Réseaux informatiques

Session de Remise à Niveau, Télécom Paris

Août-septembre 2021

Ken CHEN

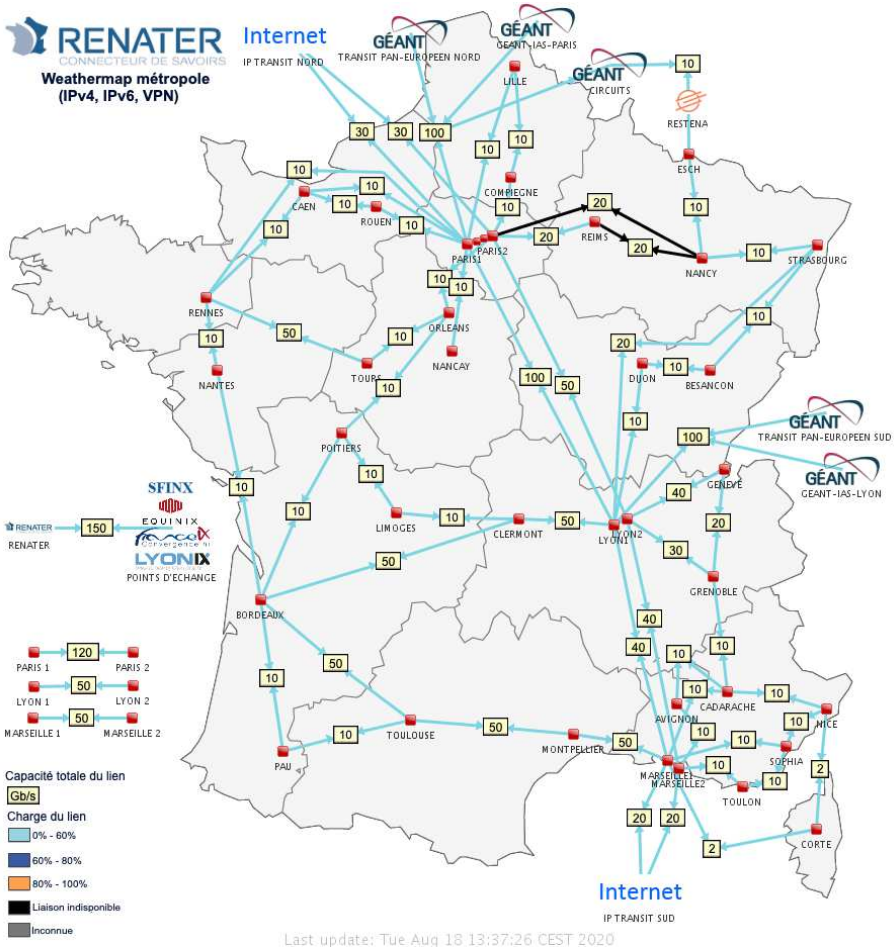
ken.chen@univ-paris13.fr

Objectif du cours

+ Acquérir une vision d'ensemble sur les réseaux
  - **architecture**, **protocoles**, **technologies**
+ Comprendre
  - les **concepts** fondamentaux des réseaux et des services offerts
  - La **logique** de fonctionnement des protocoles
+ Avoir un 1er aperçu des technologies et protocoles dominants
  - **Ethernet**, **INTERNET**
  - Les (nombreux) **mots-clés**
+ *En marge, être au courant des enjeux sous-jacents*
  - *performances, sécurité, conso. énergie, etc.*

Exemple RENATER

+ Infrastructure RENATER (https://www.renater.fr/)
  + Réseau National de télécommunications pour la Technologie l'Enseignement et la Recherche



Crédit: https://www.renater.fr/weathermap/weathermap_metropole

Un nœud du réseau

+ RAP (Réseau Académique Parisien)
  - Un des (sous-) composants de Renater
+ Ci-dessous, un noeud (PoP) de ce réseau

Crédit (capture d'écran) : http://www.rap.prd.fr/generalites/technique.php

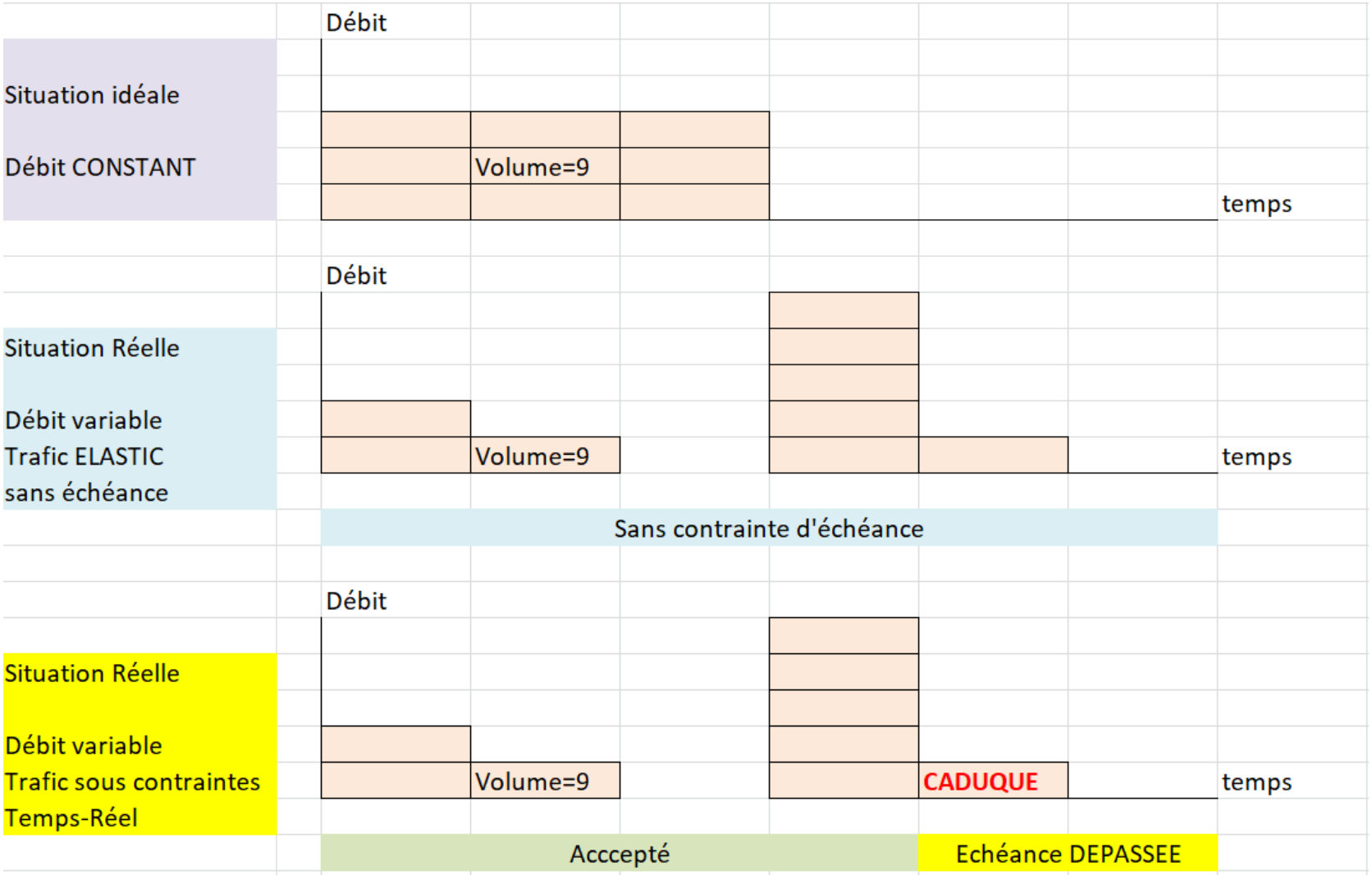
Présentation

+ Réseaux de Données
  - Réseaux informatiques (en anglais : *Computer networks*)
  - *Internet (langage courant)*
+ Principales caractéristiques
  - Réseau : structure maillée pour interconnexion des sites
    + Accès, routage, contrôle de transfert de bout-en-bout, etc.
  - Transport de données
    + Particularité : “matériel virtuel” : traitement/reproduction à volonté
+ Structures ayant des similitudes
  - Le réseau routier : **transport** (similitude) de matériel physique (différence)
  - Structures sociétales (Administration…) **interactions** soumises aux **règles**

