1 1.B - Architecture en Couches

Nous allons maintenant parler de la transmission de l'information. Qu'est-ce qu'une liaison ? Quels sont les supports de transmission ? Quelles sont les topologies présentes dans les réseaux ? Dans une seconde partie, nous parlerons des architectures réseaux, en particulier de l'architecture TCP/IP d'Internet et du modèle OSI, un cadre de référence qui décrit ce que devrait être l'architecture de n'importe quel réseau et auquel tout le monde se réfère.

1.1 Quelques éléments sur la transmission de l'information

1.1.1 L'information et sa transmission

Lors de la transmission de données, nous avons deux grandes catégories. Nous avons déjà mentionné les données multimédias, telles que le son, l'image et les vidéos. Mais il existe également des données de type discret et fini, comme par exemple un fichier de n'importe quel type. Ce fichier a un début et une fin dont le contenu peut facilement être traduit en une séquence binaire, à condition de mettre en place un codage binaire pour que chaque information contenue dans le fichier.

D'un autre côté, nous avons des données continues, c'est-à-dire correspondant à la variation en continu d'un phénomène physique comme la voix, la température ou la lumière. Dans ce cas, nous avons typiquement un signal analogique pour la voix, qui contient une infinité de valeurs dans un intervalle borné, et qui doit également être traduit en une séquence binaire avant d'être transmis sur une liaison à l'aide d'un nouveau signal adapté au support de transmission utilisé.

Quel que soit le type de données à transférer, nous devons nous ramener à une séquence binaire avant de générer un signal adapté au type de liaison que nous allons utiliser. Par exemple, pour la voix, nous devrons utiliser un convertisseur analogique-numérique pour nous ramener à une séquence binaire et échantillonner le signal, c'est-à-dire ne pas transmettre une infinité de points mais plutôt des valeurs entières sur des intervalles de temps réguliers. Ainsi, les données peuvent être transmises entre les différentes entités du réseau.

De l'autre côté, lorsqu'un équipement reçoit la séquence binaire correspondant à ce signal analogique, il effectue la conversion inverse et transforme le signal numérique reçu en un signal analogique. Il reconstitue ainsi le signal d'origine à partir des différents points et de leur valeur numérique transmise. Il est clair que, en procédant de cette manière, nous perdons de l'information lors de l'échantillonnage, mais nous sommes également en mesure de gommer les erreurs qui peuvent survenir pendant la transmission, comme des bits qui changent de valeur. En effet, la reconstruction du signal analogique permet de repositionner les points qui sont trop éloignés de la courbe sur celle-ci.

Pour les deux types de données, il est nécessaire de transformer l'information d'origine en une séquence binaire. Cette opération s'appelle le codage. Cela consiste à associer une séquence binaire à chaque symbole du fichier, comme par exemple une lettre de l'alphabet. Pour l'alphabet, c'est le code ASCII qui effectue cette tâche. Ainsi, que nous partions d'un signal analogique comme la voix ou de symboles dans un fichier, nous nous ramenons toujours à une séquence binaire qui va elle-même être transformée en un signal numérique adapté au support de transmission de chaque liaison traversée dans le réseau.

1.1.2 Techniques de transmission

Pour transmettre la séquence binaire sur la liaison, nous utilisons des techniques de transmission qui consistent à générer un signal adapté à cette liaison. Par exemple, pour Ethernet, nous fabriquons un signal électrique à deux niveaux de tension, 5 volts et -5 volts, où chaque niveau de tension correspond à une valeur binaire. Nous pouvons ainsi dire que le bit '1' correspond à 5 volts et le bit '0' correspond à -5 volts.

Pour générer ce signal électrique, la carte réseau utilise ce qu'on appelle un codeur et, de l'autre côté, un décodeur. Ces deux éléments permettent simplement de traduire la séquence binaire en un signal qui circule sur le câble Ethernet. Cette technique de transmission est appelée 'transmission en bande de base' et est relativement simple. La carte réseau peut ainsi rapidement transmettre ce signal électrique sur un support en cuivre.

Figure 1: Signal Numérique

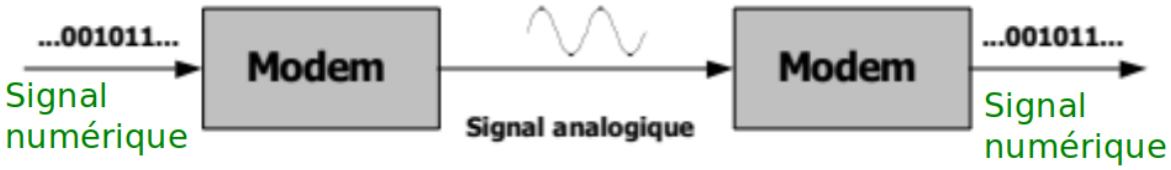

Figure 2: Signal Analogique

Il existe également d'autres techniques de transmission pour d'autres types de liaisons, qui consistent à transmettre un signal analogique modulé sur la liaison. Par exemple, cette technique est utilisée pour transmettre des signaux sans fil à des fréquences différentes.

L'avantage de ce type de transmission est qu'il permet de transmettre plusieurs signaux dans un même espace grâce à des fréquences différentes. Ces signaux ont également des propriétés qui permettent de couvrir de plus grandes distances, ce qui les rend adaptés pour des réseaux sur des distances allant jusqu'à plusieurs kilomètres.

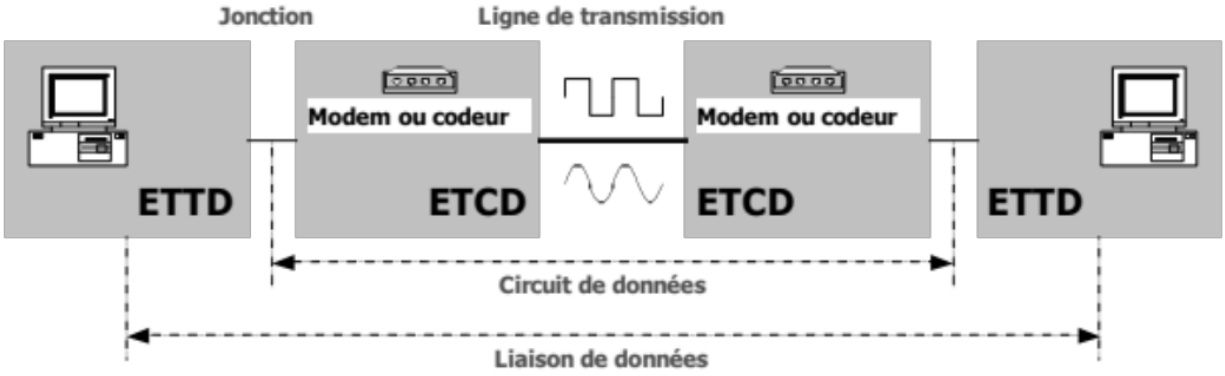

Figure 3: ligne de transmission

Quand nous parlons de liaison, nous faisons référence à une carte réseau d'un côté de la liaison, une autre carte réseau de l'autre côté et un support de transmission entre les deux qui permet de propager un signal analogique, un signal électrique ou une onde lumineuse dans le cas de la fibre optique. Le rôle de la carte réseau est de prendre la séquence binaire provenant de l'application ou de l'équipement qui souhaite transmettre une trame et de générer le signal correspondant à la séquence binaire et adapté au support de

transmission.

1.1.3 Les supports de transmission

Les supports de transmission comprennent notamment la fibre optique et les câbles en cuivre. Dans les câbles en cuivre, on utilise ce qu'on appelle une paire torsadée, principalement pour transférer les signaux de type Ethernet. Cette paire torsadée est composée de quatre paires de fils de cuivre qui se retrouvent dans un connecteur de type RJ45.

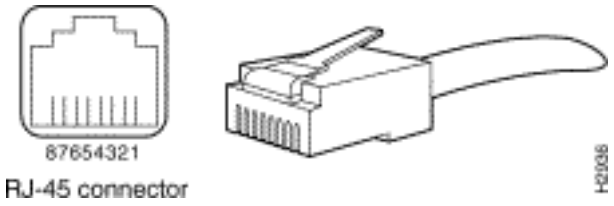

Figure 4: Prise RJ45

Généralement, une paire de fils est utilisée pour l'émission et une autre paire de fils pour la réception. Ainsi, nous utilisons au moins deux paires de fils, mais certaines versions d'Ethernet peuvent utiliser jusqu'à quatre paires de fils pour transporter un signal électrique sur de courtes distances variant de quelques mètres à quelques centaines de mètres maximum. Les propriétés de ces signaux ne permettent pas de couvrir de plus grandes distances sans l'aide d'un répéteur qui régénère le signal. Dans ce cas, il faut utiliser un équipement intermédiaire.

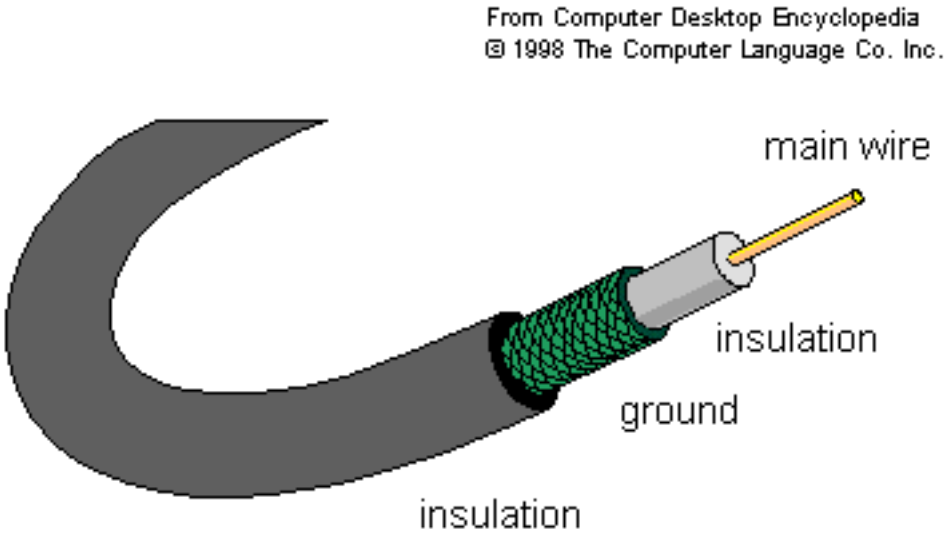

Figure 5: Cable coaxial

Le câble coaxial offre de meilleurs débits et permet de couvrir de plus grandes distances que la paire torsadée. Il est constitué d'un support en cuivre avec un seul fil plus épais qui transporte des signaux, comme ceux utilisés pour la télévision ou pour l'accès à Internet. Le câble coaxial est également mieux protégé contre les perturbations extérieures et a de meilleures performances que la paire torsadée, mais il est plus coûteux.

La fibre optique est complètement différente et consiste en la transmission d'ondes lumineuses qui ont des propriétés encore meilleures. Elle peut couvrir jusqu'à 30 kilomètres sans répéteur et est utilisée dans les réseaux locaux, métropolitains et nationaux. Elle offre également des débits supérieurs à ceux des supports en cuivre.

Les antennes sont utilisées dans les réseaux sans fil, tels que le WiFi, les réseaux mobiles et les réseaux de télévision et de radio. Elles permettent des transmissions de signaux sur de courtes ou de longues distances et sont peu coûteuses en termes d'infrastructure, car elles peuvent être placées sur les toits des immeubles plutôt que de devoir être enterrées dans les rues comme les fibres optiques. Les satellites, quant à eux, sont des répéteurs dans le ciel avec une grande couverture géographique, mais un coût de lancement très élevé. Ils ont des débits élevés mais de très mauvaises latences du fait des distances parcourues. Ils sont principalement utilisés dès lors qu'ils sont le seul moyen d'accèder au réseau Internet ou téléphonique par exemple dans le transport maritime pour les bâteaux situés en plein milieu des océans.

