***Cas** evt=réception_paquet_de_réseau*
récupérer_de_couche_réseau(paquet);
construire_entête(e); crc=calculer_crc(); trame;=encapsuler(paquet);
vers_couche_physique(trame);
***Cas** evt=arrivée_trame*
réception_trame récupérer_de_couche_physique(trame);
crc=calculer_crc();
Si erreur_checksum alors envoyer_ack_negatif();
Sinon
paquet=désencapsuler(trame);
envoyer_vers_couche_reseau(paquet);
envoyer_ack_positif();
Fin Cas;
Fin


Application
Présentation
Session
Transport
Réseau
Liaison
Physique
Expéditeur
Destinataire
Paquet
Trame
En-tête
Paquet données
En-queue
Transmission physique