Protocoles de communication

Dernière mise à jour le : 08/08/2023

Pour communiquer, les machines utilisent ce qu'on appelle des **protocoles de communication**.

Un **protocole de communication** est un ensemble de règles précisant :

- le format des informations échangées
- la manière de les échanger
- la manière d'établir la communication et de la terminer

En réalité, lorsque deux machines communiquent l'une avec l'autre, elles mettent en oeuvre plusieurs protocoles de communication et nous allons voir tout cela dans ce chapitre.

Rappels sur le protocole TCP/IP

Les protocoles IP et TCP sont deux protocoles de communication fondamentaux permettant à deux ordinateurs de communiquer entre eux via le réseau Internet. Ces deux protocoles sont tellement liés entre eux que l'on parle souvent de **protocole TCP/IP**, protocole basé sur le *découpage des données en paquets* et sur l'*encapsulation*.

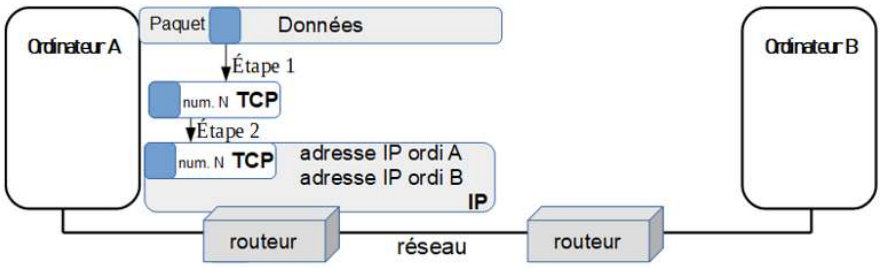
Le **protocole IP** (de l'anglais *Internet Protocol*, littéralement « protocole d'Internet ») est le protocole qui assure l'acheminement des paquets d'une machine A vers une machine B, en utilisant notamment les adresses IP de ces deux machines.

Le **protocole TCP** (de l'anglais *Transmission Control Protocol*, littéralement « protocole de contrôle de transmission ») est le protocole qui assure que la transmission des paquets entre deux machines se fait correctement (numérotation et accusés de réception).

Voici plus en détails les différentes étapes du protocole TCP/IP (qui correspond à ce qui a été vu en classe de Seconde en SNT) :

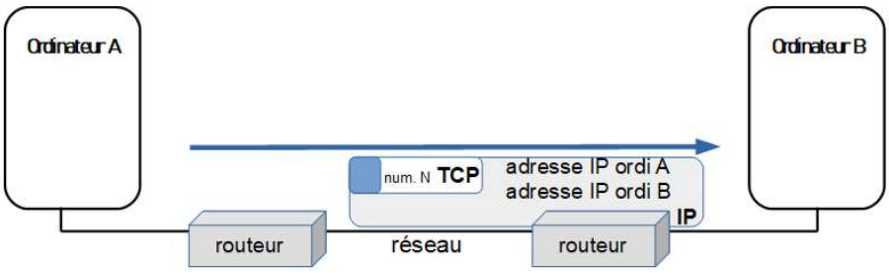
Étape 1 : Encapsulation

TCP « découpe » les données à transmettre en paquets. Il « encapsule » chaque paquet en ajoutant un numéro au paquet. C'est ensuite IP qui « encapsule » le paquet TCP en ajoutant les adresses IP de l'émetteur (A) et du récepteur (B) pour permettre le transport du paquet.



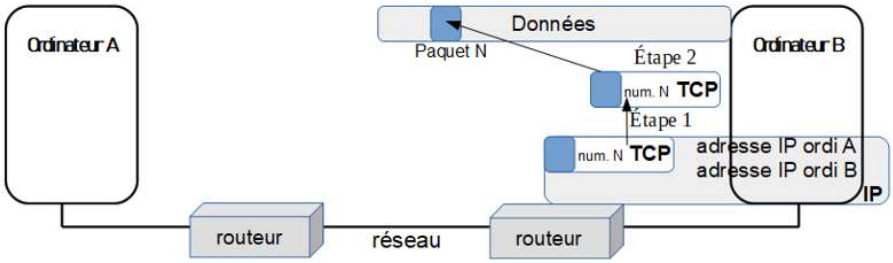
Étape 2 : Transport

Le protocole IP s'occupe ensuite de faire arriver à destination les paquets en utilisant l'adresse IP de l'ordinateur de destination. Pour cela, le paquet passe de routeurs en routeurs par le chemin le plus rapide (qui est déterminé par des algorithmes).



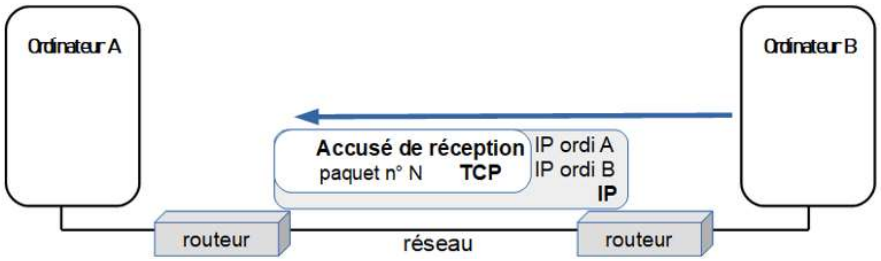
Étape 3 : Désencapsulation

Une fois arrivés à destination (ordi B), les données sont "désencapsulées" : IP donne chaque paquet à TCP qui se charge de les réordonner en fonction de leur numéro pour reconstituer les données.



Étape 4 : Envoi de l'accusé de réception

Le protocole TCP permet de s'assurer qu'un paquet est bien arrivé à destination. En effet, quand l'ordinateur B reçoit un paquet de données en provenance de l'ordinateur A, l'ordinateur B envoie un accusé de réception à l'ordinateur A ("OK, j'ai bien reçu le paquet").