Applications et flux de données

+ Partage de ressources à **distance (*remote*)**
  - CPU (rlogin, cloud), Stockage (ftp, p2p, cloud), Information (mél, web)
+ Utilisations
  - Coopération (téléconférences, etc.)
  - Diffusion d'info (Web, etc.)
  - Télécommande (surveillance, etc.)
  - ***Cloud, Internet des objets, etc. etc.***
+ Les flux à transporter et leur classification
  - Données **« élastiques »**
    + **Tolérance zéro** aux erreurs , pas de contraintes temporelles
  - ***Échanges transactionnels***
    + ***Promptitude***
  - Données **Temps-Réel** (Multimédia: Voix, Vidéo, etc.)
    + **Contraintes temporelles**, une certaine tolérance (limitée) aux erreurs

Flux: Elasticité, Contrainte Temps



Eléments constituants

+ **Infrastructures**
  - Accès, liaison, nœud, réseaux (inter)-nationaux
+ **Entités terminales**
  - PC, Serveurs, Imprimantes, *Smartphone*, etc.
+ **Applications**
  - Productrices/consommatrices d'information à distance
+ **Services**
  - Fonctionnalités fournies par une certaine partie du réseau
+ **Protocoles**
  - Automates (logiciels) coopérant pour gérer les flux d'info, en vue d'offrir un **service**, souvent dans un cadre **architectural** donné

Eléments physiques

+ Support de transmission
  - paire torsadée, *coaxial*, fibre optique, hertzien
+ Topologie
  - bus (bi/uni-directionnel), anneau, **étoile, arbre, maillage**
+ Modulation et codage
  - Codages NRZ, RZ, Bi-phase, etc.
  - bande de base, transposition en fréquence,
+ Interface
  - Spécification mécanique, électrique et fonctionnelle
  - RJ45, interface radio, etc
+ Equipement
  - Carte, Commutateur, routeur, serveur, etc.

Classification des réseaux

+ Réseaux étendus (public) (WAN) :
  - Couverture (inter)nationale,
  - Liaisons à très hauts-débits (Gbps … Tbps)
  - Gérés par des opérateurs (e.g. Orange).
  - Débit par utilisateur et/ou communication faible
+ Réseaux locaux (LAN) :
  - Couverture faible (< env. 1 km),
  - Débits élevés (10Mbps -**100 Mbps** - 1Gbps),
  - Gérés par l ’entité utilisatrice.
+ *Réseaux métropolitain (MAN)*
  - Réseaux de collecte, boucle locale

Concepts de base

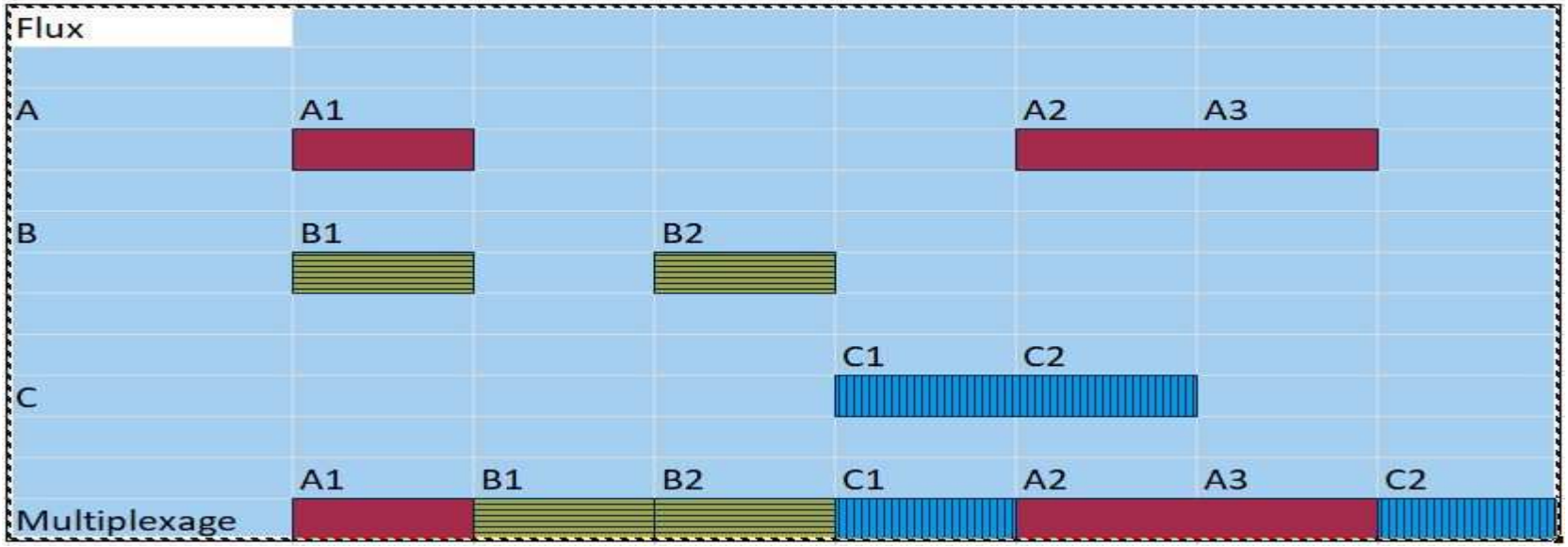
+ Mode de commutation et d'accès
  - Commutation de circuit vs **Commutation de paquet**
  - Médium partagé (**MAC**), **Commutateurs**,
+ Mode de communication
  - Orienté connexion (**circuit virtuel**) vs
  - Sans connexion (**datagramme**)
+ **Architecture** et **protocoles**
  - architecture en **couches** (découpage fonctionnelle), concept de **service**
  - Protocoles (automates dans des systèmes distribués)
+ **Routage** (acheminement dans un réseau étendu)
+ **Contrôle de bout-en-bout**
+ *Autres concepts importants et/ou connexes*
  - *Qualité de service (QoS), Sécurité, Consommation d'énergie, etc.*

Modes de commutation

+ Solution historique : ***commutation de circuit***
  - circuit = (image d'un) lien physique reliant en permanence les 2 correspondants
    + En pratique: un chaînage des liens dédiés durant toute la communication
    + Exemple : RTC (Téléphone) : coûteux
+ Support de communications multiples et simultanées
  - Un lien physique par communication ???
    + Coûteux, non-extensible
  - Commutation de circuit **doit être abandonnée !**
+ **Solution : Commutation de paquets (*packets*)**
  - « atomiser » des communications sous forme des paquets pour rendre possible leur **Multiplexage**