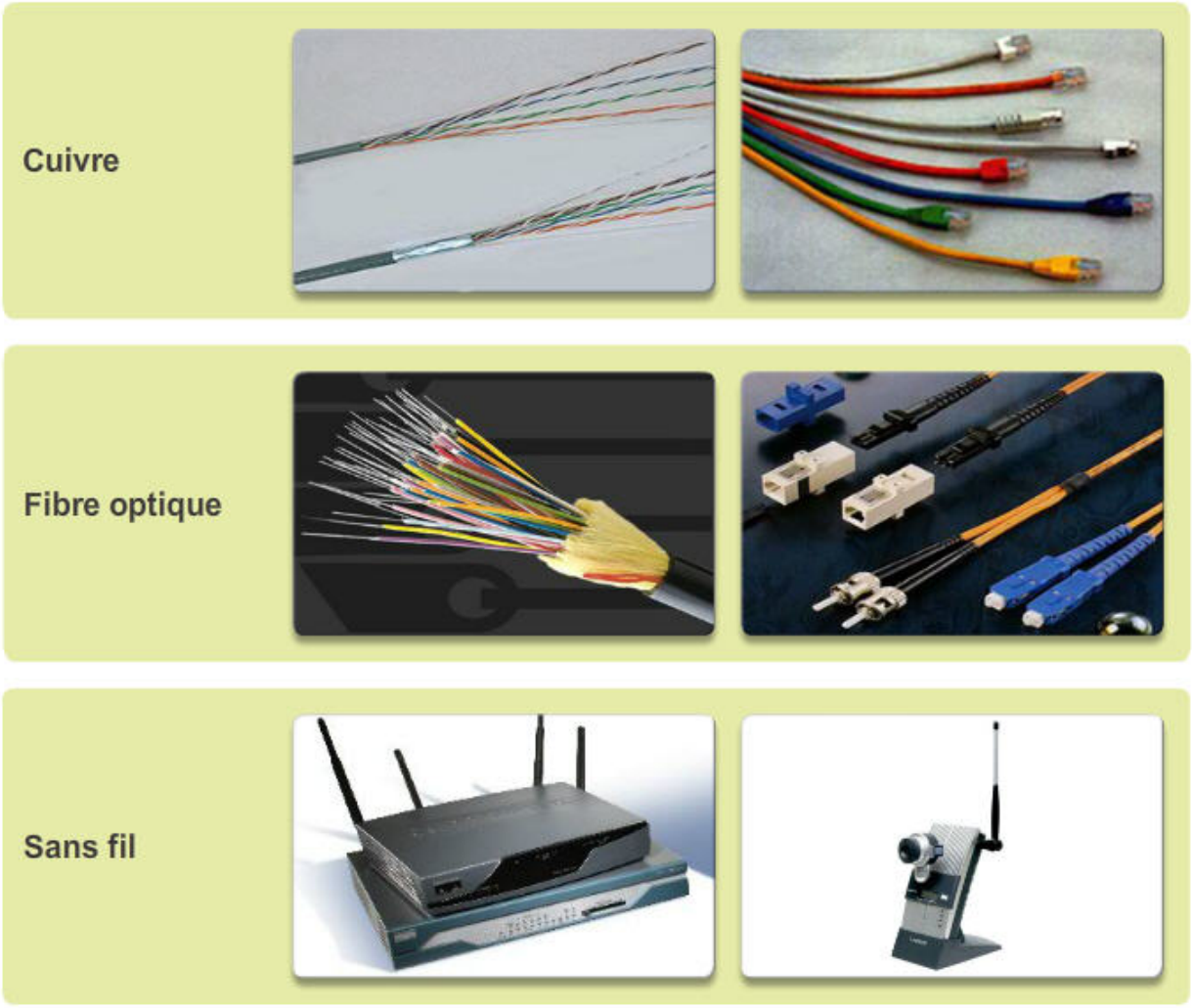

Figure 6: Supports de transmission

Quels que soient le support de transmission utilisé - cuivre, fibre optique, réseaux sans fil - des signaux seront transférés, que ce soit sous forme de signaux électriques sur des câbles en cuivre, d'impulsions lumineuses sur des fibres optiques, ou de signaux sans fil avec différentes fréquences.

Et donc c'est là où on trouve typiquement les bandes AM, FM, ou PM pour la radio. Quel que soit le support de transmission, les signaux qui sont transmis peuvent être perturbés, subir des interférences qui modifient le signal pendant le transfert et aussi de l'atténuation, c'est-à-dire que le signal perd petit à petit de la puissance et de la valeur au fil du temps et se retrouve donc dans l'obligation d'être répété lorsque les distances maximales sont atteintes.

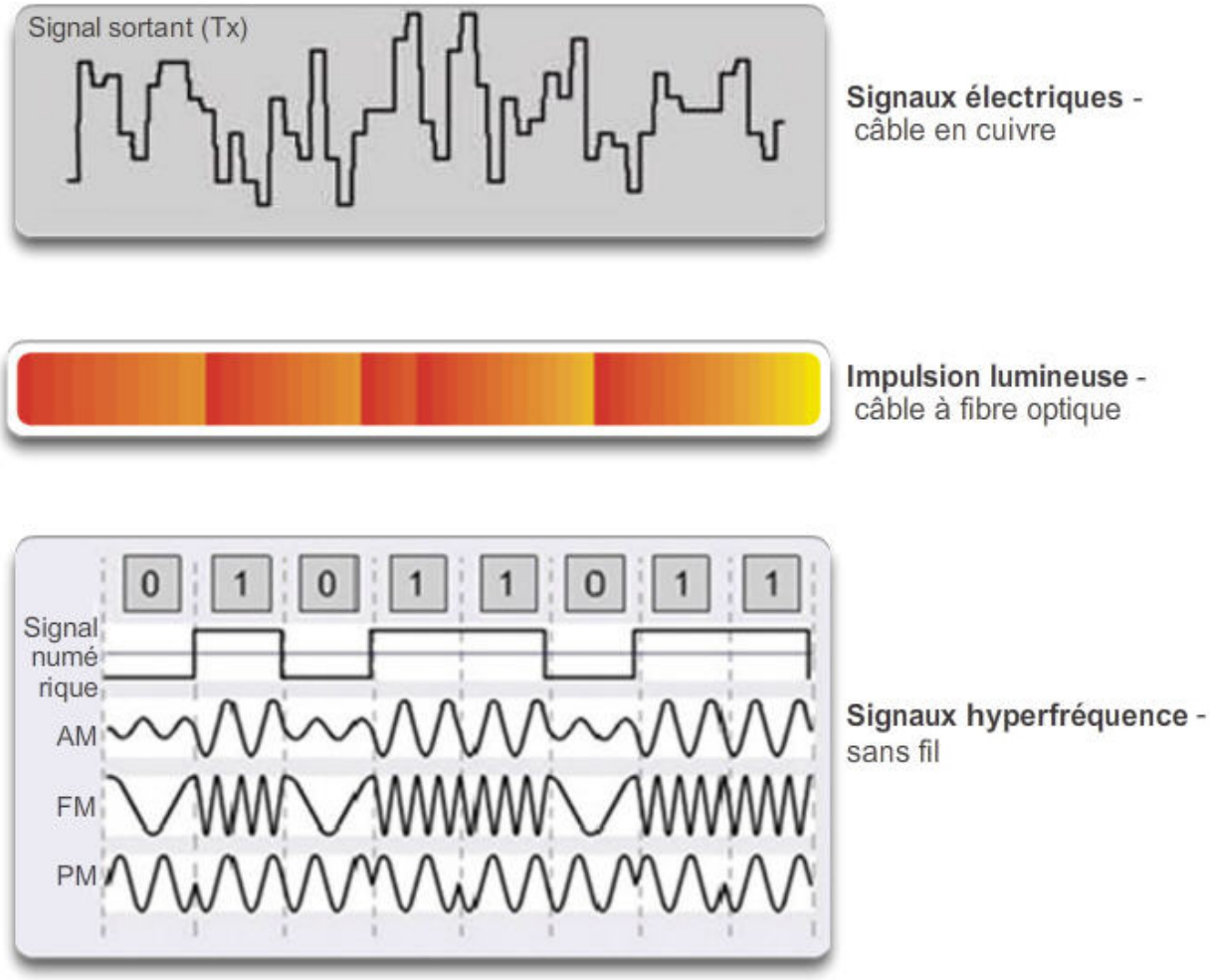

Figure 7: Signaux transmis

Par conséquent, les signaux peuvent subir des erreurs pendant leur transmission. Une erreur, c'est quand un bit change de valeur pendant sa transmission. Sur l'image ci-dessous, un signal d'interférence perturbe le signal numérique transmis. La séquence binaire reçue contient une erreur car le neuvième bit a changé de valeur. Il valait 0 avant émission mais a été lu comme la valeur 1 par la carte réseau de l'ordinateur destinataire.

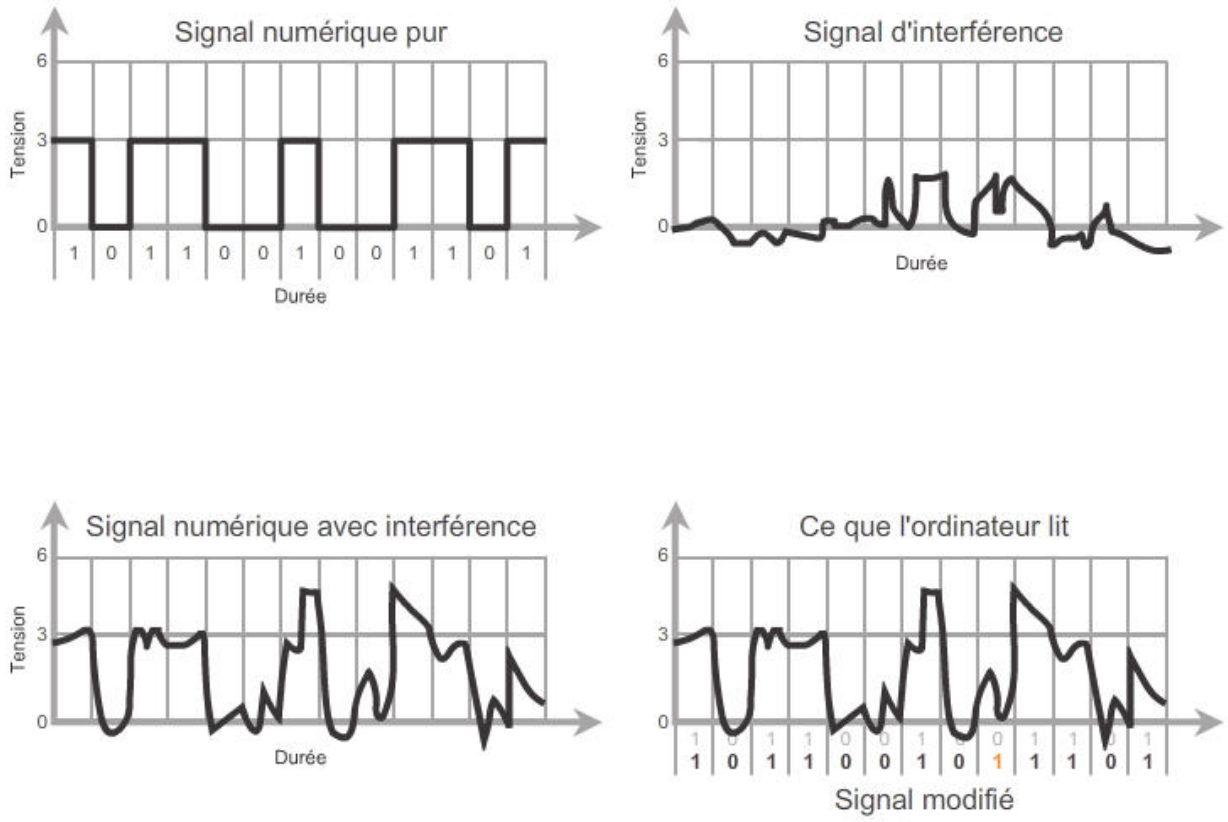

Figure 8: Signal perturbé et son interprétation

Dans un réseau, on distingue les **équipements terminaux** tels que les ordinateurs, les imprimantes, les téléphones, les tablettes, les capteurs des **équipements intermédiaires** tels que les routeurs, les commutateurs, les bornes WiFi, les pare-feu, les répéteurs. Et puis il y a les différents types de supports de transmission que nous avons déjà évoqué tels que les laisions sans fil, les liaisons filaire locales (LAN) ou longues distances (WAN) via fibre optique ou satellite.

1.1.4 Les modes de transmission

Il est important de connaître certains termes de vocabulaire liés aux liaisons. La liaison est dite **full-duplex** lorsque deux cartes réseau peuvent envoyer et recevoir des signaux simultanément. La carte réseau est alors capable de transmettre un signal tout en en recevant un autre. Cela s'applique à la plupart des réseaux filaires ou sans fil qui permettent l'envoi et la réception simultanés de signaux. Contrairement à cela, les liaisons **simplex** ne permettent l'envoi de signaux que dans un seul sens. Par exemple, les antennes de radio et de télévision ne sont capables de transmettre des données que dans une seule direction. Il y a également les liaisons **half-duplex**, dans lesquelles il est possible d'envoyer ou de recevoir des signaux, mais pas simultanément.