Une fois tous les paquets arrivés à destination, le fichier d'origine pourra être reconstitué. Si un des paquets n'arrive pas à destination, ou si l'ordinateur A ne reçoit pas cet accusé de réception en provenance de B, le fichier ne pourra pas être reconstitué, le paquet "perdu" devra être renvoyé par l'émetteur (ordinateur A), via le protocole TCP/IP au bout d'un temps prédéfini.

En réalité, ce qui a été vu en Seconde et rappelé au-dessus est une version très simplifiée des échanges entre deux machines. Dans la suite, nous allons être plus précis et plus complet.

Le modèle de couches *Internet*

Nous avons donc vu que les bits transmis entre deux ordinateurs contiennent, en plus des données utiles (le code HTML d'une page web par exemple), d'autres données importantes : le numéro du paquet (ajouté par TCP), les adresses IP de l'émetteur et du destinataire (ajoutées par IP), etc.

Mais ce n'est pas tout, car d'autres données tout aussi importantes sont ajoutées par des protocoles que nous n'avons pas encore évoqués.

En effet, les communications entre deux machines s'appuient en réalité sur une *pile de protocoles*, autrement dit une succession de plusieurs couches de protocoles. Le terme « pile » implique que chaque couche de protocole s'appuie sur celles qui sont au-dessus afin d'y apporter un supplément de fonctionnalité.

Ces couches de protocoles correspondent à ce qu'on appelle un **modèle** et l'un des modèles les plus connu s'appelle le **modèle Internet** (aussi appelé *modèle TCP/IP*, mais ce nom porte à confusion, voir point info plus bas), organisé en 4

couches :

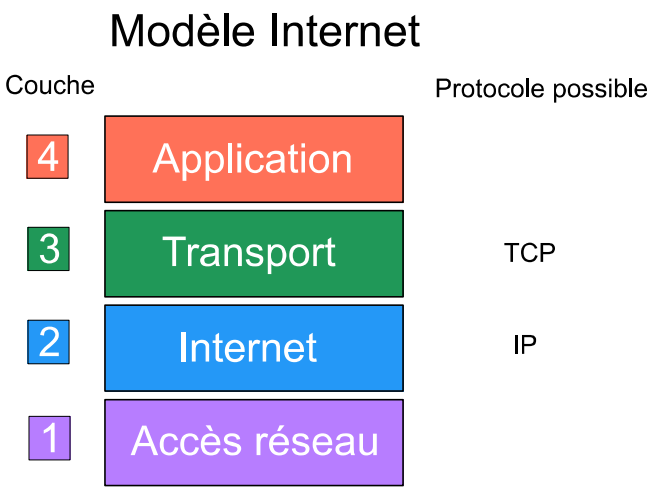


Fig. 1 - Modèle Internet en 4 couches.

Dans ce modèle :

- la couche **application** est le point d'accès aux services réseau pour les utilisateurs : elle met en oeuvre des protocoles permettant de coder et décoder les données utiles pour les applications (par exemple, le codage d'un email à envoyer)
- la couche **transport** gère les connexions entre les applications : elle met en oeuvre des protocoles chargés du contrôle des flux réseaux et de la gestion des erreurs (synchronisation des deux machines, numérotation des paquets, accusés de réception)
- la couche **internet** met en oeuvre les protocoles permettant d'acheminer les paquets jusqu'à destination via différents routeurs à travers le réseau
- la couche **accès réseau** met en oeuvre les protocoles permettant de transférer physiquement les données entre deux machines d'un même réseau (via Ethernet, Wi-Fi, fibre, etc.)

Les protocoles TCP et IP appartiennent respectivement aux couches *transport* (n°3) et *internet* (n°2) du modèle Internet. Par rapport à ce qu'on a vu, on voit qu'il existe deux couches supplémentaires au processus d'envoi d'un message via Internet : une tout au début, la couche *application* (n°4), et une tout à la fin, la couche *accès réseau* (n°1).

Pour ces deux "nouvelles" couches, le principe est toujours similaire : elles ajoutent des données à celles de la couche supérieure. Ainsi, le schéma suivant résume ce principe d'encapsulation des données à chaque couche :

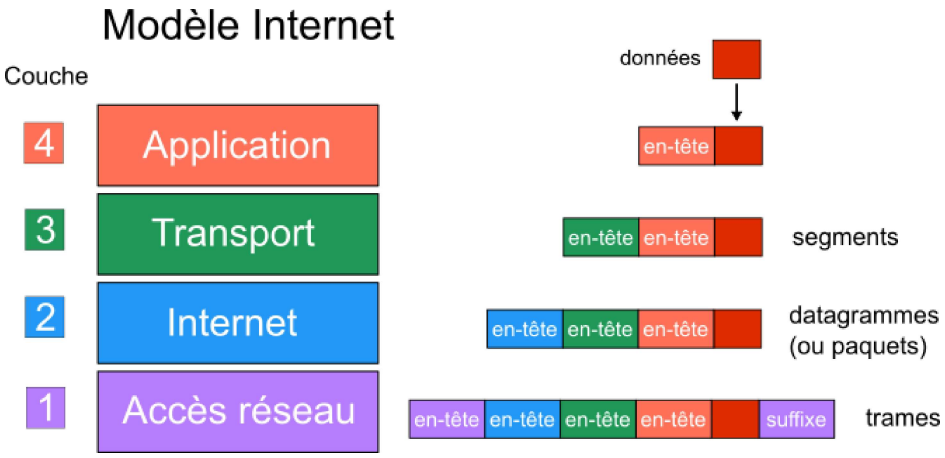


Fig. 2 - Encapsulation des données.

La suite de bits de la couche "Transport" s'appelle un *segment*, celle de la couche "Internet" s'appelle un *datagramme* ou *paquet* et celle de la couche "Accès réseau" s'appelle une *trame*.

Les couches "Transport" et "Internet" ayant déjà été décrites avec TCP et IP, il nous reste à détailler le fonctionnement des deux autres.

Le nom *modèle TCP/IP* peut porter à confusion car, bien que TCP et IP soient les deux protocoles fondamentaux des couches "Transport" et "Internet", ce ne sont pas les seuls possibles (c'est pourquoi on lui préférera le nom *modèle Internet*).
Par exemple, le protocole UDP est également très utilisé par la couche "Transport" (pour en savoir plus : https://fr.wikipedia.org/wiki/User_Datagram_Protocol). De même, la couche "Internet" met en oeuvre d'autres protocoles importants, comme ARP que vous verrons par la suite.