Modes de commutation

+ **Multiplexage : partage d'une même ressource**
  - En l'occurrence : ligne communication (bande passante)
+ Exemple : 3 flux A, B, C
  - Multiplexage sur une seule ligne (bien remplie)
  - Au lieu de 3 lignes (mal remplies)



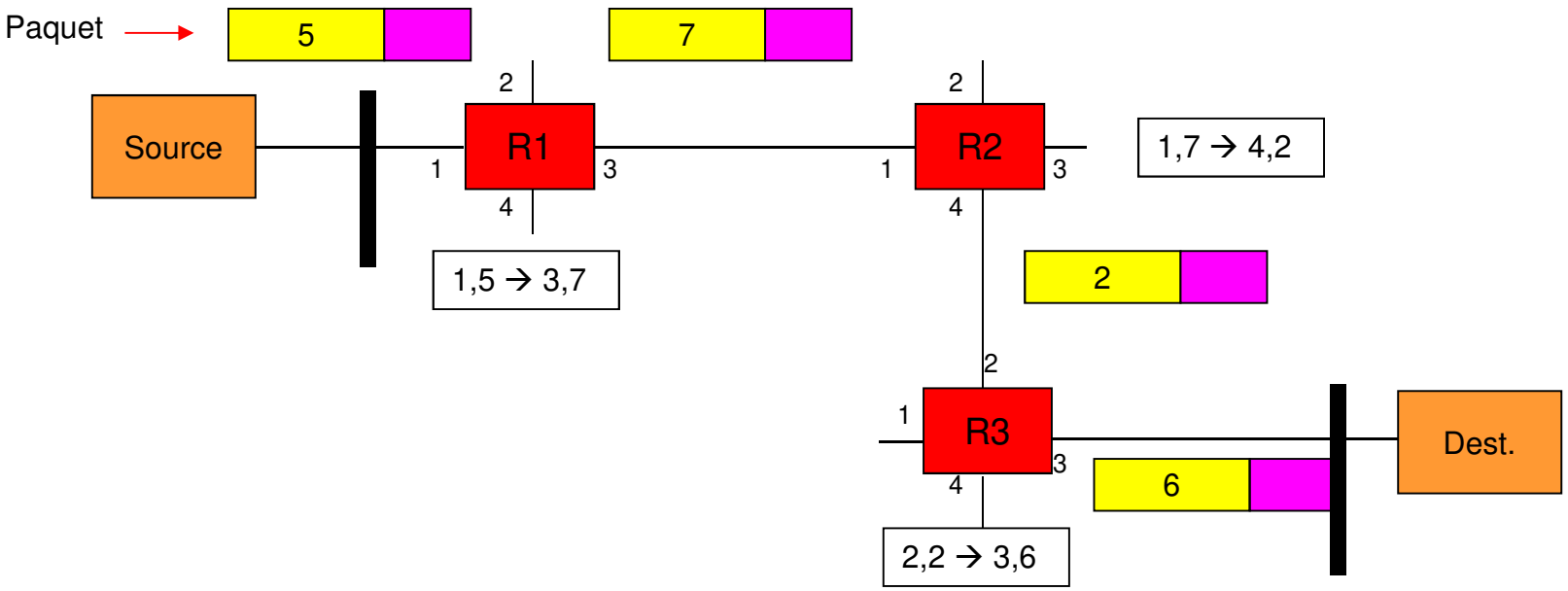
Commutation de paquets

+ Flux d'information: **Découpage** en **paquets**
  - Les paquets issus de différents flux (différentes communications) sont mélangés de manière séquentielle
+ Deux types de découpages
  - **taille variable** : procédé dominant (pour son rendement)
  - *taille fixe et petite (cellule) : procédé quasi caduc aujourd'hui*
+ Avantage :
  - Meilleures utilisation des ressources (multiplexage)
  - Support de communications multiples
+ Problème : perte de l'intégrité du flux
  - Avec commutation de circuit: **le circuit** physique « ***matérialise*** » **la communication** supportée

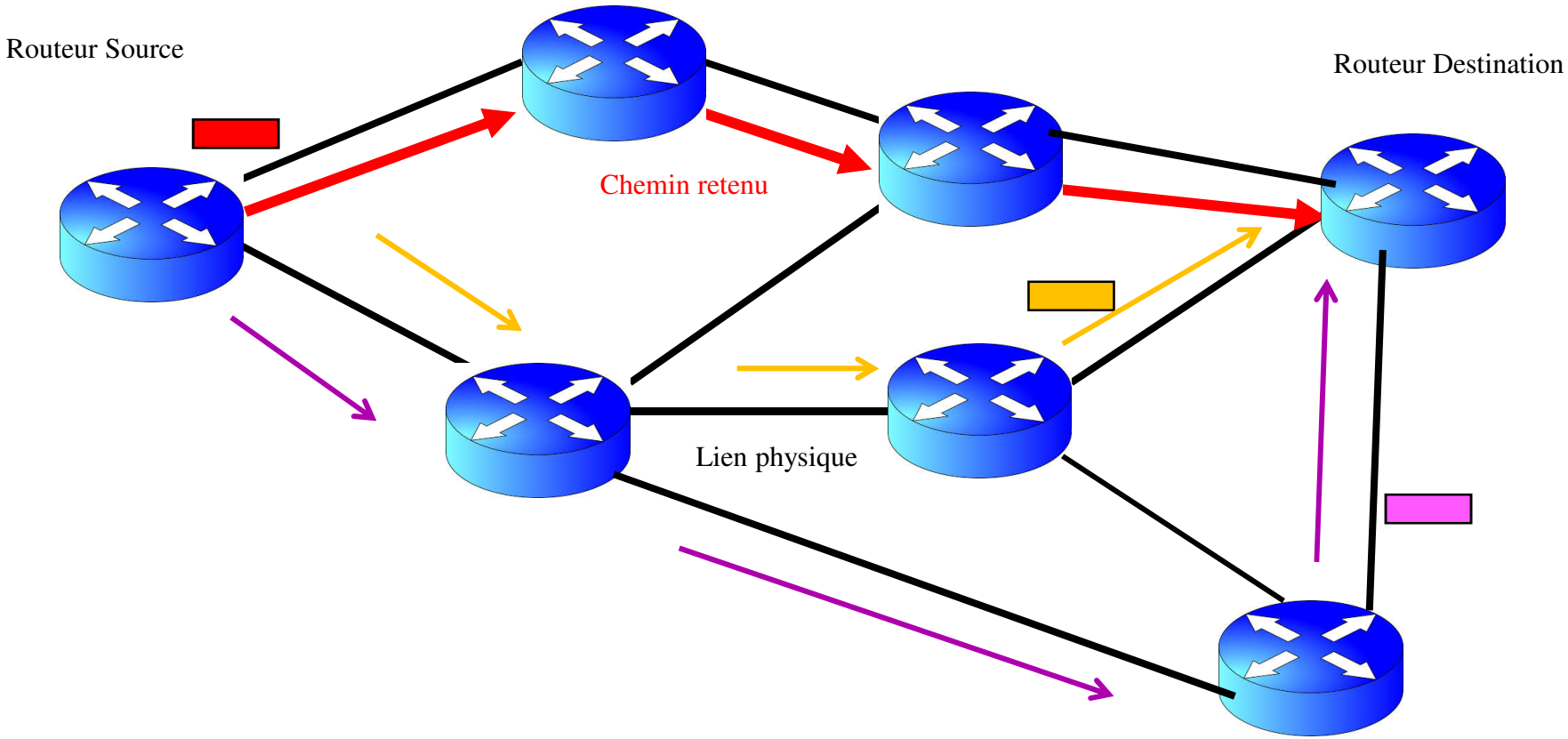
Modes de communication

+ Deux modes de communication
  - **Orienté connexion** vs **sans connexion**
+ Orienté connexion (**circuit virtuel**, un appel téléphone)
  - Possibilité de reconstitution du flux émis
  - Orienté connexion au sens strict : tuyau virtuel idéal
    + sans perte
    + Dans l'ordre (sans désordre, pas de déséquencement)
    + sans duplication
+ Sans connexion (***datagram***, un SMS, *ou une lettre à la Poste*)
  - Absence des garanties précédentes
  - En particulier, les paquets sont tous indépendants les uns des autres.