On parle de **liaison point-à-point** lorsqu'il n'y a que deux cartes réseau qui communiquent entre elles sur une seule et même liaison. À l'inverse, on parle de **liaison multipoints** lorsqu'il y a plusieurs cartes

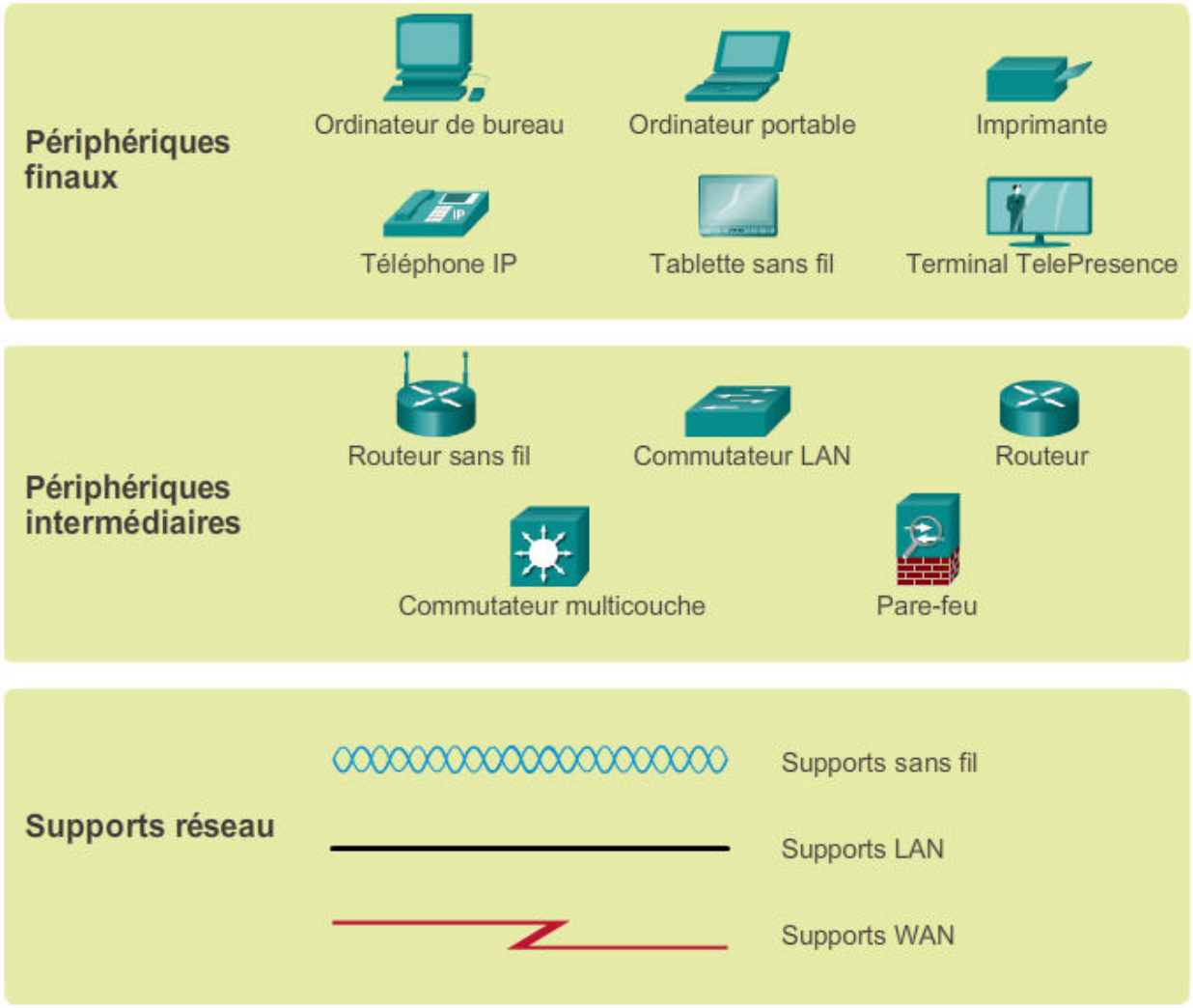

Figure 9: Les composants d'un réseau

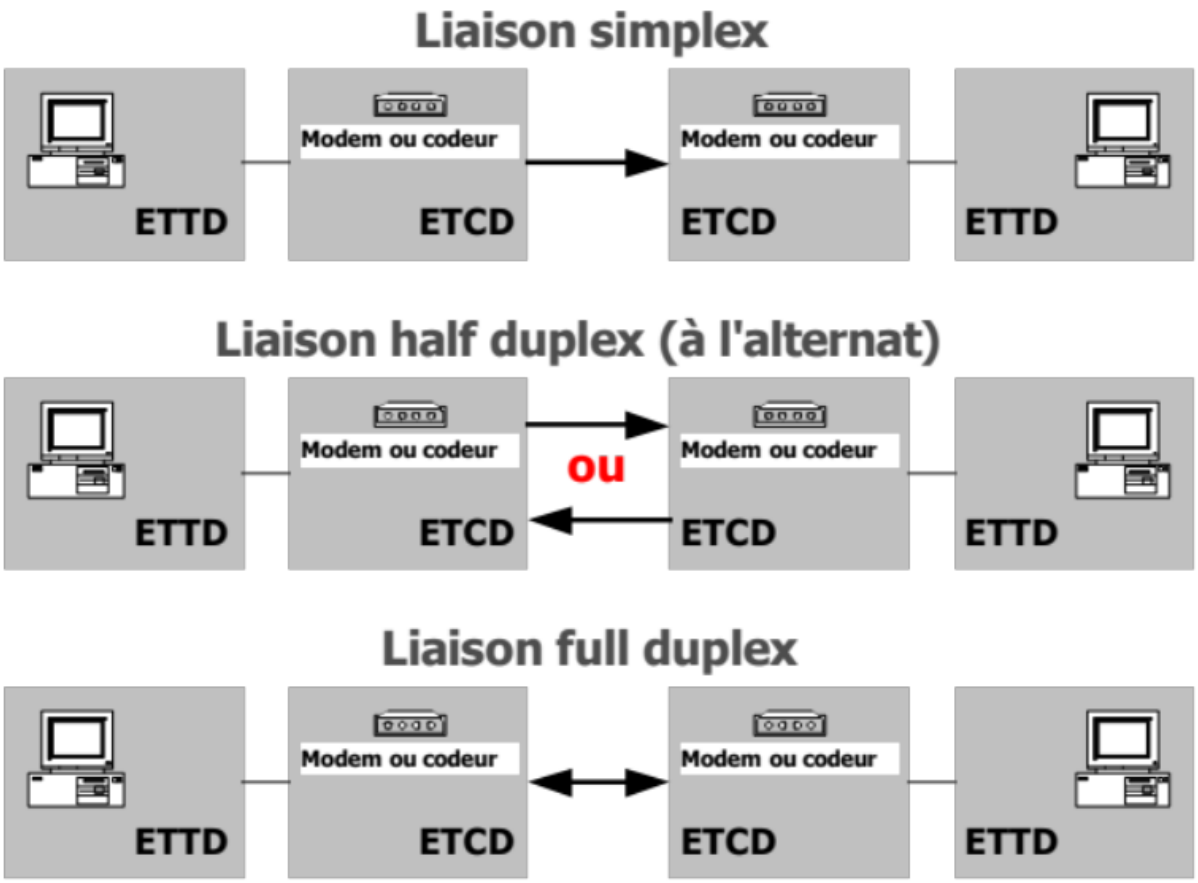

Figure 10: Différents modes de transmission

réseau qui communiquent sur le même support de transmission. Par exemple, si plusieurs ordinateurs sont connectés à un commutateur Ethernet ou à un répéteur Ethernet, ou encore si plusieurs personnes parlent simultanément dans une pièce ou si plusieurs ordinateurs communiquent via une borne WiFi, alors on se trouve dans une situation de liaison multipoints.

Il y a deux problématiques liés aux liaisons multipoints. Premièrement, il est nécessaire d'attribuer des adresses uniques aux différentes cartes réseau pour pouvoir identifier les équipements qui communiquent entre eux et savoir quelle carte réseau émet le signal et quelle(s) carte(s) réseau(x) doit (ou doivent) en être destinataire(s). Deuxièmement, il faut gérer la partage du support de transmission, c'est-à-dire déterminer qui a le droit d'émettre à un moment donné pour éviter que les signaux ne se perturbent mutuellement. Cela peut être un problème dans les réseaux Ethernet et WiFi, où plusieurs signaux doivent partager le même espace et donc soit utiliser des fréquences différentes soit émettre à tour de rôle comme nous le faisons nous les humains.

Le partage de la parole peut se faire selon trois principaux modes : le mode maître-esclave dans lequel une des cartes réseau distribue la parole aux autres cartes réseau ; le mode politesse où chaque équipement écoute avant de parler pour s'assurer qu'il n'y a pas déjà un signal en cours de transmission ; et le mode jeton où une "pièce de monnaie" virtuelle circule entre les cartes réseau et seul l'équipement qui possède le jeton peut parler pendant un temps limité. Quand il a fini, il passe le jeton au suivant et ainsi de suite.

1.1.5 Différentes topologies de liaisons

Passons maintenant aux différentes topologies qui cohabitent dans les réseaux. La première est **le bus** qui consiste en une liaison multipoints avec plusieurs ordinateurs ou cartes réseau qui communiquent entre eux et qui sont tous reliés par un unique support de transmission, un "bus central", par exemple un câble coaxial sur lequel on branche plusieurs ordinateurs ou une pièce dans laquelle plusieurs personnes se parlent ou une pièce dans laquelle plusieurs ordinateurs communiquent avec une borne Wifi.

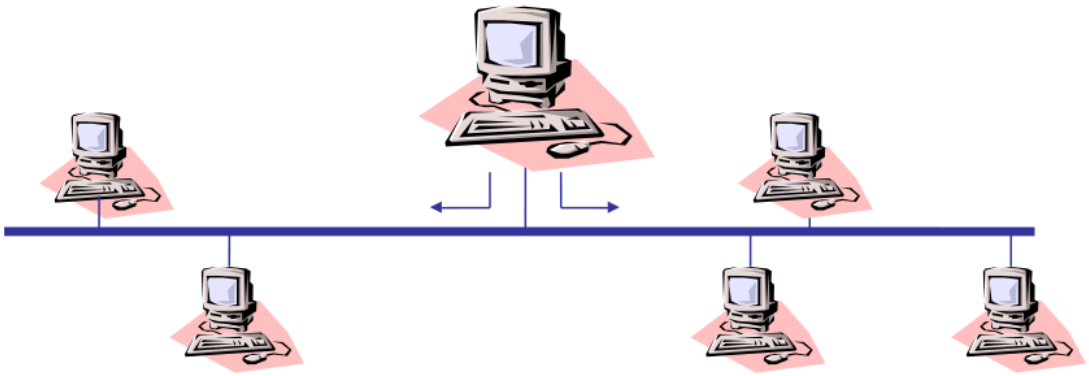

Figure 11: Topologie de bus

Avec cette topologie en bus, on a donc bien la problématique à la fois de l'adressage des cartes réseau et du partage du support.

La topologie en **étoile** consiste en un équipement central auquel sont reliées les cartes réseau. Cet équipement intermédiaire peut gèrer la liaison multipoints et le partage de la parole. C'est par exemple le rôle d'un commutateur. Mais si cet équipement n'est qu'un simple répéteur, la topologie physique est une étoile mais d'un point de vue logique elle est similaire à celle du bus car la problématique du partage de la parole perdure. Dans les deux cas, si l'équipement tombe en panne, tous les autres équipements sont coupés du réseau et ne peuvent plus communiquer.