Couche *Accès réseau*

Le datagramme IP de la couche 2 n'est pas envoyé directement, il est lui même encapsulé dans ce qu'on appelle une **trame** qui sera ensuite envoyée *physiquement* sur le réseau.

En effet, la communication entre deux machines ne peut s'effectuer qu'à travers une interface physique identifiée par ce qu'on appelle une **adresse MAC** (abrégée @MAC dans la suite), aussi appelée *adresse physique* (MAC signifie *Media Access Control* soit "Contrôle d'accès au support").

Cette adresse MAC est définie de manière unique par le constructeur de la machine, il n'existe pas deux adresses MAC différentes. Elle est codée sur 6 octets : XX:XX:XX:XX:XX:XX en hexadécimal. Exemple d'adresse MAC : 5E:FF:56:A2:AF:15.

Les interfaces physiques les plus connues sont "Ethernet", "Wi-Fi" et "OTN" (pour *Optical Transport Network*, soit "réseau de transport optique", c'est-à-dire la fibre). Nous allons évoquer la trame Ethernet dans la suite, mais le principe est similaire pour les autres types de trames.

Trame Ethernet

Dans le cas d'une liaison Ethernet, la **trame** de la couche "Accès réseau" s'appelle tout simplement une **trame Ethernet**.

Comme la communication physique entre deux machines se fait en utilisant leurs @MAC, il est nécessaire de connaître ces adresses physiques. Or, le datagramme IP ne contient que les adresses IP de l'émetteur et du destinataire, et ces adresses IP sont distinctes des adresses MAC. Comment faire ?

Protocole ARP

C'est le **protocole ARP** qui permet de connaître l'adresse MAC correspondant à une adresse IP. Le principe est simple :

- l'ordinateur émetteur, qui ne connaît pour le moment que l'adresse IP du destinataire du paquet, envoie une requête (appelée *requête ARP*) à toutes les machines du réseaux : « quelle @MAC a cette @IP ? »
- les machines non concernées ne répondent pas
- celle qui reconnaît son @IP répond en donnant son @MAC

Une fois que l'ordinateur émetteur a connaissance de l'adresse MAC du destinataire, il peut encapsuler le datagramme IP en ajoutant les adresses MAC de l'émetteur et du destinataire, pour obtenir ce qu'on appelle la **trame Ethernet**. Cette dernière pourra alors être envoyée sur le réseau jusqu'au destinataire.

La requête ARP n'est pas toujours nécessaire, car chaque machine garde en mémoire une table de correspondance entre @IP et @MAC. La requête ARP n'est envoyée que si l'@MAC n'est pas dans la table.

Vous avez peut-être remarqué que dans la trame Ethernet étaient ajoutées, en plus de l'en-tête, des données à la fin du paquet (suffixe) : ces informations supplémentaires correspondent à des codes de contrôle d'erreurs destinés à repérer des erreurs dans la transmission physique des octets à travers le réseau. Ce n'est pas du tout au programme, mais sachez que ces contrôles d'erreurs sont basés sur des propriétés mathématiques et constituent un bel exemple d'application des mathématiques en informatique.

Couche *Application*

Revenons maintenant tout au début. Au départ, le message encapsulé par TCP contient lui-même des données encapsulées par le protocole utilisé au niveau de la couche "Application".

En effet, l'application utilisée par un usager met en oeuvre un protocole pour encapsuler les données à transmettre. Par exemple :

- un navigateur web utilise le protocole HTTP/HTTPS pour encapsuler les données à transmettre au serveur (voir Thème 4, Chapitre 2 : Dialogue client-serveur sur le Web)
- un logiciel de messagerie utilise des protocoles comme SMTP, POP ou IMAP pour transférer des courriers électroniques vers les serveurs de messsagerie électronique
- un logiciel de transfert de fichiers vers un serveur peut utiliser le protocole FTP pour envoyer des fichiers sur un serveur
- etc.

Conclusion

Pour qu'une machine envoie un message à une autre, chaque couche encapsule les données de la couche supérieure en appliquant un protocole de communication. Comme nous l'avons vu, il existe différents protocoles pour chaque couche du *modèle Internet* :

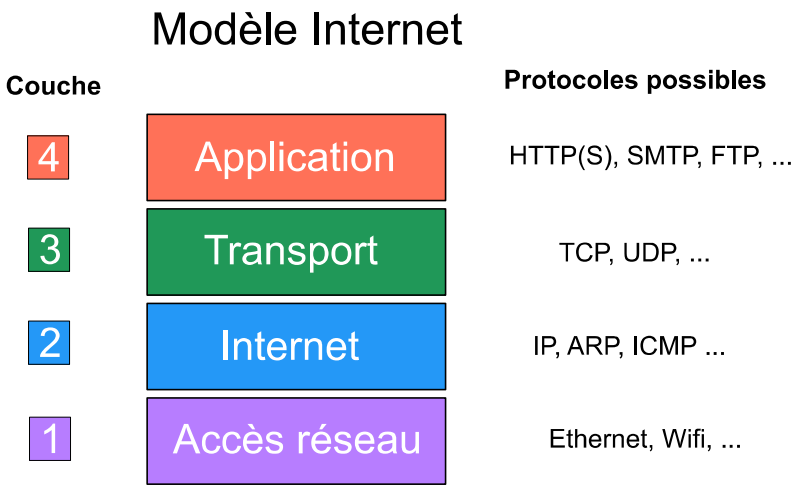


Fig. 3 - Les différents protocoles par couche.

Lors de la réception du message, l'ordinateur destinataire procède de manière inverse, selon les mêmes protocoles, en désencapsulant les données avant de les donner à la couche supérieure jusqu'à l'application censé recevoir les données.