Circuit virtuel



Datagrammes



Trois paquets suivant chacun son chemin

Un scénario assez rare en pratique, mais toujours ***POSSIBLE***

Variantes

+ Variante de « sans connexion » :
  - Chaque datagram est acquitté
  - Exemple : LLC3
  - Application : transaction fiabilisée
+ Variante de « orientée connexion »
  - Etablissement d'un circuit, mais « brut »
    + **sans** correction en cas de perte (supposé rare), **ni** rétablissement de l'ordre, **ni** filtrage de duplication (supposé quasi inexistants)
    + Contexte : infrastructure physique (réseau commuté) assez fiable et n'introduisant pas de désordre
  - Exemple : MPLS sur une liaison dédiée dans une fibre optique
  - Application : « liaison » virtuelle **dédiée** entre sites et/ou entité distante

Le modèle OSI/ISO

OSI à travers un exemple

+ Un usager (U) se connecte à un serveur Web (S)
+ Interfaces physique (e.g. connecteur RJ45=Ethernet)
+ Interface application (e.g. HTTP)
+ liaison pt-à-pt :
  - le serveur est relié à son routeur par un câble Ethernet
  - le client est relié à son point d'accès via WiFi
+ réseaux : routage
+ contrôle de bout en bout (TCP)
+ autres fonctionnalités (non abordées ici)
  - Synchronisation de flux
  - Conversion de formats

Modèle OSI: exemple

Couches	Usager	Box d'accès		Réseau(x) d'opérateur(s)					Routeur d'accès		Serveur
Programme utilisé	*Chrome*										*Appache*
Application	HTTP										HTTP
Présentation											
Session											
Transport	TCP										TCP
Réseau	IP	IP		IP	IP	IP	IP	IP	IP		IP
Liaison	WiFI	WiFi	PPP	PPP	*Infrastructure*			PPP	PPP	Ethernet	Ethernet
Physique	WiFi	WiFi	ADSL	ADSL				Modem	Modem	RJ45	RJ45
		Onde E-M		ADSL				Liaison Fibre optique OC-48		Ethernet	

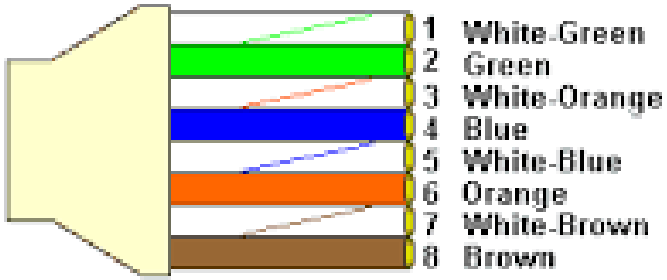
Modèle OSI

+ Open System Interconnection
  - **Modèle de référence (1984)**
  - défini par l ’ISO (*International Organisation for Standardization*)
    + certains aspects sont démodés, mais reste la seule référence
+ Découpage fonctionnel en 7 couches
  - la plus haute = Interface avec les applications
  - la plus base = Interface avec le matériel de la transmission numérique
+ Indépendance entre les couches adjacentes
  - Services et Primitives
  - SDU: Service Data Unit
+ Encapsulation de données :
  - PDU (Protocol Data Unit) = **En-tête** + SDU

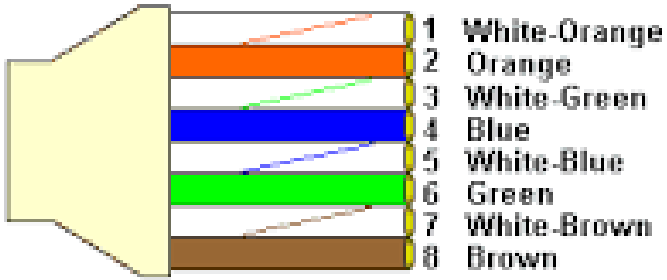
OSI : les 7 couches (*layers*)

+ 7: application
  - interface vers les programmes et/ou utilisateurs
+ 6: présentation (*presentation*)
  - conversion de formats
+ 5: session
  - synchronisation,
+ 4: transport,
  - Transfert de bout en bout, contrôle de **fiabilité**
+ 3: réseau (*network*)
  - échange les données via un réseau maillé
+ 2: liaison de données (*data link*):
  - accès entre noeuds voisins
+ 1: physique (*physic*)
  - Transmission (modulation) d'information sur le médium