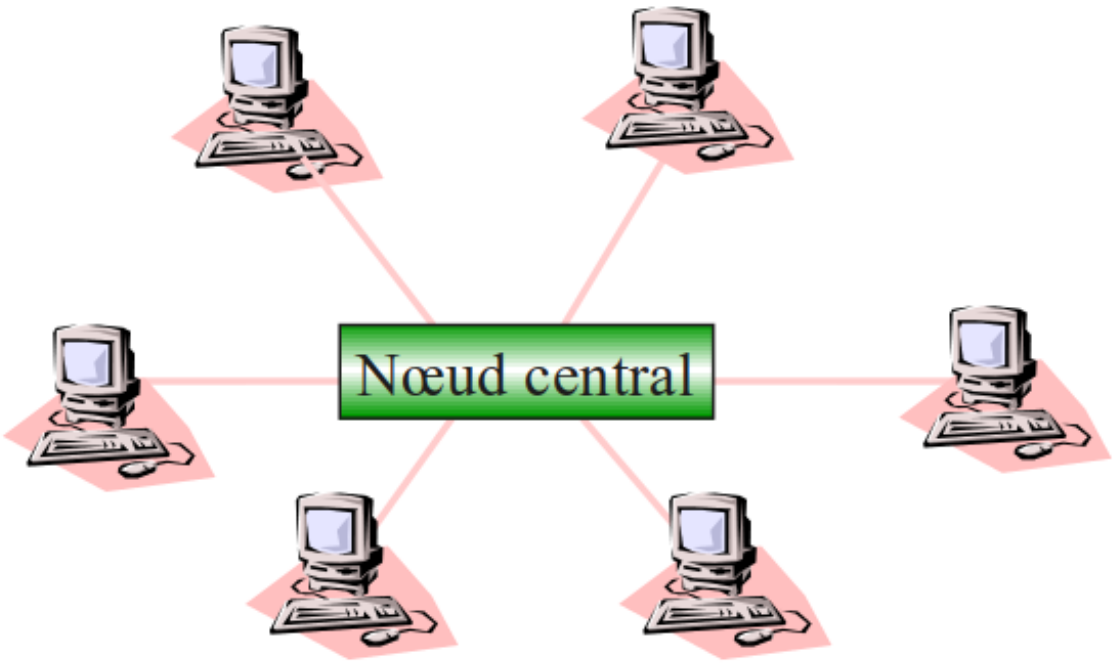

Figure 12: Topologie en étoile

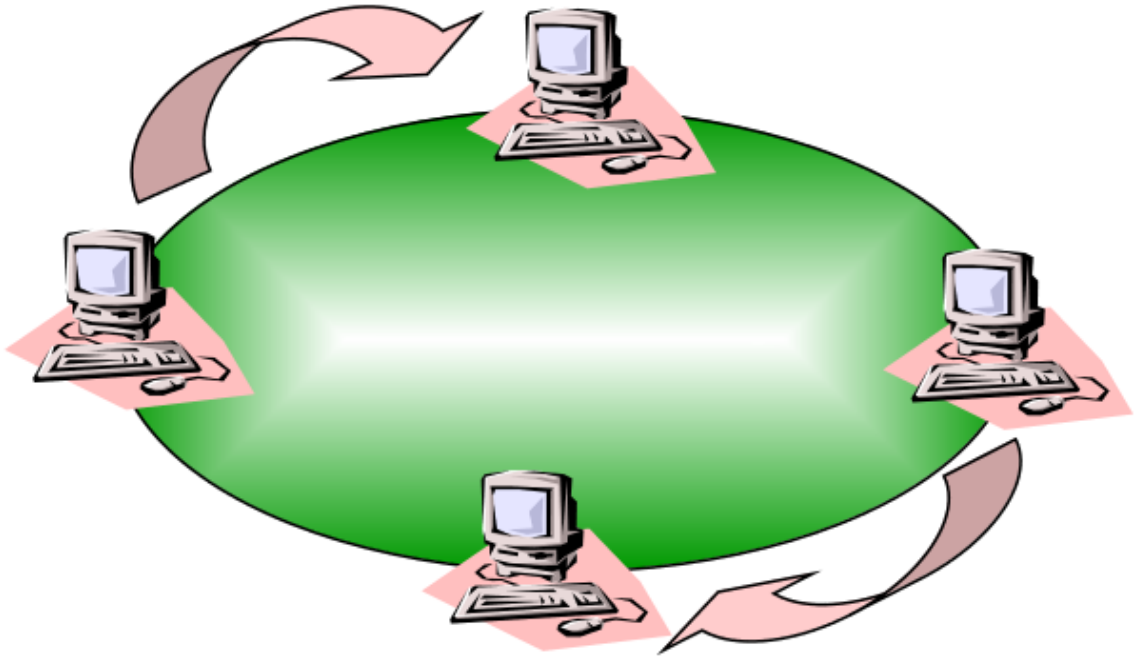

Figure 13: Topologie en anneau

Dans la topologie en **anneau**, chaque équipement est relié à deux autres équipements, le précédent et le suivant. Cela permet de parcourir de plus longues distances puisque pour aller d'un équipement à un autre, le signal doit passer par plusieurs autres équipements qui le répètent.

La topologie **maillée** consiste à relier chaque équipement à plusieurs autres et donc à rendre plusieurs chemins possibles entre deux équipements. Cela permet de choisir le meilleur chemin et de changer de chemin si nécessaire, en cas de panne par exemple. Cette topologie est souvent utilisée dans les cœurs de réseau pour avoir de la redondance et de la tolérance aux pannes. Cependant, cela coûte cher en termes de nombre de liaisons et de câbles. On parle de topologie maillée totale lorsque chaque équipement est relié directement à tous les autres, sans passer par un intermédiaire. Cette topologie est très coûteuse car elle nécessite un nombre de liaisons de l'ordre de n au carré si n est le nombre d'équipements. Sur Internet, on peut retrouver tous ces types de topologie mais les cœurs de réseau utilisent généralement une topologie maillée.

La redondance, c'est le fait de pouvoir avoir plusieurs chemins disponibles. Cela permet d'avoir de meilleures performances et de tolérer les pannes. Dans les réseaux locaux, on utilise souvent la topologie en étoile, avec un équipement central qui sert de commutateur ou de borne Wifi pour relier entre eux les ordinateurs locaux mais aussi de routeur pour sortir sur Internet. C'est typiquement ce que fait la box fournie par votre opérateur. Si cet équipement tombe en panne, le réseau local ne peut plus communiquer ni en interne ni en externe.

Il existe de nombreuses liaisons sous-marines sur Internet, en plus des liaisons par satellite. Vous pouvez consulter ces différentes liaisons sur le site submarinecable.com, qui vous donnera des informations sur le propriétaire et le gestionnaire de chaque liaison, ainsi que sur les points d'accès disponibles.

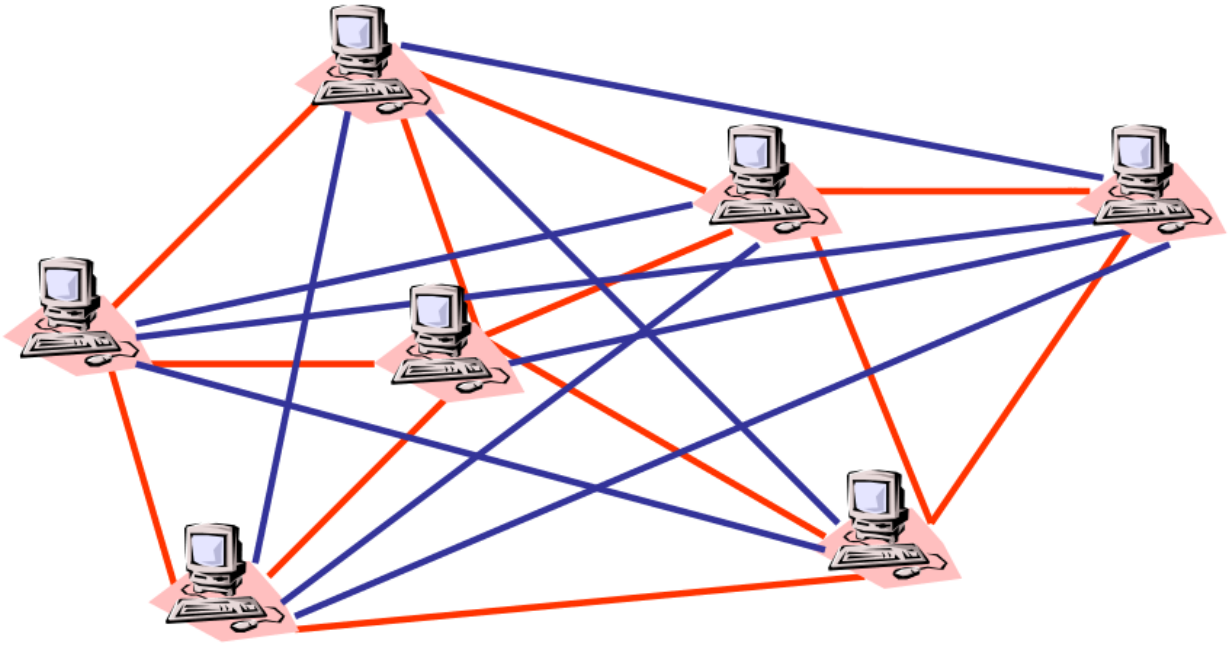

Figure 14: Topologie maillée

1.2 Les architectures protocolaires

Dans cette partie, nous allons décrire les architectures de réseau. Une architecture de réseau est un ensemble de protocoles qui cohabitent et permettent tous ensemble de faire fonctionner le réseau. Par exemple, **l'architecture TCP/IP** regroupe l'ensemble des protocoles qui font fonctionner Internet. Nous allons voir comment ces architectures sont conçues et comment elles fonctionnent.

1.2.1 Architecture en couches et encapsulation

Une architecture de réseau est un assemblage d'algorithmes, de logiciels et de matériel qui coopèrent pour faire fonctionner un réseau dans sa globalité. Historiquement, les premiers réseaux informatiques étaient créés par un unique constructeur, ce qui posait des problèmes de compatibilité entre équipements et logiciels de différents constructeurs. Afin de développer un marché plus vaste et un nombre accru d'utilisateurs, il est donc rapidement devenu nécessaire de normaliser les architectures en rendant les briques qui les constituent normalisées et connues de tous. Le **modèle de référence OSI** (Open Systems Interconnexion) a alors été conçu pour rassembler dans une seule architecture toutes les briques de base présentes dans tous les réseaux, chaque réseau spécifique étant finalement un sous-ensemble de ce modèle de référence théorique, qui n'a pas d'existence réelle en tant que tel.

L'**architecture protocolaire** est donc l'ensemble des protocoles qui sont utilisés pour faire fonctionner un réseau. On distingue généralement trois familles de protocoles : les protocoles applicatifs, qui définissent pour chaque application comment elle doit échanger des données pour rendre le service qui est attendu d'elle ; les protocoles de transport qui permettent de définir comment transporter l'information d'un émetteur à un récepteur ; et les protocoles de liaison qui sont proches du matériel et qui servent à transmettre un signal sur une liaison de transmission. Ces protocoles sont exécutés respectivement dans les applications, le système d'exploitation et la carte réseau.