Le schéma suivant résume les différentes encapsulations permettant d'envoyer une requête HTTP (contenant ici les données d'un formulaire) via une liaison Ethernet, selon le *modèle Internet* :

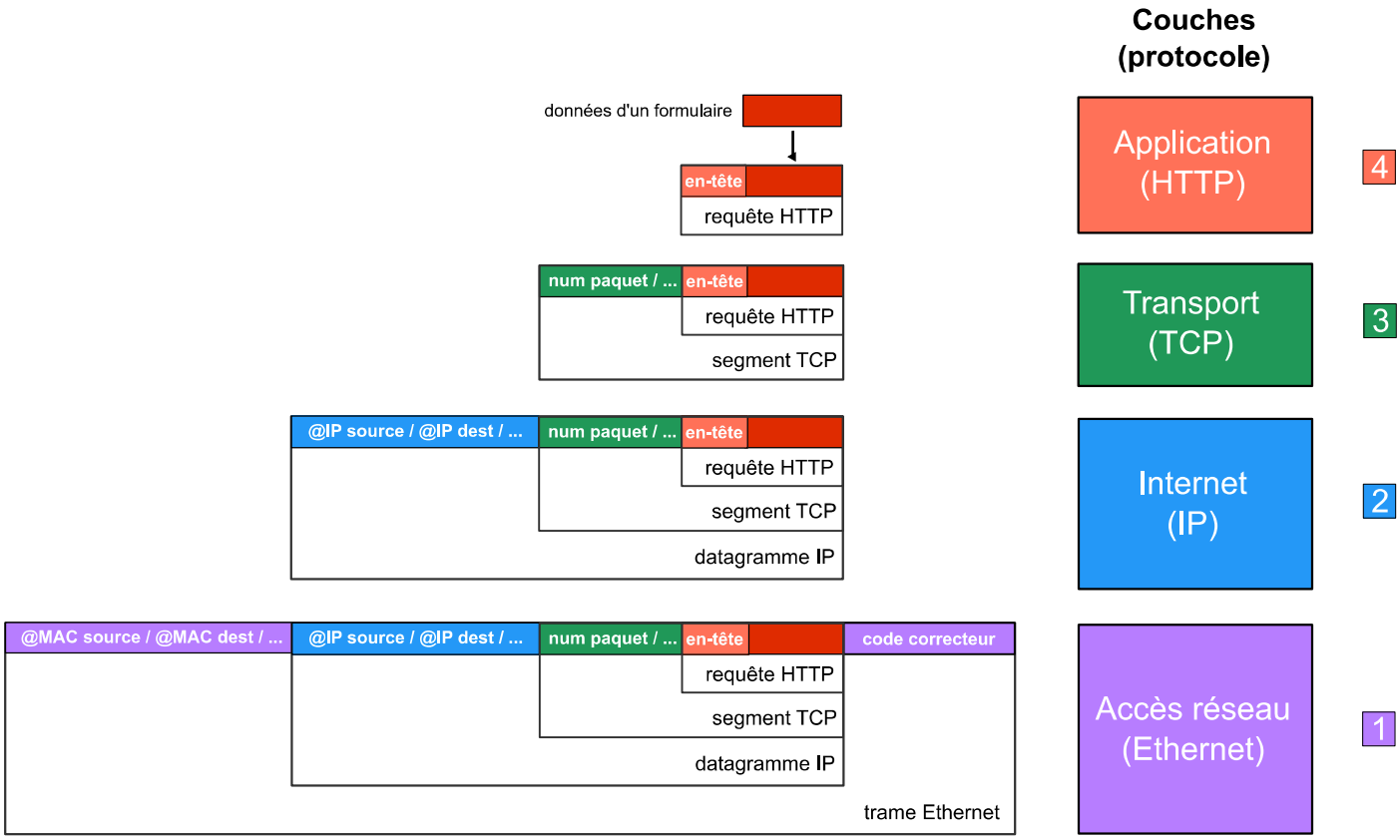


Fig. 4 - Encapsulation des données d'une requête HTTP dans une trame Ethernet.

Le modèle OSI

Le modèle Internet décrit précédemment n'est qu'un modèle théorique, qui n'est adapté que pour des communications via le réseau Internet. Il existe un autre modèle, appelé **modèle OSI** (Open Systems Interconnection, 1984) organisé en 7 couches qui permet de définir les différentes couches pour des communications via un réseau (pas nécessairement Internet).

Le schéma ci-dessous présente ce modèle OSI et sa correspondance avec le modèle Internet :