PHY: Ethernet : Câble et Prise RJ45

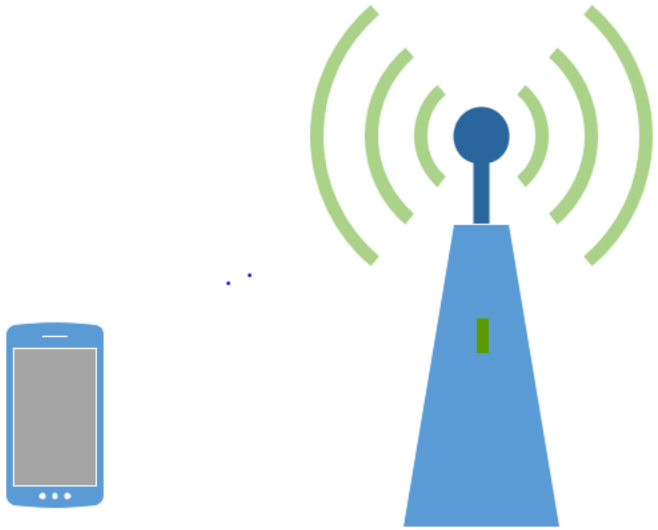


568A CABLE END



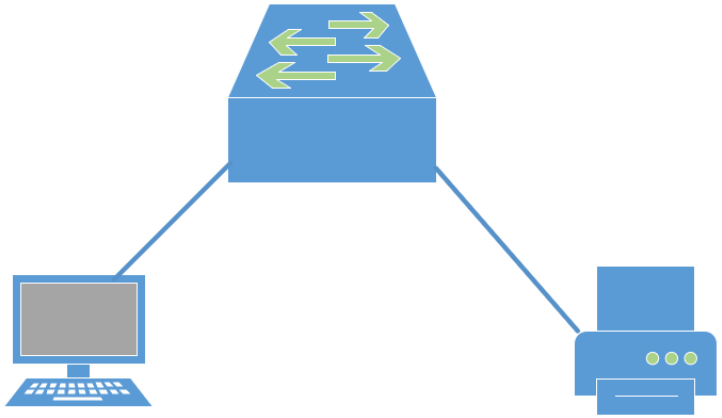
568B CABLE END

Liaison

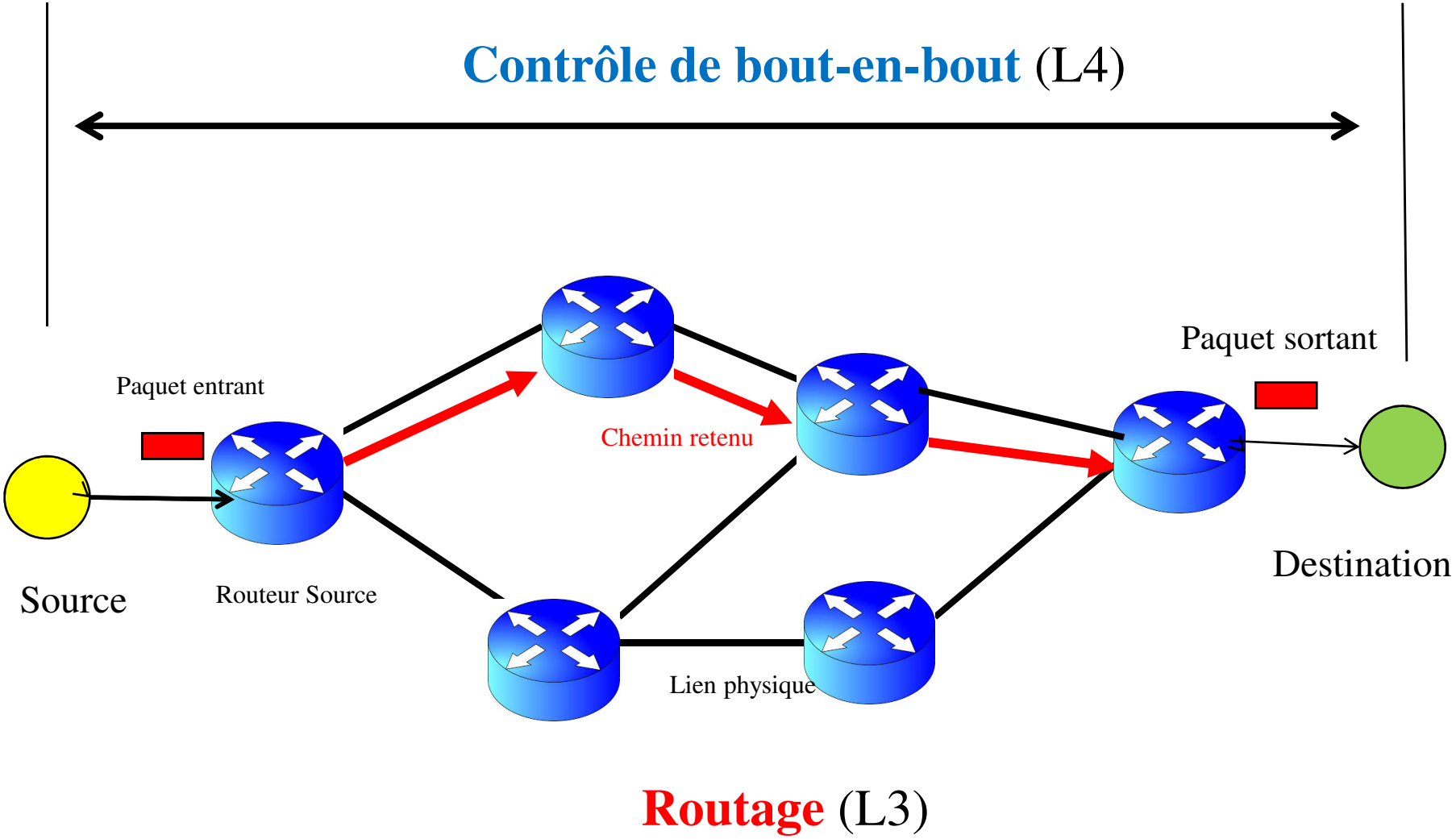


Sans fil
(Point d'accès wifi)

Filaire
(Commutateur Ethernet)



Routage et Contrôle de bout-en-bout



Exemples par couche

+ Interfaces physique
  - RJ45, antenne de WiFi émettant sur la bande IMS
+ liaison pt-à-pt :
  - Lien Ethernet entre un PC et son routeur (via un switch)
  - Lien WiFi entre un smartphone et le « box » ADSL
+ réseaux :
  - Routeurs CISCO, le « box » ADSL
+ contrôle de bout en bout :
  - TCP
+ Interface application
  - HTTP (MS Edge, Chrome), SMTP (Eudora, Outlook)

Constitution par couche

+ Couche 1,2 = Prise, carte, commutateur
+ Couche 3 = Routeur, logiciel (driver)
+ Couche 4 = Logiciel
+ **Couches 1 – 4 = Transfert de données**
+ Couche 4 = verrou bout en bout
+ Couche 7 –5 : logiciels
+ Couches 7 – 5 = Traitement local des données transmises

+ ***Cette formation se concentre sur les couches 2 à 4***

Modèle OSI: résumé

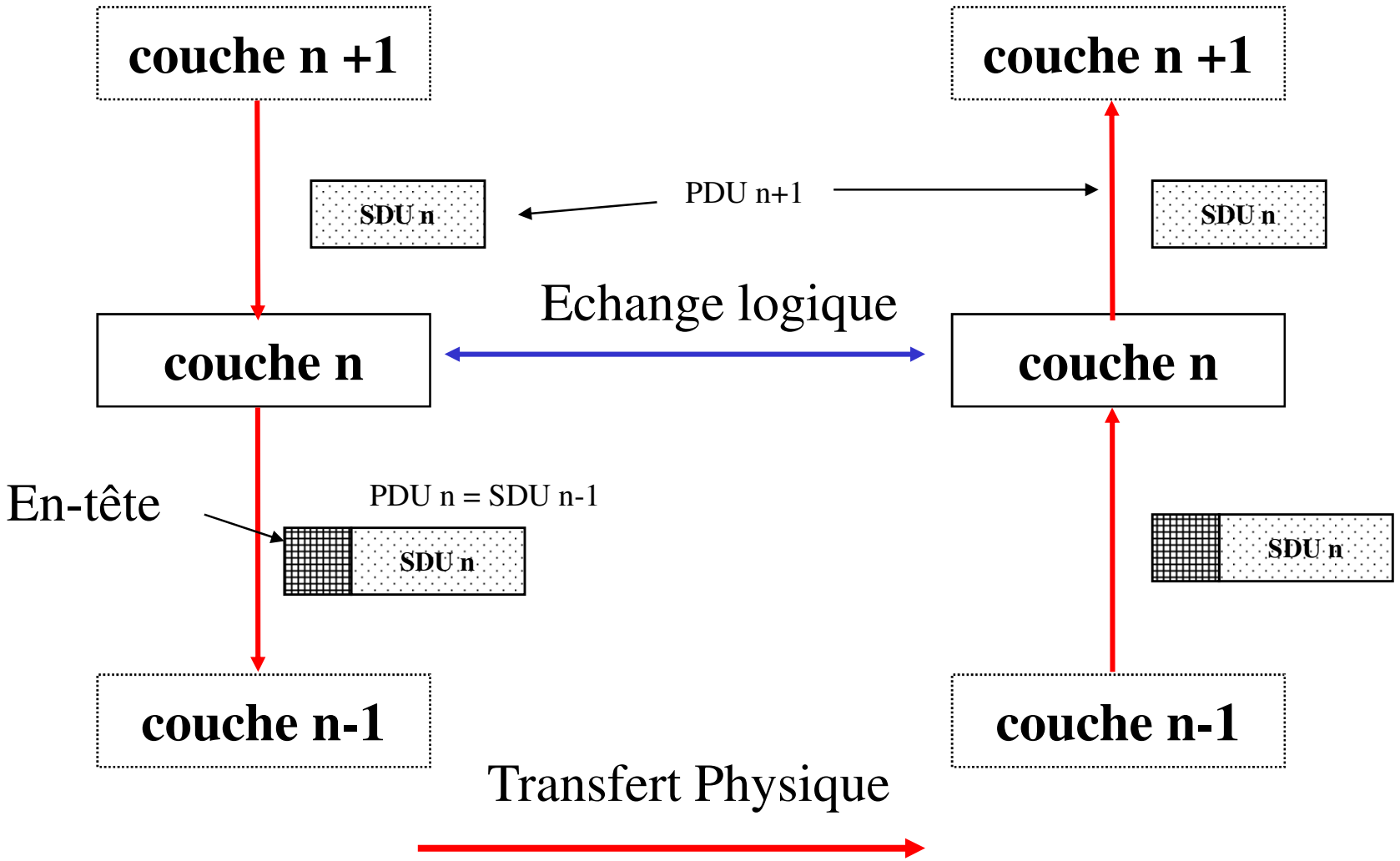
Support de		Courriel, Visio conférence, WeB, Vidéo Streaming, Surveillance, etc. etc.			
	Numéro	Couche	Forme	HW	Protocoles, SW
	7	**Application**	data	FW, DPI	HTTP, SMTP, FTP, SNMP, etc
	6	**Presentation**	data		ASCII, JPG, etc.
	5	**Session**	data		Socket, RPC
	4	**Transport**	segment	FW, DPI	TCP, UDP
	3	**Network**	paquet	Routeur	IP
	2	**Data Link**	trame	Commutateur	Ethermet, HDLC, etc.
	1	**Physical**	élémentbinaire	Hub, Modem	RJ45, RS232, etc
s'appuie sur		Moyen de transmission (Paires torsadées, Hertzien, Fibre optique, etc.)			