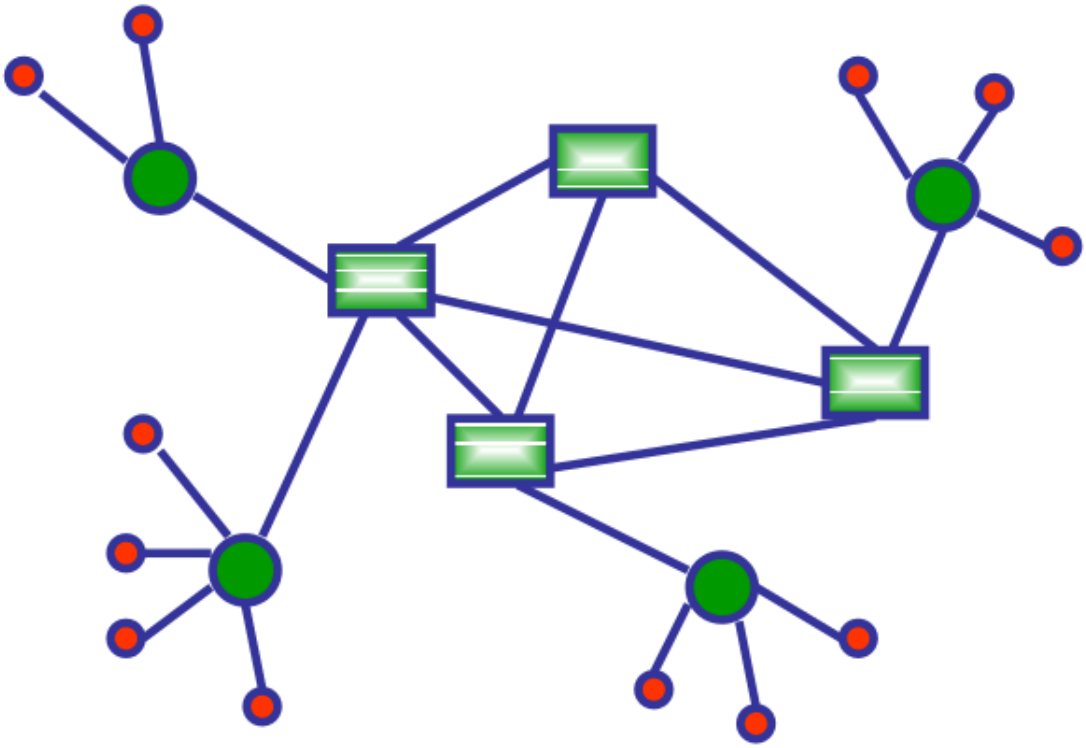

Figure 15: Topologie d'Internet

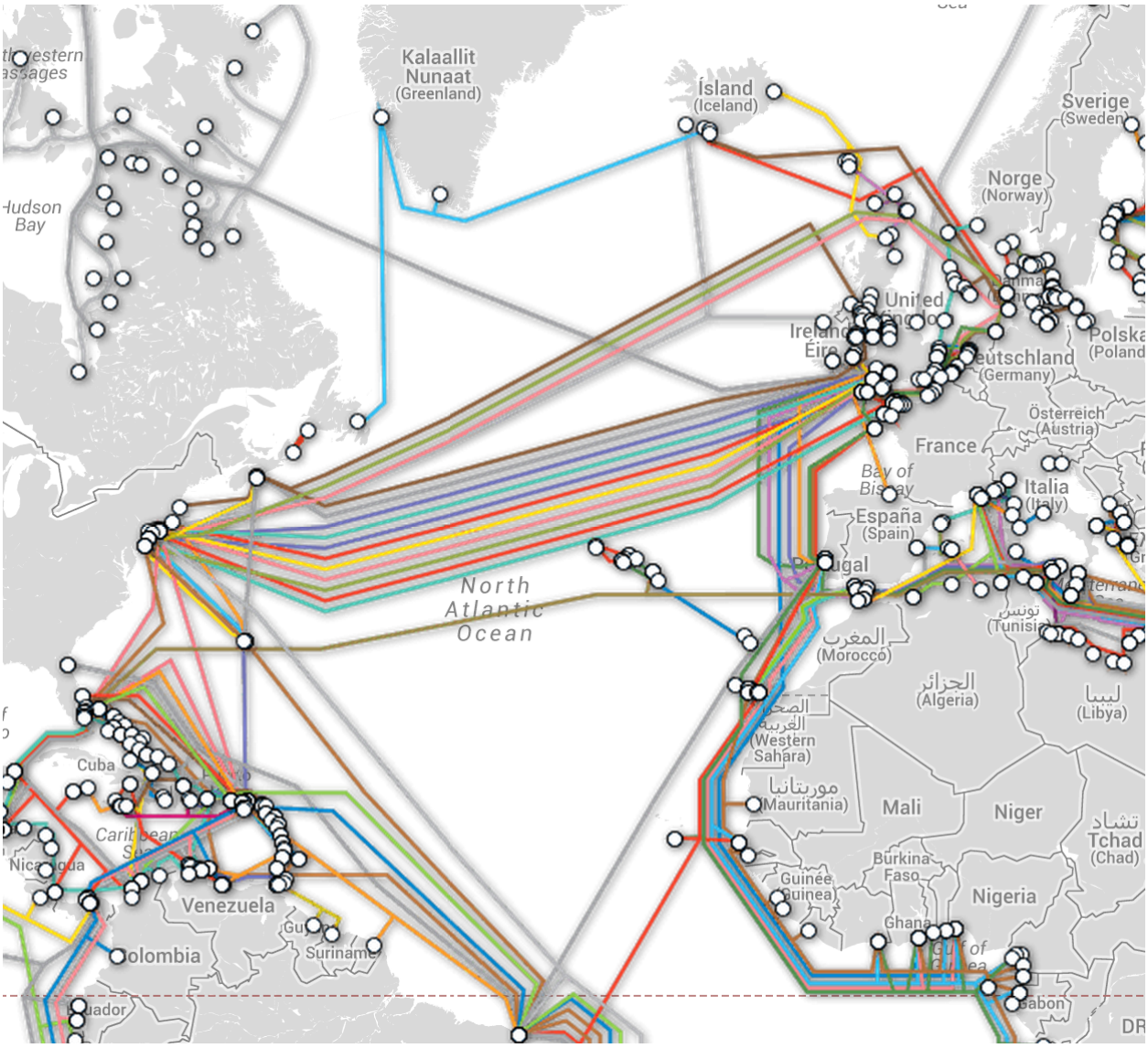

Figure 16: Cables sousmarins

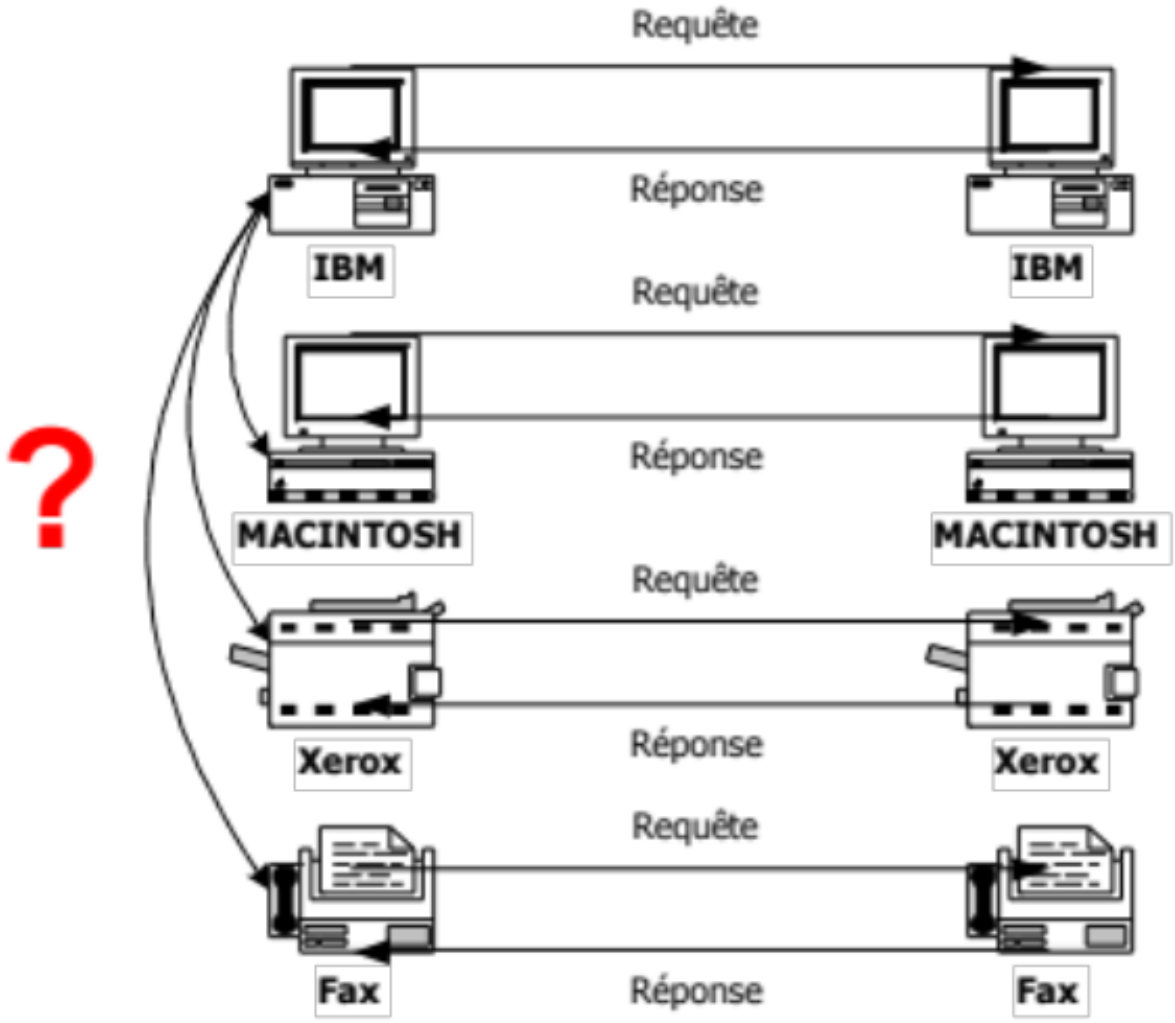

Figure 17: Besoin de transparence

1.2.2 Protocole

Un protocole est une brique de l'architecture réseau qui a un rôle, une fonction bien définie. Il rend un service à la demande d'un utilisateur du réseau, d'une application ou d'un autre protocole du réseau. Il s'exécute entre au moins deux entités comme un client et un serveur d'une même application, deux systèmes d'exploitation de deux ordinateurs ou équipements intermédiaires ou deux cartes réseaux. Il définit l'algorithme qui s'éxecute de part et d'autre ainsi que le format des messages échangés entre ces deux entités. Par exemple, un acquittement c'est pour dire "j'ai bien reçu !". Pourquoi avons-nous besoin de normaliser les protocoles ? Si un protocole est normalisé, cela signifie que les règles d'échanges, les algorithmes et le format des messages sont spécifiés dans un document accessible à tous les acteurs économiques, tels que les concepteurs d'applications, de systèmes d'exploitation et de matériel réseau. Ainsi, lorsqu'un constructeur souhaite mettre en œuvre un protocole d'une architecture réseau, il doit se référer à cette norme pour garantir que ce qu'il va produire sera compatible avec n'importe quel autre équipement ou logiciel nécessitant l'usage du même protocole. Un système d'exploitation qui ne respecterait pas la norme IP ou TCP ne pourrait pas utiliser Internet. Un navigateur Web qui ne respecte pas la norme HTTP ou DNS ne peut pas demander une page Web à un serveur Web.

On parle également d'architecture en couches. Une couche est l'implémentation d'un protocole. L'architecture en couches regroupe les différentes couches qui font appel les unes aux autres pour faire fonctionner un réseau. Chaque couche, après avoir réalisé son travail et ajouté ses informations aux données de la couche supérieure transmet son message à la couche inférieure pour qu'elle réalise à son tour le travail qui est attendu d'elle. L'intérêt de diviser l'architecture réseau en plusieurs couches est de pouvoir isoler les différentes fonctions du réseau et de rendre l'architecture évolutive. Par exemple, le protocole HTTP gère les requêtes et réponses pour échanger des pages Web, le protocole IP achemine les paquets sur Internet, et le protocole Ethernet s'occupe de la transmission de signaux entre deux cartes réseau sur un support Ethernet.

En effet, en divisant l'architecture réseau en couches, on peut facilement faire évoluer certaines parties du réseau sans avoir à toucher aux autres. Cela rend l'architecture plus évolutive et permet de réutiliser des fonctions communes à différentes applications. Par exemple, le protocole IP est nécessaire pour la plupart des applications, de même que le protocole TCP qui assure la fiabilité des transmissions. On peut donc imaginer cette architecture en couches comme une grande application composée de différentes librairies qui coopèrent pour réaliser l'ensemble des fonctions du réseau.

Le protocole DNS est utilisé pour nommer les machines sur Internet, en particulier les serveurs. Par exemple, lorsqu'un utilisateur tape l'adresse d'un site Web dans son navigateur, le DNS est utilisé pour trouver l'adresse IP associée au nom de serveur. Cela permet d'acheminer les requêtes depuis les navigateurs Web vers le serveur Web en traversant Internet. Toutes les applications font appel au DNS pour trouver les adresses IP associées aux noms de serveurs. C'est pourquoi le DNS se trouve également dans une couche séparée afin de permettre l'évolution éventuelle de ce protocole et la réutilisation de ce service par toutes les applications.

On fait souvent l'analogie entre l'architecture en couches des réseaux et le fonctionnement du courrier postal. Par exemple, pour envoyer une lettre de France en Italie, il est nécessaire d'établir un dialogue entre l'expéditeur et le destinataire ne serait-ce que pour se mettre d'accord sur la langue dans laquelle ils se parlent, éventuellement faire appel à un service de traduction. Il faut ensuite un protocole d'échanges entre les services postaux des deux pays pour déterminer comment acheminer la lettre, le format de l'adresse et définir le chemin à suivre, les moyens de transport à utiliser. Cette analogie met en avant les trois couches principales de l'architecture en couches des réseaux : les couches applicatives, la couche de transport et la couche de transmission physique.

Ainsi, l'architecture en couches des réseaux comprend trois couches principales : les couches applicatives, avec un protocole pour chaque application, la couche de transport, qui détermine le chemin à suivre, et la couche de transmission qui gère la transmission physique des données sur le réseau. Cette architecture en couches est comparable au fonctionnement du courrier postal, avec les différents services et fonctions à mettre en place pour acheminer une lettre ou un colis.

Cette figure est peut-être la plus importante de toutes. Elle définit ce qu'est une architecture en couches et