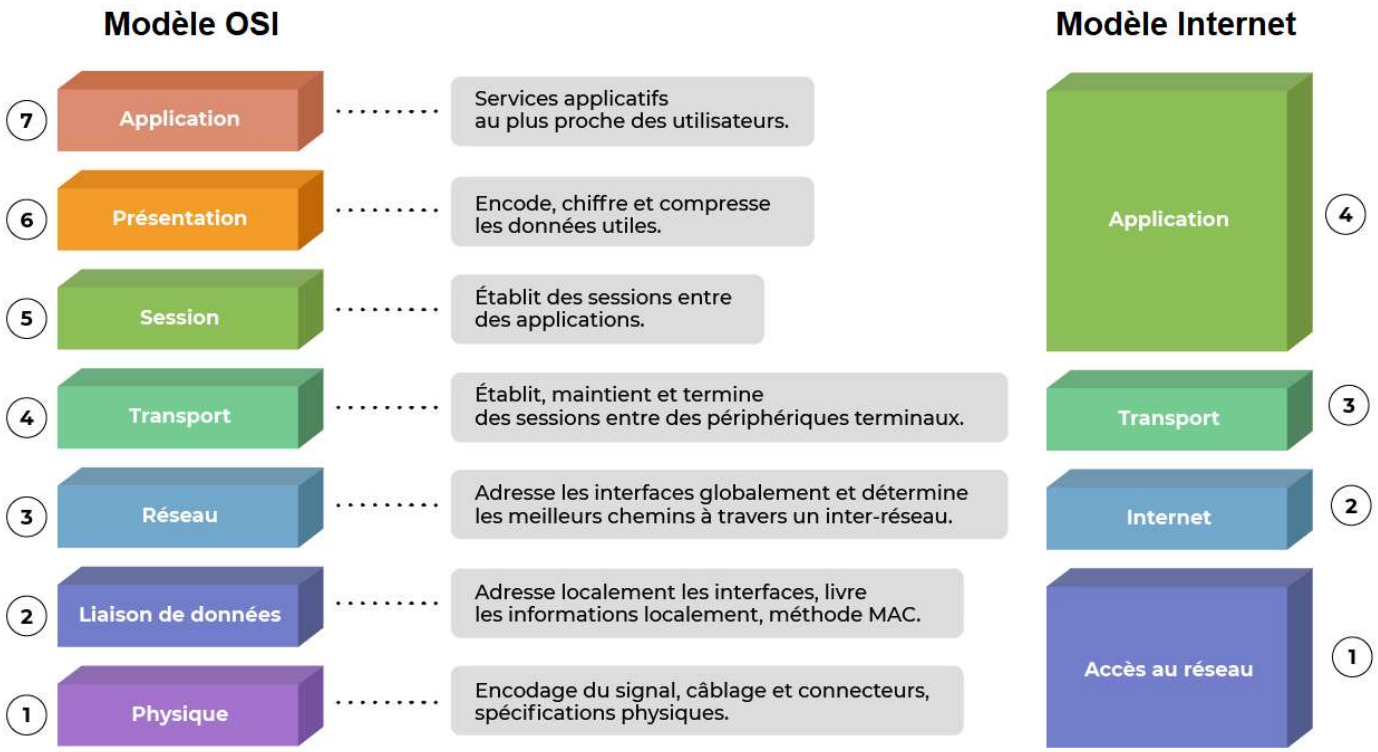


Fig. 5 - Modèle OSI et correspondance avec le modèle Internet

Crédit : Adaptation de schémas du cours Concevez votre réseau TCP/IP d'OpenClassrooms (Damien A.), diffusé sous licence CC BY-SA 4.0.

Le découpage est plus important que le modèle Internet : la couche *Application* est divisée en trois couches différentes et la couche *Accès réseau* en deux. Par ailleurs, la couche *Internet* s'appelle couche *Réseau* dans le modèle OSI car, comme on l'a dit, ce modèle concerne des communications dans n'importe quel type de réseau.

Même s'il y a plus de couches, le principe d'encapsulation perdure : chaque couche encapsule les données fournies par la couche supérieure.

Le protocole du bit alterné

Crédits : Cette dernière partie, accompagnée de superbes schémas, est reprise dans sa quasi totalité de l'excellent cours de Gilles Lassus 🙏 sur les Protocoles de communication, diffusé sous licence CC BY-SA.

La couche *Transport* gère les flux de données, c'est donc à ce niveau que des protocoles sont mis en oeuvre pour s'assurer de la fiabilité des transmissions de données.

Nous avons vu que le protocole TCP utilisait un mécanisme de numérotation pour remettre dans l'ordre les paquets à destination, et d'accusé de réception pour s'assurer qu'ils parviennent tous à destination.

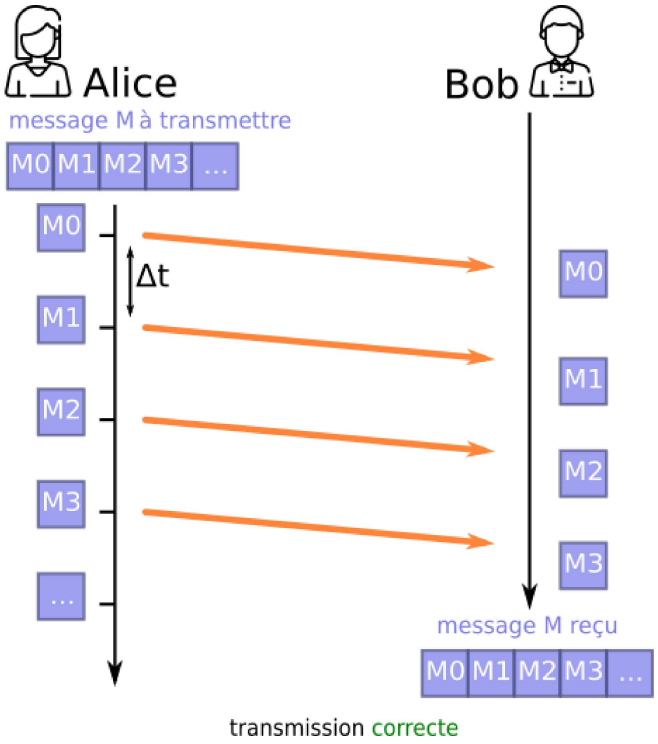
C'est un protocole très efficace mais cher en ressource. Dans ce dernier paragraphe, nous allons décrire une solution plus basique mais pas infaillible appelée **protocole du bit alterné**.

Contexte

- Alice veut envoyer à Bob un message M, qu'elle a prédécoupé en sous-messages M0, M1, M2,...
- Alice envoie ses sous-messages au fur et à mesure

Situation idéale

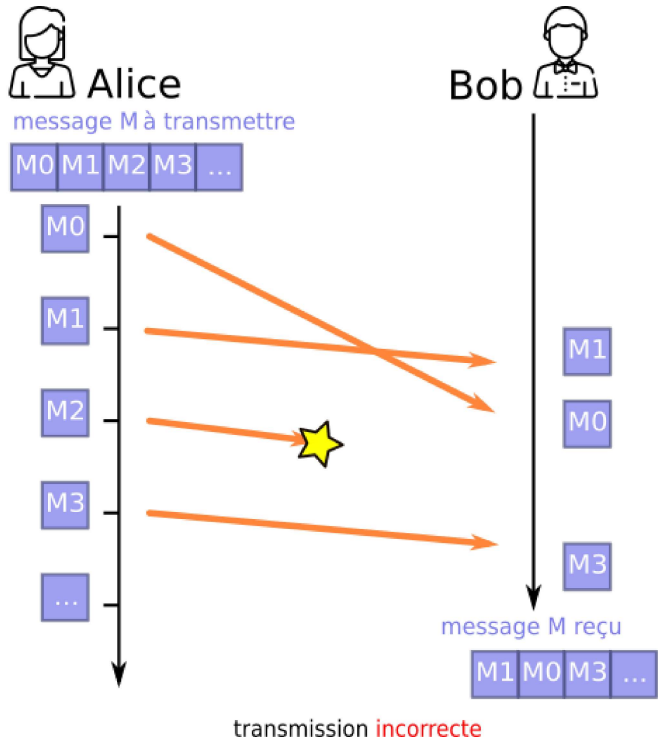
Les messages sont envoyés par Alice à une cadence Δt fixée :



Dans cette situation, les sous-messages arrivent tous à destination dans le bon ordre. La transmission est correcte.