+ Fw=Firewall, DPI=Deep Packet Inspection
+ Routeur = Couches **1 à 3**
+ Commutateur = Couches **1 et 2**
+ Terminal = Couches **1, 2, *3*, *4*, (5, 6), 7**

OSI : Rôle d'une couche

+ Abstraction d'un bloc fonctionnel (les services)
  - Les services participant au transfert de données
+ Fournir des **services** à la couche supérieure
  - Service rendu = transfert des données
  - Données = **SDU** (Service Data Unit)
+ Utiliser des services de la couche inférieure
+ Réalisation concrète d'un service : protocole
  - Protocole: règles communes d'échange de données entre deux (ou plusieurs) entités de **même couche**
  - Les données sont codées selon un certain format précis: **PDU** (Protocol Data Unit)

Couches et encapsulation



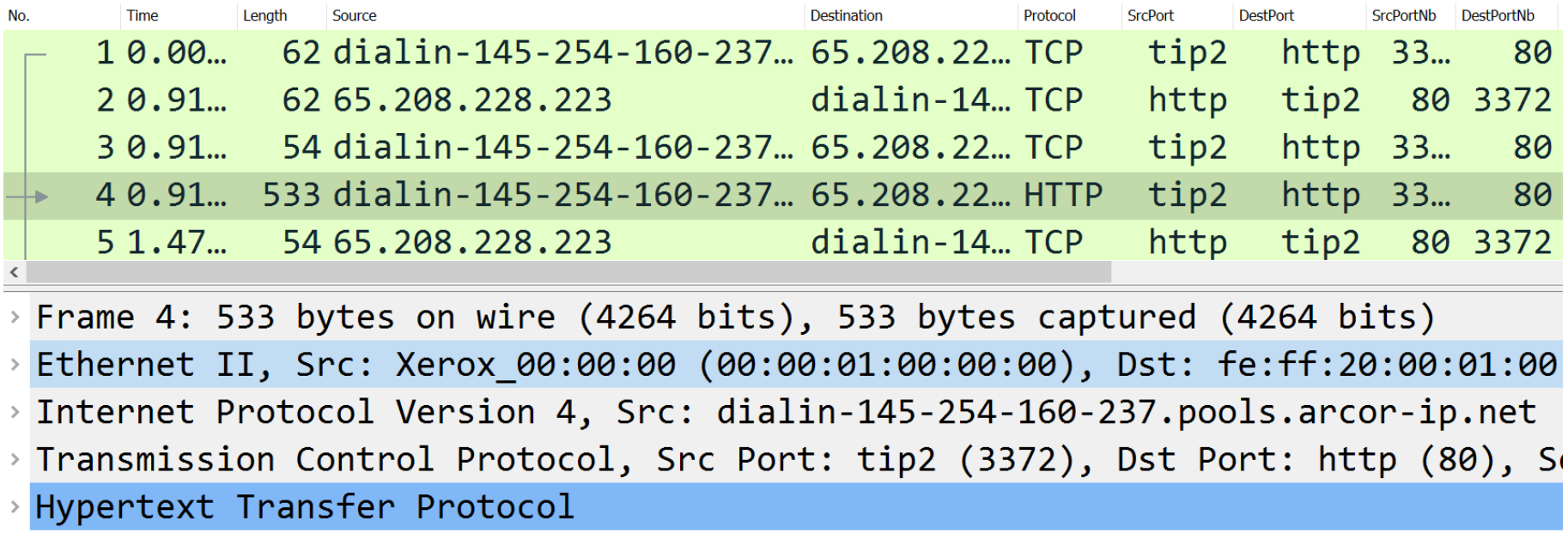
Protocoles

+ Principe
  - **Règles communes** entre **entités distantes** pour réaliser ensemble des fonctionnalités (services) à travers des **actions concordantes**
  - Besoin **d'échange d'informations** pour collaborer
    + Informations « ***pertinentes*** » mise dans l'en-tête (**header**) de chaque paquet
    + ***« Vision »*** de l'**Extérieur** construite à partir des données reçues
+ Base scientifique
  - **Système distribué** : **algorithme** (**protocole**) + **message** (**PDU**)
+ Entité de protocole
  - **Automate** activé par, et réagi à, l'arrivée de PDU ou SDU
  - Généralement réalisé sous forme logicielle
+ Trois prises de vue sur un protocole
  - **Sémantique** : ce qu'il fait (ses fonctionnalités)
  - **Syntaxique** : comment les données sont formatés
  - **Temporel** : **indispensable** pour réagir aux anomalies et éviter le *deadlock*

Empilement de protocoles

+ Modèle OSI :
  - Les couches qui s'enchaînent, chacune joue un rôle donné (avec des services donnés)
+ A Chaque couche correspondent des protocoles qui sont des réalisations particulières
  - Chaque protocole vise un sous-ensemble de services de la couche et possède ses propres règles de fonctionnement
+ Une pile de protocole (*protocol stack*) :
  - un ensemble particulier de protocoles formant une architecture particulière
  + **Il n'y pas de mélange libre des protocoles**
+ Exemple : Internet
  - L3=IP, L4=TCP ou UDP, L7=HTTP, SMTP, FTP, etc.