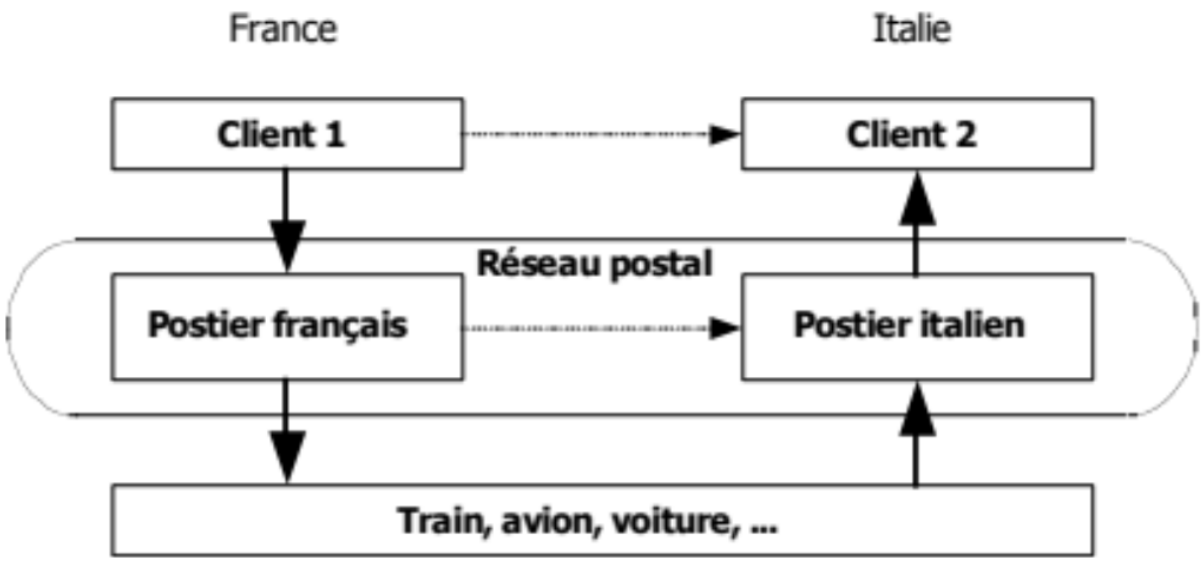

Figure 18: Courrier postal

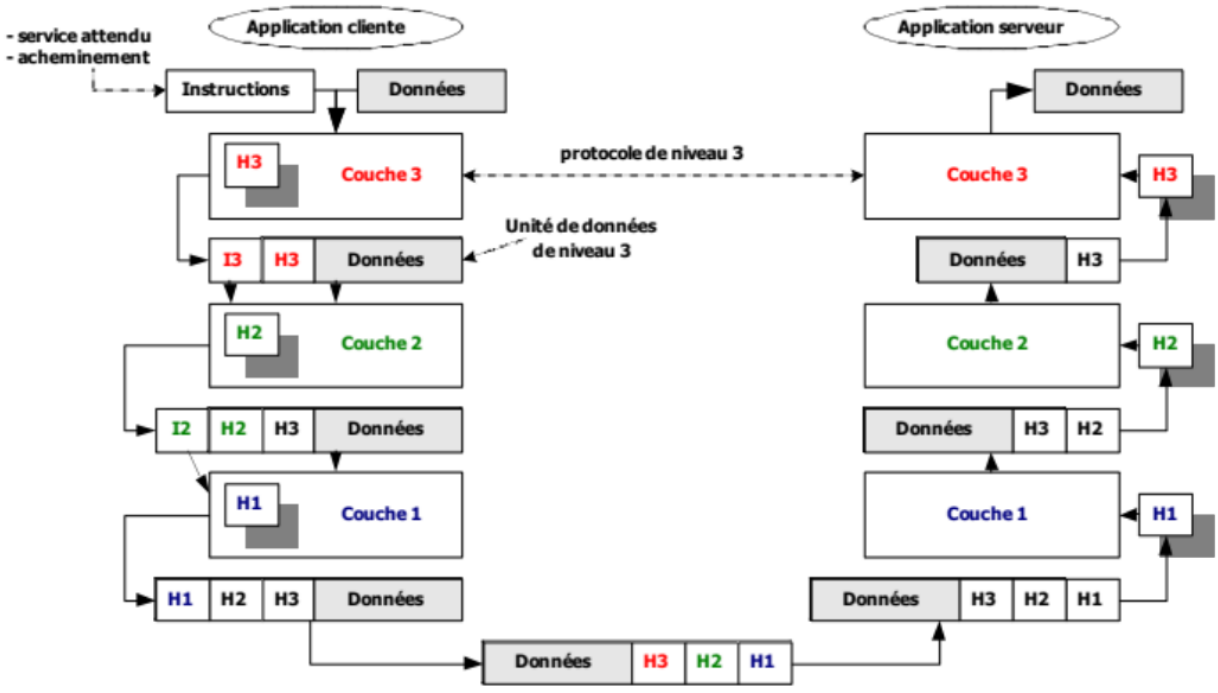

Figure 19: Principe de l'architecture en couches

comment fonctionne n'importe quel réseau.

Vous avez d'un côté, à gauche sur la figure, une application cliente qui va faire une demande vers une application serveur, à droite sur la figure. Le client, un navigateur Web par exemple, constitue une requête qui contient les données à transmettre, celles qui se trouvent en gris C'est pour leur transmission qu'un appel est fait au réseau. Typiquement, les données vont contenir un fichier ou des informations qui sont nécessaires pour que le serveur, l'application serveur, comprenne ce qu'elle doit faire. Et pour transmettre ces données au serveur, on va faire appel ici à trois protocoles qui sont implémentés dans trois couches : la couche une, deux et trois. Il s'agit ici d'un modèle simplifié en trois couches, comme on l'a vu précédemment. Mais ce qu'on va décrire est valable pour un nombre de couches qui pourrait être supérieur ou inférieur. Le principe est que chaque couche fait appel à la couche d'en dessous pour réaliser un service qui est nécessaire pour l'acheminement global des données.

Donc ici les données sont préparées par l'application cliente qui fait appel à la couche trois à l'aide d'instructions qui lui sont transmises pour réaliser le protocole applicatif, par exemple le protocole HTTP. Et pour réaliser ce protocole, on a besoin de définir l'entête (header en anglais) du protocole, c'est-à-dire le H3 (Header de niveau 3). Par exemple, l'entête HTTP doit contenir les informations nécessaires pour demander la page Web et si besoin transmettre les informations qui ont été saisies dans un formulaire de la page Web. Cet entête H3 est donc fabriqué sur l'émetteur par l'application cliente, et va être lu par la couche trois de l'autre côté, c'est-à-dire le récepteur soit l'application serveur qui va répondre à la demande du client exprimée dans H3 auquel les données fournies par l'application (en gris) sont associées. Ses données contiennent par exemple les informations qui ont été saisies dans la page Web.

Une fois que l'entête a été fabriquée pour réaliser la fonction de la couche trois, on procède à l'encapsulation. Cela consiste à ajouter l'entête H3 aux données à transférer (en gris) que nous notons D3 et à les regrouper sous forme d'une unité de données de niveau trois qui vont devenir les données de la couche 2 soit D2. Ainsi, `D2=H3+D3`. Ces données D2 sont transmises à la couche d'en dessous, ici la couche 2. Ainsi, l'en-tête de niveau trois et les données de niveau trois deviennent les données de niveau deux. Cet ensemble est alors transmis à la couche d'en dessous, qui peut par exemple réaliser la fonction de transport en utilisant le protocole IP. Pour acheminer les données jusqu'à leur destination finale (le serveur), cette couche doit également ajouter son propre entête (H2) contenant les adresses IP source et destination. Ce paquet de données est alors transmis via des équipements intermédiaires (les routeurs) qui utilisent l'adresse IP destination pour trouver le chemin le plus adéquat. De nouveau, il y a encapsulation : l'entête H2 et les données de niveau deux sont regroupées en un seul paquet de données de niveau un, qui est transmis à la couche une. Cette dernière peut elle aussi ajouter son entête (par exemple, le protocole Ethernet) afin de transférer le signal depuis la carte réseau de l'application cliente jusqu'à celle de l'application serveur si aucun équipement intermédiaire n'est traversé. Les adresses MAC identifiant les cartes réseau doivent être ajoutées dans H1 pour déterminer la carte réseau de destination.

En résumé, chaque protocole exécute un algorithme et ajoute des informations dans un entête qui sera transmis de l'émetteur au récepteur. Ces informations permettent de réaliser les algorithmes et sont encapsulées côté émetteur et décapsulées côté récepteur. Chaque couche fait appel à la couche d'en dessous lors de l'émission et à celle d'au-dessus lors de la réception, une fois que son entête a été lu et que son algorithme a été exécuté.

Sur le schéma présenté, on retrouve les instructions I3 et I2 qui déclenchent l'appel à la couche d'en dessous. Par exemple, les sockets (interface logicielle permettant de transférer des données depuis l'application vers la couche transport du réseau) se situent entre le protocole HTTP et le système d'exploitation. Elles permettent de déposer les données de l'application dans un buffer (zone mémoire) afin de les transmettre avec l'entête de la couche trois à la couche deux. Autre exemple d'instruction : lorsqu'un téléphone sonne, cela signifie qu'une personne essaie de vous appeler et vous avez le choix de décrocher ou non. L'instruction déclenche ou non la prise en compte de cet appel.

En réception, les instructions permettent de passer les données de la couche N à la couche N+1 en enlevant l'entête de la couche N qui a été lu par la couche N. Ainsi, on retrouve deux types de dialogue sur ce schéma. Un dialogue horizontal virtuel (flèches en pointillés) entre deux couches de même niveau, représentant l'échange de l'entête qui permet de réaliser le protocole en lui-même. Un dialogue vertical entre chaque

couche, représenté par les flèches pleines et les instructions, indiquant que chaque couche appelle la couche d'en dessous. Les entêtes s'ajoutent les unes aux autres et circulent sur les différentes liaisons, constituant le chemin réel des données.

Pour résumer, la couche N+1 fait une demande de service à la couche N en lui passant un message qui contient l'entête et les données de la couche N, ce qui constitue l'encapsulation et devient les données de la couche N-1. On peut écrire cela sous la forme : les données de la couche N sont l'entête de la couche N+1 + les données de la couche N+1 soit `Dn = Hn+1 + Dn+1`.

Nous allons illustrer ce fonctionnement par un exemple de protocole qui permet de transférer de manière fiable un fichier, c'est-à-dire sans erreur, sans perte ni duplication et dans l'ordre. Ce protocole est assez simple car nous avons des hypothèses fortes sur le réseau qui est sans perte et FIFO, c'est-à-dire qu'il ne crée pas de désordre et que les paquets arrivent dans l'ordre. Cependant, des erreurs peuvent se produire. Nous allons donc voir comment effectuer un transfert fiable d'un fichier en gérant uniquement les erreurs. Si le fichier est volumineux, il est préférable de le couper en morceaux, car la probabilité qu'une erreur se produise augmente avec la taille du fichier et en outre, les mémoires sont limitées dans les protocoles, en particulier dans les cartes réseaux.

Voici l'algorithme de ce protocole :

```
- envoyer le fichier en une succession de paquets
- envoyer un « checksum » pour chaque paquet
- contrôler le checksum sur le récepteur et renvoyer un message OK ou Not-OK à l'émetteur
- l'émetteur attend le OK ou Not-OK avant de demander le transfert du paquet suivant
- l'émetteur attend le dernier message OK avant de clore la connexion
- si Not-OK pour un paquet, re-transférer le paquet
```

Pour détecter les erreurs, nous utilisons un mécanisme appelé checksum ou somme de contrôle, également connu sous le nom de CRC ou FCS selon les protocoles. Le checksum est un élément calculé sur l'émetteur sur les données avant l'envoi et envoyé au récepteur. Lorsque les données arrivent en réception, nous refaisons le même calcul sur les données reçues et comparons le checksum avec celui des données avant l'envoi pour déterminer s'il y a eu une erreur pendant le transfert. S'il n'y a pas d'erreur, nous envoyons un message d'acquittement OK, sinon nous envoyons un message d'erreur Not-OK. Pour chaque morceau de fichier, nous calculons un checksum et vérifions son intégrité à l'arrivée. C'est ce qu'on appelle la détection d'erreur.

Pour chaque morceau du fichier, l'émetteur doit attendre l'acquittement positif ou négatif avant d'envoyer le morceau suivant du fichier. S'il reçoit un acquittement positif, il passe au morceau suivant, sinon il retransmet le paquet qui a subi une erreur. Ce protocole, appelé "send and wait", garantit la fiabilité du transfert du fichier en ajoutant un checksum qui circule dans l'en-tête du paquet, en utilisant des messages d'acquittement spécifiques et en mettant en place un algorithme de retransmission lorsque cela est nécessaire.

Il est important de souligner que ce protocole fait appel à un autre protocole pour acheminer chaque morceau du fichier à travers le réseau en utilisant des liaisons précises et qu'un autre protocole est mis en œuvre sur chaque liaison, illustrant ainsi la notion d'architecture en couches. Le protocole spécifie la syntaxe, le format et la signification de chaque message, comme le message d'acquittement positif ou négatif et l'algorithme à suivre en cas de réception de l'un de ces messages. En somme, le protocole définit les règles à suivre pour assurer la fiabilité du transfert du fichier.

Il existe deux types de protocoles en fonction de leur portée : les protocoles de **bout en bout** et les protocoles **point-à-point**. Les protocoles de bout en bout s'exécutent uniquement aux extrémités du réseau, c'est-à-dire sur les équipements terminaux de l'utilisateur, tels que les ordinateurs, les téléphones et les tablettes. Ils sont généralement utilisés par les applications, comme les navigateurs web ou les serveurs web, ou par les téléphones pour passer des appels. Les protocoles point-à-point, en revanche, s'exécutent également sur les équipements intermédiaires du réseau, tels que les répéteurs, les bornes WiFi, les box, les commutateurs, les routeurs et les pare-feux. Ils ont donc une existence à l'intérieur du réseau alors que les protocoles de bout en bout n'en ont pas.