Situation réelle

Mais parfois, les choses ne se passent pas toujours aussi bien. Car si on maîtrise parfaitement le timing de l'envoi des sous-messages d'Alice, on ne sait pas combien de temps vont mettre ces sous-messages pour arriver, ni même s'ils ne vont pas être détruits en route.

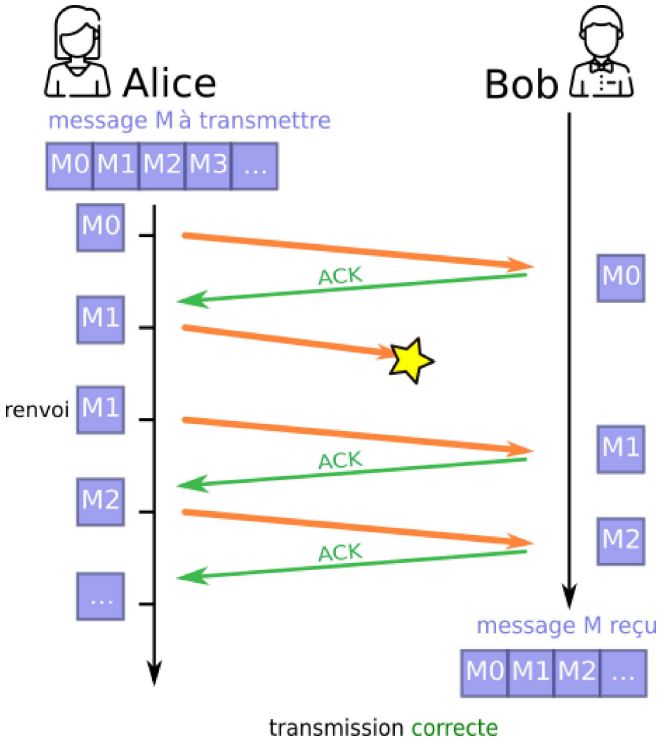


Le sous-message M0 est arrivé après le M1, le message M2 n'est jamais arrivé...

> ? Que faire ?

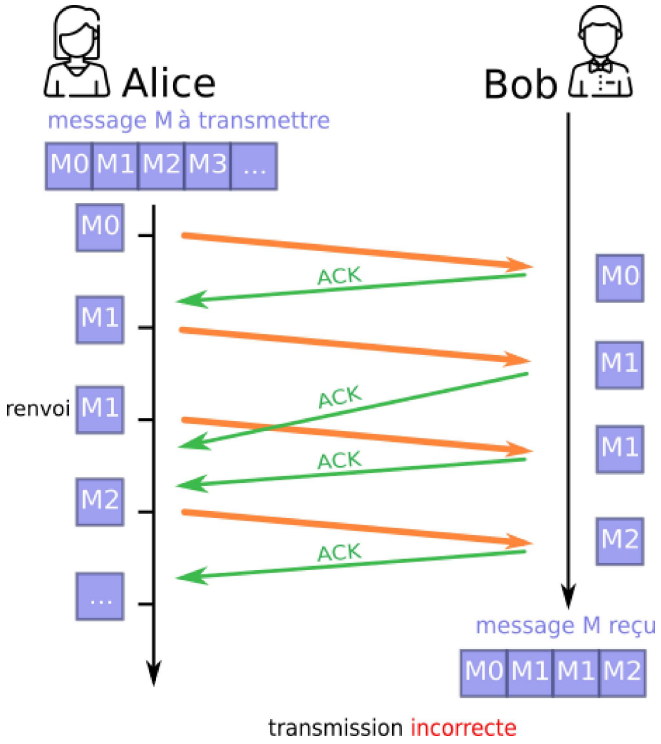
Solution naïve...

Pourquoi ne pas demander à Bob d'envoyer un signal pour dire à Alice qu'il vient bien de recevoir son sous-message ? Nous appelerons ce signal ACK (comme *acknowledgement*, traduisible par «accusé de réception»). Ce signal ACK permettra à Alice de renvoyer un message qu'elle considérera comme perdu :



N'ayant pas reçu le ACK consécutif à son message M1, Alice suppose (avec raison) que ce message n'est pas parvenu jusqu'à Bob, et donc renvoie le message M1.

Mais peu efficace...



Le deuxième ACK de Bob a mis trop de temps pour arriver et donc Alice a supposé que son sous-message M1 n'était pas arrivé. Elle l'a donc renvoyé, et Bob se retrouve avec deux fois le sous-message M1.

À faire

Que se passe-t-il si le deuxième ACK de Bob est perdu ?

Faites un schéma correspondant à cette situation et expliquez le problème.

Dans les deux cas la transmission est incorrecte, et pour l'instant rien ne nous permet de le détecter.