Encapsulation/Empilement



No.	Time	Length	Source	Destination	Protocol	SrcPort	DestPort	SrcPortNb	DestPortNb
1	0.00…	62	dialin-145-254-160-237…	65.208.22…	TCP	tip2	http	33…	80
2	0.91…	62	65.208.228.223	dialin-14…	TCP	http	tip2	80	3372
3	0.91…	54	dialin-145-254-160-237…	65.208.22…	TCP	tip2	http	33…	80
4	0.91…	533	dialin-145-254-160-237…	65.208.22…	HTTP	tip2	http	33…	80
5	1.47…	54	65.208.228.223	dialin-14…	TCP	http	tip2	80	3372

+ Une trame Ethernet encapsulant
+ Un paquet IP qui véhicule
+ Un segment TCP qui transporte
+ Un message HTTP

Capture/Décodage

No.	Time	Length	Source	Destination	Protocol	SrcPort	DestPort	SrcPortNb	DestPortNb	Info
4	0.91…	533	dialin-145-254-160-237…	65.208.22…	HTTP	tip2	http	33…	80	GET

```
Frame 4: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits)
Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe
Internet Protocol Version 4, Src: dialin-145-254-160-237.pools.arcor-ip.net (145
Transmission Control Protocol, Src Port: tip2 (3372), Dst Port: http (80), Seq:
Hypertext Transfer Protocol
  GET /download.html HTTP/1.1\r\n
  Host: www.ethereal.com\r\n
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20046
```

```
0030  25 bc a9 58 00 00 47 45  54 20 2f 64 6f 77 6e 6c   %··X··GE T /downl
0040  6f 61 64 2e 68 74 6d 6c  20 48 54 54 50 2f 31 2e   oad.html  HTTP/1.
0050  31 0d 0a 48 6f 73 74 3a  20 77 77 77 2e 65 74 68   1··Host:  www.eth
0060  65 72 65 61 6c 2e 63 6f  6d 0d 0a 55 73 65 72 2d   ereal.co m··User-
0070  41 67 65 6e 74 3a 20 4d  6f 7a 69 6c 6c 61 2f 35   Agent: M ozilla/5
0080  2e 30 20 28 57 69 6e 64  6f 77 73 3b 20 55 3b 20   .0 (Wind ows; U;
```

+ Wireshark permet de capturer le trafic puis de le décoder
+ Ici, le décodage du début des data traitées par HTTP
  - PDU vs HTTP, SDU vs TCP

Protocole à travers un exemple

Objectif

+ Comprendre la réalisation d'un protocole pour un service donné
  - Service visé : service orienté connexion
+ Identifier les ingrédients pour réaliser un protocole
  - Analyse des besoins fonctionnels
  - Analyse des conditions de réalisation
  - Identification des règles et mécanismes (algorithmes)
  - Identification des informations à échanger
  - Identification des ressources à mettre en place

+ *On ne va pas ici jusqu'à la réalisation concrète*:
  - Réalisation concrète: voir l'exemple HDLC

Conception d'un protocole

+ *Nous allons nous livrer à un exercice de conception d'un protocole*
+ Il faut d'abord définir le service rendu (càd ses fonctionalités)
+ Service : ***Communication orientée connexion***
+ Fonctionnalités à réaliser (il faut et il suffit de vérifier)
  - Pas de perte, ni désordre, ni duplication
+ Il faut à présent
  - déterminer les conditions de réalisation
  - Choisir les procédés
  - Déterminer les informations internes à échanger

Com. Orientée connexion

+ Analyse des conditions de réalisation :
  - Matérialisation du circuit : **Numérotation des paquets**
+ Regardons le 1er facteur d'erreur: les **pertes**
+ Détection de perte
  - vérification numéro
+ Reprise sur les pertes
  - Retransmission
+ Mécanismes de retransmission
  - Coopération émetteur/récepteur, diverses approches

Retransmission: conditions

+ Côté émetteur : :
  - Mise dans un **tampon** des data déjà transmis
  - Corollaire : Acquittement (**ACK**) donné par le récepteur
    + Les data dûment acquittés sont alors éliminés côté émetteur
  - Précaution: **timeout** (sinon risque de blocage fatal (*deadlock*))
+ Côté récepteur
  - Il faut détecter d'éventuelles pertes
    + ou, plutôt, s'assurer la continuité du flux
  - Il faut envoyer des ACK si continuité de data
    + Sans ACK, il y aurait **débordement** du tampon
  - Deux comportements possibles en cas de perte
    - **NE PAS** donner **ACK** aux data postérieures reçues (car NON contigües)
    - Signaler **explicitement** la perte

Coopération Emetteur-Récepteur

+ Retransmission dictée par le récepteur
  - Signalisation implicite ou explicite de perte
+ Signalisation explicite
  - Suite à la détection d'un « trou »
+ Signalisation implicite (*par déduction*)
  - Récepteur envoie des **ACK uniquement**
  - **Perte** : Absence d'ACK au bout du **timeout**
    + Mécanismes ARQ (*automatic repeat request*)
    + **Déduction** d'une « perte »
+ Stratégie de retransmission en cas de perte
  - Go-Back-N
  - Retransmission sélective (SR)

Go-Back-N vs SR

+ Go-Back-N
  - Retransmission de **toutes** les PDU **à partir de** celle perdue
+ Retransmission sélective
  - Retransmission de la **seule** PDU perdue
+ Go-Back-N
  - Plus simple à réaliser (surtout côté récepteur)
  - Consommation de la bande passante
+ Retransmission sélective
  - Consommation de la bande passante minimale
  - Gestion plus élaborée côté récepteur
    + Chaînage des data reçues avec « trous »

Com. Orientée connexion

+ Les deux autres facteurs d'erreurs sont traités plus facilement

+ Reprise sur désordre
  - **Assimilée** à une perte
  - Ceci simplifie la logique du traitement
  - Motivation similaire à Go-Back-N

+ Reprise sur duplication,
  - **Ignorer** la duplication

Autres fonctions d'optimisation

+ Fenêtre d'anticipation
  - Envoi d'un train de plusieurs messages **avant** le retour d'ACK du première d'entre eux
+ Procédé *piggy-backing* :
  - ACK portée par une PDU de données (émise par *l'entité d'en-face*)
  - Hypothèse : communication **bidirectionnelle**
+ Contrôle de flux
  - Suspension de transmission demandée par le récepteur
  - Dû aux difficultés locales du récepteur
    + (à ne pas confondre avec le contrôle de congestion = lié aux problèmes au sein du réseau)
+ Exemples de protocoles : HDLC (LAP-B, etc), TCP

Trois phases

+ **Etablissement de connexion**
  - Une communication orientée connexion nécessite
    + Une description précise de son **état** (position des flux entrants et sortant)
    + Tampon d'émission
  - Nécessité de créer un **environnement spécifique** (état, tampon) pour chaque communication
    + Nécessité de **mobiliser** une fraction des **ressources** du système
  - Consommation de CPU (gestion) et de mémoire (tampon)
+ **Transfert de données**
  - Les automates protocolaires sont en action et gèrent les flux
+ **Libération**
  - Opération duale permettant de **libérer** les ressources occupées

HDLC

Protocole à travers un exemple concret

Présentation à caractère illustratif

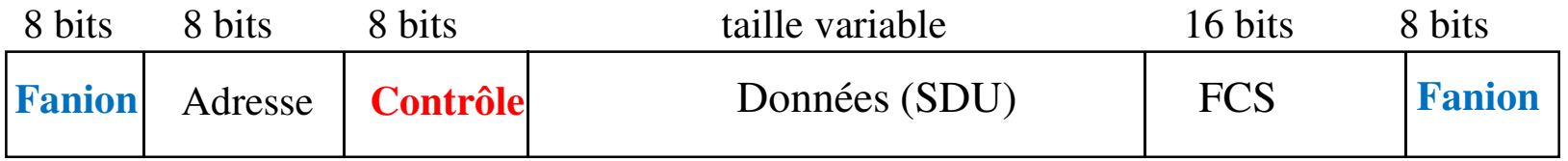
Ce protocole n'est pas décrit dans tous ses détails