Il est important de faire cette distinction entre protocoles de bout en bout et protocoles point-à-point car

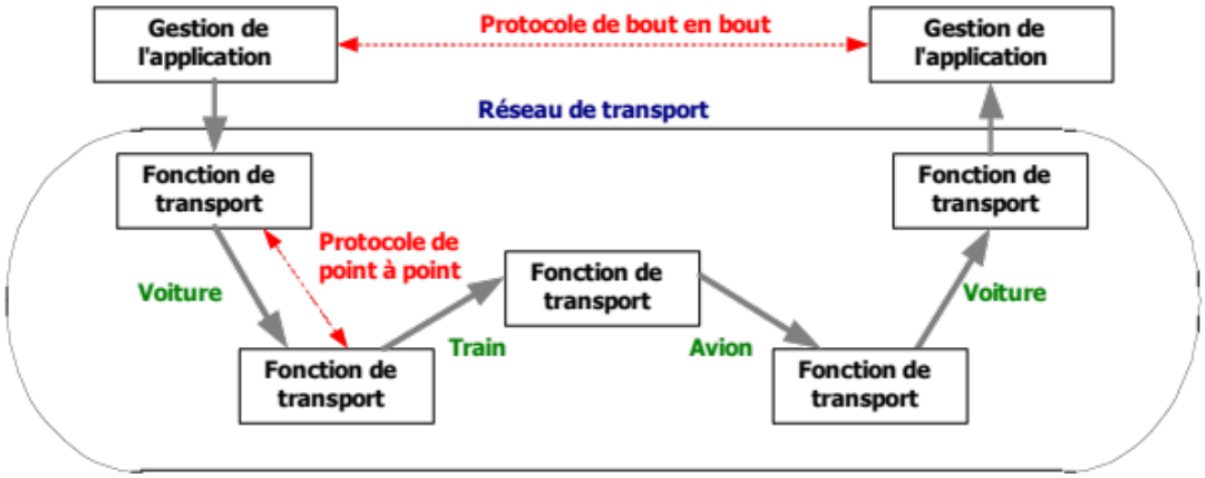

Figure 20: Bout-en-bout vs point-à-point

chaque équipement intermédiaire va mettre en œuvre un sous-ensemble des protocoles de l'architecture complète. Par exemple, sur Internet, les protocoles de bout en bout sont les protocoles applicatifs tels que HTTP, ainsi que TCP et UDP, tandis que le protocole IP est point-à-point car il s'exécute sur tous les routeurs traversés. Il est donc important de comprendre où s'exécutent ces différents protocoles pour comprendre leur portée et leur fonctionnement.

1.3 Le modèle de référence - OSI

Nous allons maintenant étudier le modèle OSI, un modèle de référence théorique qui décrit le fonctionnement d'une architecture réseau dans son ensemble et définit le rôle de chaque couche qui peut constituer une architecture de réseau.

Le modèle OSI décrit une architecture réseau constituée de sept couches, allant de la couche la plus basse (couche physique) à la couche la plus haute (couche application). Chaque couche réalise une fonction spécifique dans le réseau. La **couche physique** s'occupe de la transmission d'une séquence binaire sur un support de transmission. La **couche liaison** s'occupe de l'interface entre deux cartes réseau et de la gestion du dialogue sur une liaison multipoint. Pour cela, elle utilise les adresses des cartes réseaux, appelée adresses physiques ou adresses MAC pour Medium Access Control. La **couche réseau** (couche trois) a pour rôle de déterminer par où les paquets vont passer pour aller d'un point à un autre dans le réseau. Elle utilise pour cela les adresses spécifiques à chaque architecture réseau. Il s'agit des adresses IP sur Internet. Il s'agit du numéro de téléphonne dans le réseau téléphonique. Les trois premières couches sont des couches point-à-point qui s'exécutent tant sur les équipements terminaux que sur les équipements intermédiaires, tandis que les quatre couches supérieures sont de bout en bout et ne s'exécutent que sur les équipements terminaux.

La **couche de transport** du modèle OSI assure la fiabilité des transferts, garantissant ainsi que tout ce qui arrive à destination correspond exactement à ce qui a été émis initialement. Cette couche veille à ce qu'il n'y ait pas d'erreurs, de pertes, que les paquets arrivent dans l'ordre et sans duplication. Au niveau de l'Internet, c'est la couche TCP qui est chargée de cette tâche.

La **couche de session** permet de créer des points de reprise pour les applications qui en ont besoin, comme pour un transfert de fichiers, ou d'interrompre et reprendre le transfert à un moment donné. Elle permet également de différencier les applications utilisant la couche de transport, en utilisant des numéros de port dans les protocoles TCP ou UDP qui permettent d'identifier le processus applicatif ciblé par l'émission ou

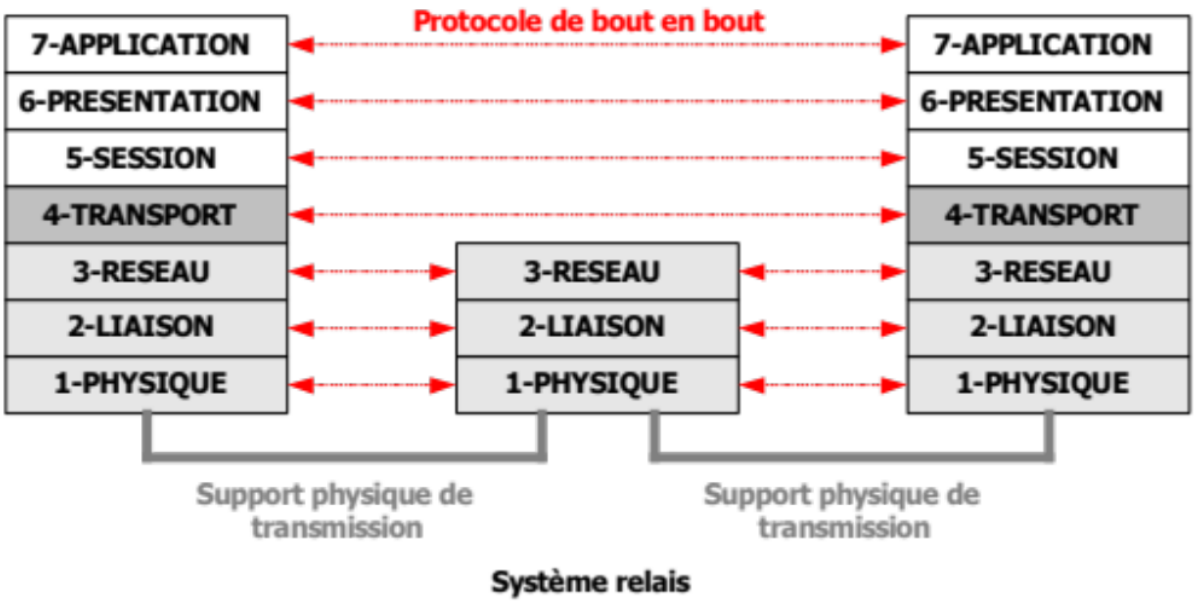

Figure 21: Modèle OSI

la réception du message. La **couche de présentation** s'occupe de transformer les données provenant de l'application, comme pour des raisons de sécurité en utilisant le chiffrement par le protocole SSL sur Internet. Enfin, la **couche d'application** est celle qui gère le dialogue entre le processus client et le processus serveur permettant de rendre le service attendu par cette application. Par exemple, le protocole HTTP sur Internet permet de demander une page Web à un processus serveur qui héberge cette page Web.

Voici donc les sept couches du modèle OSI. Ensuite, nous retrouvons le vocabulaire déterminé précédemment. La **trame** est l'unité de transmission au niveau de la carte réseau, ou plus précisément, de la couche de liaison. Le **paquet** est quant à lui l'unité de transmission au niveau de la couche de réseau, nous parlons alors de paquet IP. Enfin, pour les couches supérieures, nous utilisons le terme générique de **message**.

Le modèle OSI décrit chaque couche qui ajoute son en-tête, de la couche sept jusqu'à la couche deux. La couche une n'ajoute pas d'en-tête car elle ne fait que transmettre une séquence binaire sur un support de transmission, ce qui ne nécessite pas l'ajout d'un en-tête. En revanche, dans la couche deux, nous ajoutons également un **enqueue**, ou **trailer** en anglais, qui permet de stocker le checksum effectuant la détection d'erreur au niveau des trames transmises. Pourquoi le checksum se retrouve-t-il à la fin de la trame ? Cela est dû au fait que nous pouvons dans une carte réseau commencer à transmettre l'en-tête et les données avant d'avoir terminé le calcul du checksum. C'est donc seulement après avoir vu passer toutes les données dans la carte réseau que le calcul du checksum est terminé et ajouté à la fin de la trame.

Le modèle OSI décrit également les mécanismes généralement présents dans toutes les architectures de réseau. Par exemple, le **contrôle de flux**, qui permet à un récepteur de ralentir les émissions d'un émetteur lorsque les réceptions ne sont plus possibles, voire même d'ordonner à un émetteur de cesser d'envoyer, afin d'éviter les pertes. Le **maintien en séquence** garantit que les données arrivent dans l'ordre, tandis que l'**accusé de réception** permet d'acquitter ou non les messages reçus. L'acquittement positif consiste à dire : "oui j'ai bien reçu" ou à l'inverse, l'acquittement négatif exprime "Non, je n'ai pas bien reçu". C'est un mécanisme qu'on va retrouver dans la plupart des protocoles.

L'adaptation de la taille des unités de données est un phénomène qui se produit dans chaque couche. En général, chaque protocole a une taille maximale de messages et donc il faut respecter cette taille maximale lors de la transmission de l'unité de données à la couche inférieure. Par conséquent, chaque protocole doit adapter sa propre taille de données à cette taille maximale, c'est-à-dire que, si nécessaire, il faudra découper

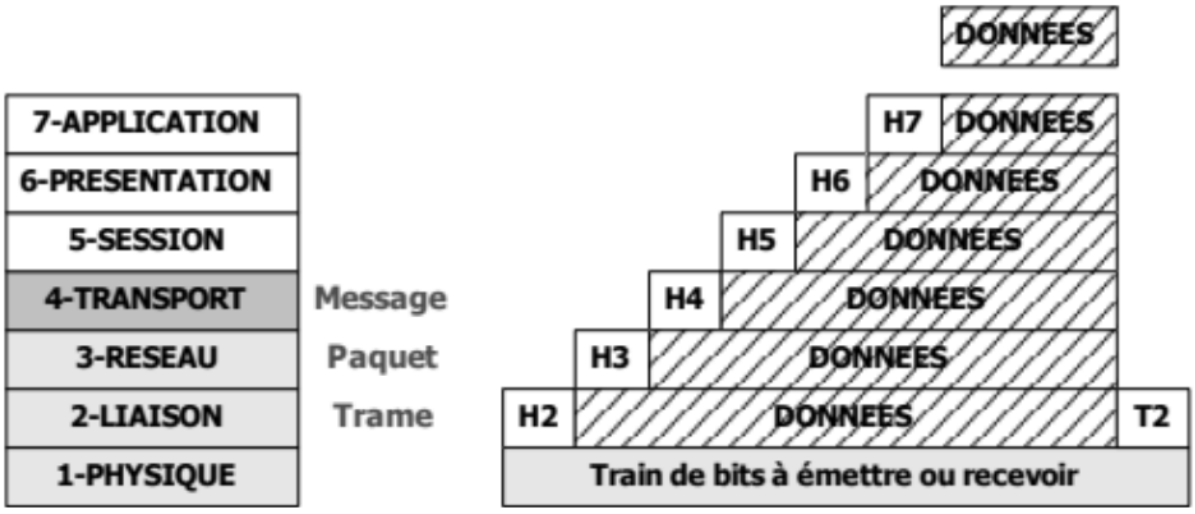

Figure 22: encapsulation dans le modèle OSI

le message en plusieurs parties si la taille maximale est dépassée. C'est par exemple ce que fait le protocole IP avec la fragmentation sur Internet. C'est-à-dire que le protocole IP découpe les paquets qui sont trop gros lorsque les trames ont une taille maximale qui ne permet pas de transporter ces paquets trop volumineux.

La détection d'erreur permet, grâce au checksum, de vérifier en réception que les données reçues sont erronées ou non, et donc, en fonction du résultat, de décider ou non de poursuivre la transmission voire de faire des retransmissions si la fiabilité est mise en œuvre.

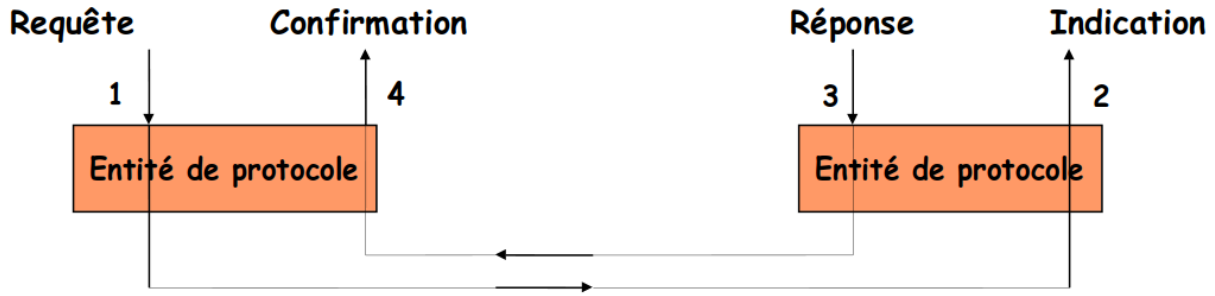

Figure 23: Primitive de service

On retrouve également dans les différents protocoles la notion de **mode connecté** et de **mode non connecté**. Le mode connecté signifie que lorsqu'un protocole souhaite transmettre des informations à l'autre partie, il doit d'abord demander à cette dernière si elle est d'accord pour recevoir les données. Il y a donc une phase de connexion qui consiste à envoyer une requête de l'émetteur au récepteur en lui demandant "Es-tu d'accord pour te échanger avec moi ?". Le récepteur est notifié et il répond soit "d'accord", soit "pas d'accord". Ce n'est qu'après confirmation de la connexion que les données peuvent être envoyées entre l'émetteur et le récepteur.