Le protocole du bit alterné

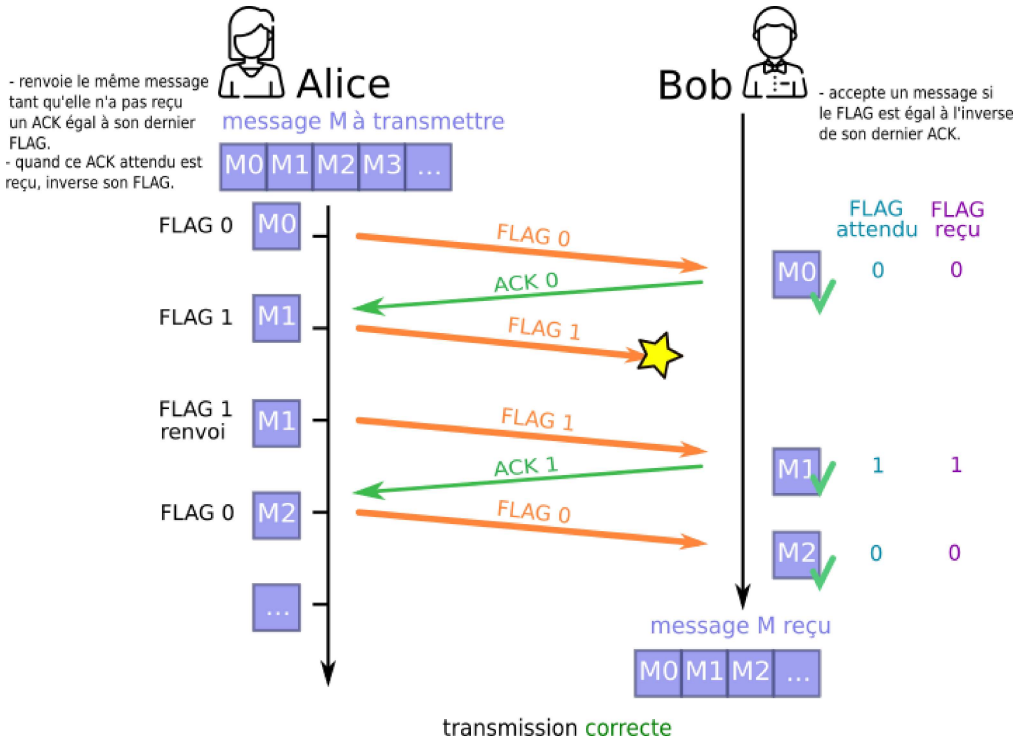
Bob va maintenant intégrer une méthode de validation du sous-message reçu. Il pourra décider de le garder ou de l'écarter. Le but est d'éviter les doublons.

Pour réaliser ceci :

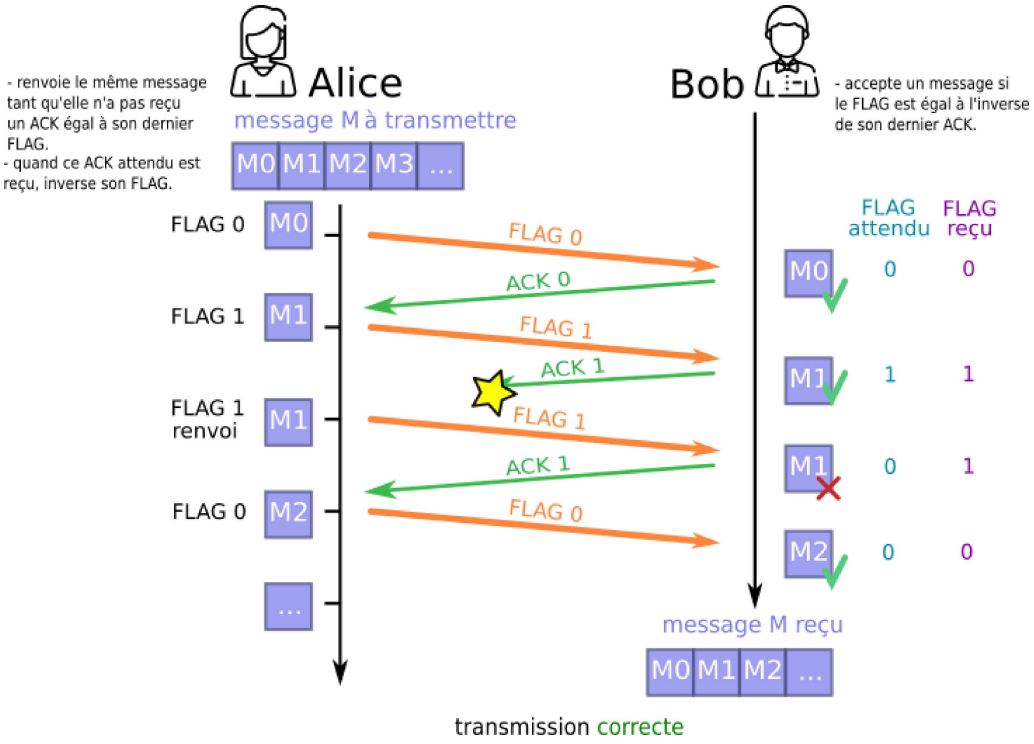
- Alice va rajouter à chacun de ses sous-messages un bit de contrôle, que nous appelerons FLAG (drapeau). Au départ, ce FLAG vaut 0.
- Quand Bob reçoit un FLAG, il renvoie un ACK **égal au FLAG reçu**.
- Alice va attendre ce ACK contenant le même bit que son dernier FLAG envoyé :
  - tant qu'elle ne l'aura pas reçu, elle continuera à envoyer *le même sous-message, avec le même FLAG*.
  - dès qu'elle l'a reçu, elle peut envoyer un nouveau sous-message en *inversant* (« alternant ») *le bit de son dernier FLAG* (d'où le nom de ce protocole).
- Bob, de son côté, va contrôler la validité de ce qu'il reçoit : il ne gardera que *les sous-messages dont le FLAG est égal à l'inverse de son dernier ACK*. C'est cette méthode qui lui permettra d'écarter les doublons.

Observons ce protocole dans plusieurs cas pour voir qu'il résout les problèmes identifiés précédemment :

Cas où le sous-message est perdu

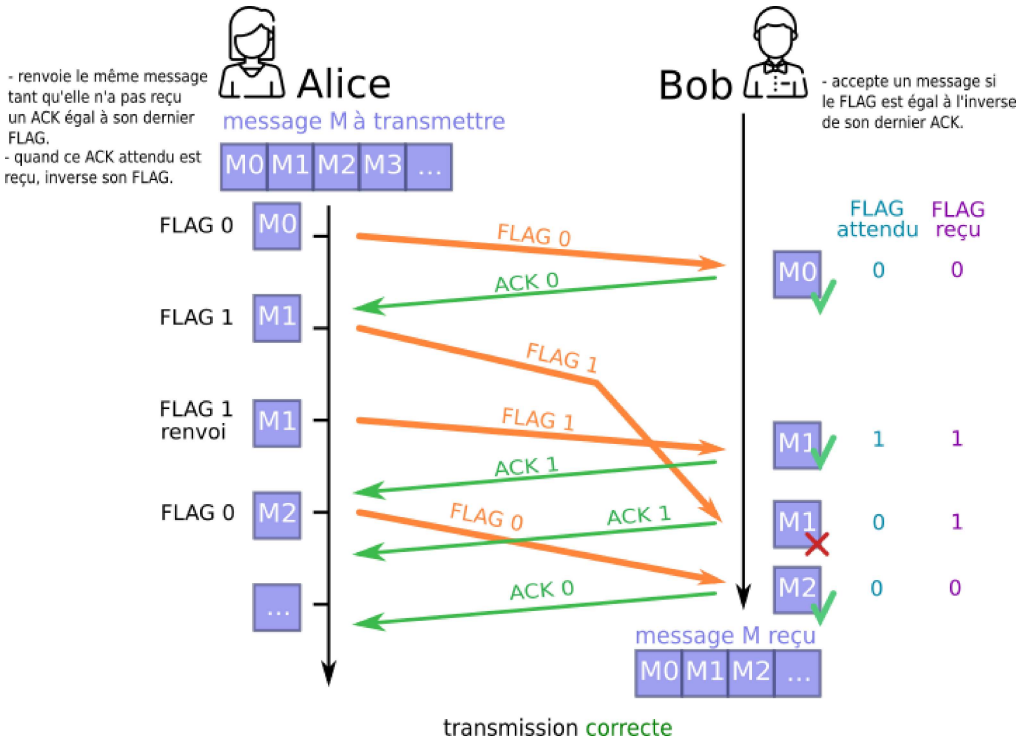


Cas où le ACK est perdu



Le protocole a bien détecté le doublon du sous-message M1.

Cas où un sous-message est en retard



Le protocole a bien détecté le doublon du sous-message M1...

Cas où le protocole du bit alterné est inefficace

Il existe cependant des situations dans lesquelles le protocole du bit alterné ne permet pas d'assurer une transmission correcte.

À faire

L'objectif est de trouver deux situations pour lesquelles la transmission selon le protocole du bit alterné est inefficace.

1. Que se passerait-il si notre premier sous-message M1 était *encore plus* en retard ? Faites un schéma pour expliquer le problème.
2. Trouvez une autre situation dans laquelle la transmission est inefficace. *Indice : ce protocole ne sait compter que jusqu'à 1.*

Conclusion

Le protocole du bit alterné a longtemps été utilisé au sein de la couche 2 du modèle OSI (distribution des trames Ethernet). Simple et léger, il peut toutefois être facilement mis en défaut, ce qui explique qu'il ait été remplacé par des protocoles plus performants, mais plus complexes, comme TCP (qui intervient au niveau de la couche *Transport* quant à lui).

Certains protocoles ne prévoient pas de système d'accusés de réception : l'émetteur ne sait donc pas si tous les paquets arrivent à destination. C'est le cas du *protocole UDP*, très similaire à TCP (couche *Transport*), mais dont les transmissions sont moins fiables puisqu'un paquet perdu l'est définitivement et n'est jamais renvoyé.

En revanche, UDP est beaucoup plus rapide (puisqu'il n'y a pas d'accusé de réception à transmettre, et qu'en plus l'établissement de la connexion est inutile) et est très souvent utilisé lorsque la rapidité des échanges est primordiale et que la perte d'un datagramme est moins gênante : c'est par exemple le cas du *streaming vidéo* (il vaut mieux perdre une partie d'une image que d'attendre quelques secondes avant que celle-ci ne soit renvoyée) ou de la *voix sur IP* (VoIP, *Voice on IP* en anglais) utilisée par les applications de messageries (WhatsApp, Messenger, Discord, Skype, etc.), ou encore des jeux en ligne.

Bilan

- Un **protocole de communication** est un **ensemble de règles** précisant le format des informations échangées entre deux machines, la manière de les échanger et la manière d'établir la communication et de la terminer.
- Les messages échangés sont souvent découpés en plusieurs paquets, chaque paquet étant acheminé vers le destinataire. Les protocoles TCP et IP sont les plus utilisés actuellement pour faire communiquer les machines sur le réseau Internet.
- En réalité, les communications sur un réseau se font via des **couches de protocoles** qui encapsulent les informations de la couche supérieure en ajoutant d'autres informations essentielles. Chaque couche a un rôle précis et on distingue deux modèles importants : le **modèle Internet** (ou *modèle TCP/IP*), et le **modèle OSI** qui est plus généraliste.
- La couche la plus "haute" (celle la plus proche de l'utilisateur) s'appelle la couche *Application* et correspond aux protocoles mis en oeuvre par les applications d'un ordinateur pour encapsuler les données à envoyer (un email, une requête HTTP, ...).
- La couche la plus "basse" (celle la plus proche du matériel) s'appelle la couche "Accès réseau" (ou "laison de données" dans le modèle OSI) et met en oeuvre des protocoles pour fabriquer la suite de bits (appelée *trame*) qui sera envoyée physiquement entre les deux machines. Cette trame contient les **adresses MAC** (aussi appelées *adresses physiques*) de ces dernières.
- Des protocoles existent pour assurer la fiablilité des échanges lorsque c'est nécessaire. Le **protocole du bit alterné** en est un exemple simple mais qui se révèle inefficace dans certaines situations, c'est pourquoi d'autres protocoles bien plus complexes sont actuellement utilisés, comme par exemple TCP.

Références :

- Cours *Introduction aux réseaux* du DIU EIL de l'université de Nantes, Pierrick Passard et Salima Hamma.
- Cours de Gilles Lassus sur les Protocoles de communication.
- Cours de David Roche sur le Modèle TCP/IP

Germain Becker, Lycée Emmanuel Mounier, Angers.

Voir en ligne : info-mounier.fr/premiere_nsi/archi_os/protocoles