Un exemple typique de mode connecté est une communication téléphonique. On compose un numéro et on appelle, ce qui correspond à une demande de connexion. Le téléphone sonne et la seule indication est que la personne appelée réponde ou non. Cela permet d'entrer en communication ou non.

Sur Internet, la plupart des protocoles fonctionnent en mode non connecté. Le seul protocole parmi tous ceux que l'on va voir qui est en mode connecté est le protocole TCP. Pourquoi ? Parce qu'il assure la fiabilité et que pour mettre en œuvre l'algorithme permettant d'assurer cette fiabilité, il a besoin de cette notion de connexion.

Le mode non connecté, à l'inverse, consiste à envoyer un message sans demander au récepteur à l'avance s'il est d'accord ou non pour recevoir le message. C'est donc équivalent à l'envoi d'un SMS, d'un courrier électronique ou d'une lettre par la poste. Le paquet est transmis sans aucune garantie, sans demander à l'avance si le paquet peut être envoyé ou non. Sur Internet, la plupart des protocoles fonctionnent de cette manière, à l'exception de TCP.

En mode non connecté, il n'est pas garanti que le récepteur acceptera ou même sera en mesure de recevoir les données. Dans ce cas, on est généralement dans un mode **best effort**, c'est-à-dire que l'on fait de notre mieux sans être sûr que les données arriveront, ni que le récepteur pourra les recevoir. Par conséquent, il n'y a aucune garantie à l'avance sur la bonne transmission des informations.

1.3.1 Une version simplifiée du modèle OSI

Nous allons maintenant décrire l'architecture TCP/IP, qui est l'ensemble des protocoles permettant à Internet de fonctionner et qui repose uniquement sur quatre couches, contrairement au modèle OSI qui en compte sept. Cependant, même sur Internet, certaines applications comme **NFS** respectent la présentation du modèle OSI.

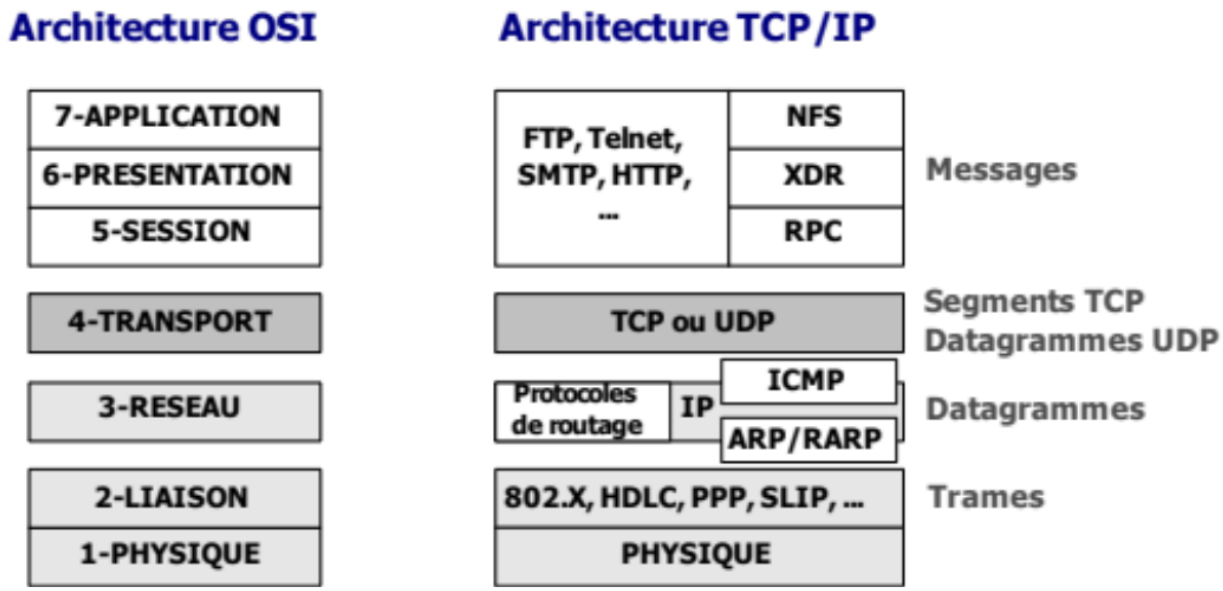

Figure 24: Architecture TCP/IP

Voici une version de l'architecture TCP/IP et la correspondance avec les sept couches du modèle OSI. L'application **NFS** (Network File System) repose sur les sept couches, puisqu'elle utilise les protocoles XDR et RPC au niveau de la couche présentation et session. Cette application est très utilisée car elle permet à un utilisateur d'accèder à ses fichiers lorsqu'ils sont hébergés sur un serveur situé dans le réseau.

Sur Internet, on retrouve généralement dans la couche application les protocoles qui permettent à chaque application de décrire comment s'échanger des données. On retrouve par exemple le protocole **FTP** (File Transfer Protocol) pour le transfert de fichiers, le protocole **HTTP** (HyperText Transfer Protocol) pour le Web, le protocole **Telnet** pour les connexions à distance, ou encore le protocole **SSH**, qui est sécurisé. Il y a le protocole **SMTP** (Simple Mail Transfer Protocol) pour l'envoi des courriers électroniques et le

protocole **DNS** (Domain Name Service) pour la résolution des noms de serveurs. Bien sûr, il existe bien d'autres applications qui ont chacune leur propre protocole. **Les protocoles applicatifs s'exécutent directement dans l'application** c'est-à-dire dans le processus client (par exemple le navigateur Web) et le processus serveur (par exemple le serveur Web).

Au niveau de la couche quatre du modèle OSI, c'est-à-dire la couche transport, on a le choix entre **TCP** (Transmission Control Protocol) ou **UDP** (User Datagram Protocol). TCP réalise la fiabilité, comme le décrit le modèle OSI, tandis que UDP ne le fait pas. UDP est donc utilisé par les applications qui ont besoin de rapidité ou qui n'ont pas besoin de fiabilité. Typiquement, les applications multimédia utilisent UDP. UDP ne fait qu'ajouter les numéros de port qui permettent de déterminer pour quelles applications, pour quel processus client et serveur, les données sont émises ou reçues. UDP fait la détection d'erreur à l'aide d'un checksum mais ne fait pas de retransmissions.

Le protocole **IP** (Internet Protocol) permet de faire l'acheminement des paquets sur Internet, c'est-à-dire de trouver le chemin par lequel ils doivent passer. La couche transmission, la couche la plus basse, est la superposition des couches un et deux du modèle OSI : la couche physique et la couche liaison. La couche transmission correspond au protocole des cartes réseau comme **Ethernet** (norme 802.3), le **Wifi** (normes 802.11) ou encore **Bluetooth** (normes 802.15). Elle assure la transmission des trames qui circulent sur chaque liaison en utilisant les **adresses MAC** des cartes réseaux et en transformant la trame en une séquence binaire qui elle-même va devenir un signal émis ou reçu par chaque carte réseau. La séquence binaire transmise contient tous les entêtes de tous les protocoles traversés ainsi que les données provenant de l'application.

Les protocoles des couches 3 et 4 comme TCP, UDP, ICMP, IP, ARP s'exécutent dans le système d'exploitation c'est-à-dire que l'implémentation du protocole est intégrée dans Windows, Linux ou MacOS sur un ordinateur, dans iOS ou Androïd sur un téléphone, dans Cisco IOS sur un routeur Cisco, etc.

Les protocoles des couches 1 et 2 comme Ethernet (802.3), le Wifi (802.11), Bluetooth (802.15), HDLC s'exécutent dans la carte réseau c'est-à-dire que l'implémentation du protocole est embarquée dans une carte réseau intégrée à un ordinateur, un téléphone ou un équipement intermédiaire (routeur, commutateur, répéteur, borne Wifi, etc).

On peut voir sur cette architecture que d'autres protocoles interviennent au niveau de la couche 3. Il y a les **protocoles de routage dynamique** qui permettent de construire les tables de routage sur Internet de manière automatique, le protocole **ICMP** qui permet de faire du **ping** et de tester les échanges de messages entre deux ordinateurs ou entre un ordinateur et un équipement intermédiaire. Enfin, le protocole **ARP** (Address Resolution Protocol), également situé au niveau de la couche trois, permet de faire le lien entre les adresses IP utilisées pour trouver le chemin sur Internet et les adresses **MAC** (Medium Access Control), qui sont les adresses des cartes réseau et qui permettent de savoir vers quelle carte réseau le signal doit être envoyé.

Comme nous l'avons vu avec le modèle OSI, les couches un, deux et trois sont des couches à point, c'est-à-dire qu'elles peuvent s'exécuter sur des équipements intermédiaires. Cette analogie existe également sur Internet. Les protocoles qui sont de bout en bout sont les protocoles applicatifs ainsi que TCP ou UDP. Les autres protocoles peuvent s'exécuter sur des équipements intermédiaires tels que des routeurs ou des commutateurs. Les routeurs exécutent le protocole IP pour faire le routage, tandis que les protocoles des cartes réseau et les commutateurs n'exécutent que les protocoles des cartes réseau.

Pour terminer, voici un exemple d'une requête HTTP lancée à partir d'un navigateur web dans l'URL. Dans l'URL, on peut voir le nom du serveur web, ici www.univ-lyon.fr. Cela déclenche deux requêtes qui partent du navigateur web : tout d'abord une requête DNS pour obtenir l'adresse IP du serveur web. Elle est nécessaire pour déterminer le chemin pour atteindre le serveur web. Cette requête DNS utilise le protocole UDP au niveau de la couche transport et est envoyée à un serveur DNS qui répond en renvoyant l'adresse IP du serveur web.

Ensuite, le navigateur web construit une requête HTTP pour demander la page web indiquée dans l'URL et l'envoie au serveur web. La requête HTTP est envoyée au serveur web en utilisant le protocole TCP, ce qui

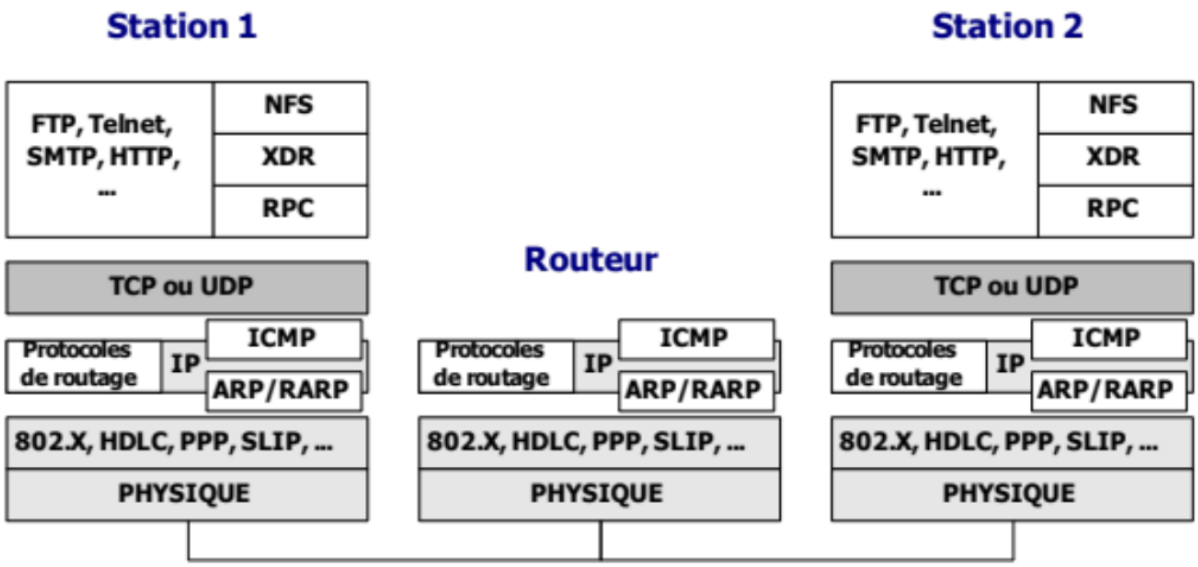

Figure 25: Interconnexion dans TCP/IP

Figure 26: Exemple d'une requête HTTP

crée une connexion entre le navigateur web et le serveur web. Cette connexion TCP permet alors de réaliser tous les échanges de manière fiable entre le client et le serveur web, notamment pour renvoyer la page web demandée.

Comme indiqué précédemment, HTTP et DNS sont intégrés au navigateur Web et sont exécutés par lui. TCP, UDP et IP sont dans le système d'exploitation de la machine qui exécute le navigateur Web. Ethernet s'exécute dans la carte réseau de l'ordinateur par laquelle le signal contenant la requête du navigateur va être envoyée.