HDLC en bref

+ HDLC = High level Data Link Control
  - Dérivé du SDLC (Synchronous Data Link Control), un produit IBM (réseau SNA)
+ Norme ISO (**1976**)
+ **Couche 2**: Gestion de liaisons point-à-point
+ **Service offert**: Communications orientées connexion
+ Adaptation/dérivation nombreuses
  - LLC, LAP-B (X25), LAP-D (RNIS), LAP-F (Relais de trame), PPP (accès fournisseur via modem), etc.
+ 3 modes opérationnelles
  - *NRM, ARM : modes maîtres/esclaves*
  - **ABM** : communication full duplex (*balanced*) asynchrone

Format (1/2)

+ Fanion = 01111110 (6 « 1 » consécutives)
  - Délimiteur
  - Pas de séquence « 111111 » dans les autres champs
  - Technique de *bit-stuffing* (cf plus loin) pour la transparence
+ Adresse (1 octet)
  - Identificateur de destinataire (supporte liaisons multipoints)
  - Indication de Commande/Réponse (LAP-B, LLC)
  - Indication de Primaire vs secondaire en pt-à-pt



Format (2/2)

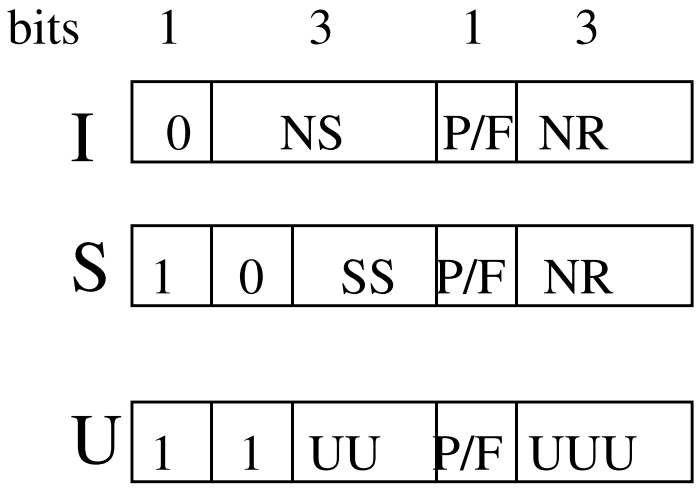
+ Contrôle (1 octet, variante = 2 octets)
  - Information pour la gestion de liaison
  - 3 types de trames
    + I=Information
    + S=Supervision
    + U=Unnumbered
  - 2 octets si numérotation sur 7 bits
+ Données : taille variable
+ FCS (2 octets) = Détection d'erreur
  - Code CRC (x**16+x**15+x**2+1)
  - Élimination de trame si erreur => perte

Bit-Stuffing

+ Rendre « 111111 » spécifiques aux fanions dans les séquences binaires transmises
+ A l'émission, entre les deux fanions
  - Comptage de « 1 » consécutifs
  - Insertion **systématique** d'un « 0 » après 5 « 1 » consécutifs
+ A la réception
  - Condition : fanion de début détecté
  - Comptage de « 1 » consécutifs
  - Si 5 « 1 » consécutifs, alors
    + Si le sixième est un « 0 », **retirer** le « 0 » pour **récupérer les données initiales**
    + Si le sixième est un « 1 », prêt pour reconnaître le fanion de la fin

Le champ Contrôle

+ Trois types de trames
  - I=Information
  - S=Supervision
  - U=Unnumbered
+ NS = numéro de la trame



Contrôle (2/2)

+ NR = numéro de la prochaine trame attendue
  - Réalisation de ACK et de piggy-backing
+ Contrôle sur 2 octets pour les trames I et S si numérotation (NS, NR) sur 3 bits
+ P/F : ***si validé (P/F=1),*** interprété suivant l 'indication dans le champ « adresse » comme
  - Poll : invitation à émettre/répondre (**commande**)
  - Final : **réponse** (à un Poll précédent)

Trames I

+ Flux matérialisé (numérotation)
  - Modulo 8
  - Il y a une *version étendue* de HDLC avec modulo 128
    + Champ de contrôle sur 2 octets
    + N(S) et N(R) sur 7 bits
+ Fenêtre d'anticipation :
  - Limite protocolaire : NS-1
    + Si 8 trames émises (ACK des 8 **équiv.** Rien reçu)
    + ***Voir plus loin pour une discussion plus détaillée***
  - Limite réelle = Min (Limite physique, Limite protocolaire)

Trames S

+ Quatre types
  - RR (SS=00) = Receive Ready
  - RNR (SS=10) = Receive Not Ready
  - REJ (SS=01) = Reject = Go-Back-N
  - SREJ (SS=11) = Selectif reject (***optionnel***)
+ Supervision =Contrôle l'échange des données
  - Accusé de réception = donné par NR
    + Toute trame S porte NR donc ACK des trames jusqu'à NR-1
  - Reprise sur perte
    + REJ, SREJ
  - Contrôle de flux
    + Le couple <RR, RNR>

Trames U (1/2)

+ SABM (001P1111) : Demande de connexion
  - Uniquement en mode commande
  - SABME (011P1111) pour numérotation 7-bits
+ DISC (010P0011) : Libération
  - Uniquement en mode commande
+ UA (011F0011) : ACK non numéroté
  - Uniquement en mode réponse
  - accepter une connexion (SABM)
+ DM (000F1111) : Mode déconnecté
  - Uniquement en mode réponse
  - Refuse une connexion (SABM)
  - Confirmer une libération
  - Réagir aux trames I et S si déconnecté

Trames U (2/2)

+ FRMR (100F0111) : Rejet de trame
  - Rejet des trames non conformes à la norme
+ XID (101(P/F)1111) : Echange de capacité
  - Modes commande/réponse
+ TEST (111(P/F)0011) : Echange d'identité
  - Modes commande/réponse
+ RSET (100P1111) : Reset N(R)
  - En cas de problème (perte de synchronisation)
+ *---- trames supplémentaires LLC*
  - *UI () : Information non numérotée*
  - *Mode commande pour être acquitté par UA*

Exemple 1 : une trame I (Info)

+ 01111110 = Fanion
+ Adrs = (01) =00000001
+ Contrôle = 0 101 0 001
  - (I), NS=5, P/F=0, NR=1
+ Données :
  - *Données soumises (dont 7 "1"):* ***1111111 0000 1***
  - **Transmise (et reçue)** via HDLC : **11111<u>0</u>1100001**
  - *Données restituée* : ***1111111 0000 1***
+ FCS : le code qu'il faut
+ 01111110 = fanion de la fin

Exemple 2 : Une trame S

+ 01111110 = Fanion
+ Adrs = (03) =00000011
+ Contrôle = 10 00 0 011
  - (S), RR, NR=3, P/F=0
+ Données : AUCUNE (trame de supervision)
+ FCS : le code qu'il faut
+ 01111110 = fanion de la fin

Automate HDLC (1)

+ Le protocole HDLC s'exécute à l'aide d'un couple d'entités logicielles (***automate*** HDLC) qui exécutent toutes les règles du protocole HDLC
+ Chaque entité a besoin de
  - Un tampon : garder les trames émises en attente de ACK
  - Un couple d'indicateurs : <V(S), V(R) > (état du lien)
    + V(S) = numéro de la prochaine trame à émettre
    + V(R) = numéro de la prochaine trame attendue
  - Quelques paramètres de configuration
+ Pour une communication bi-directionnel:
  - Il y a deux flux **indépendants** que HDLC doit gérer
  - Le flux entrant (repéré par V(R))
  - Le flux sortant (repéré par V(S))

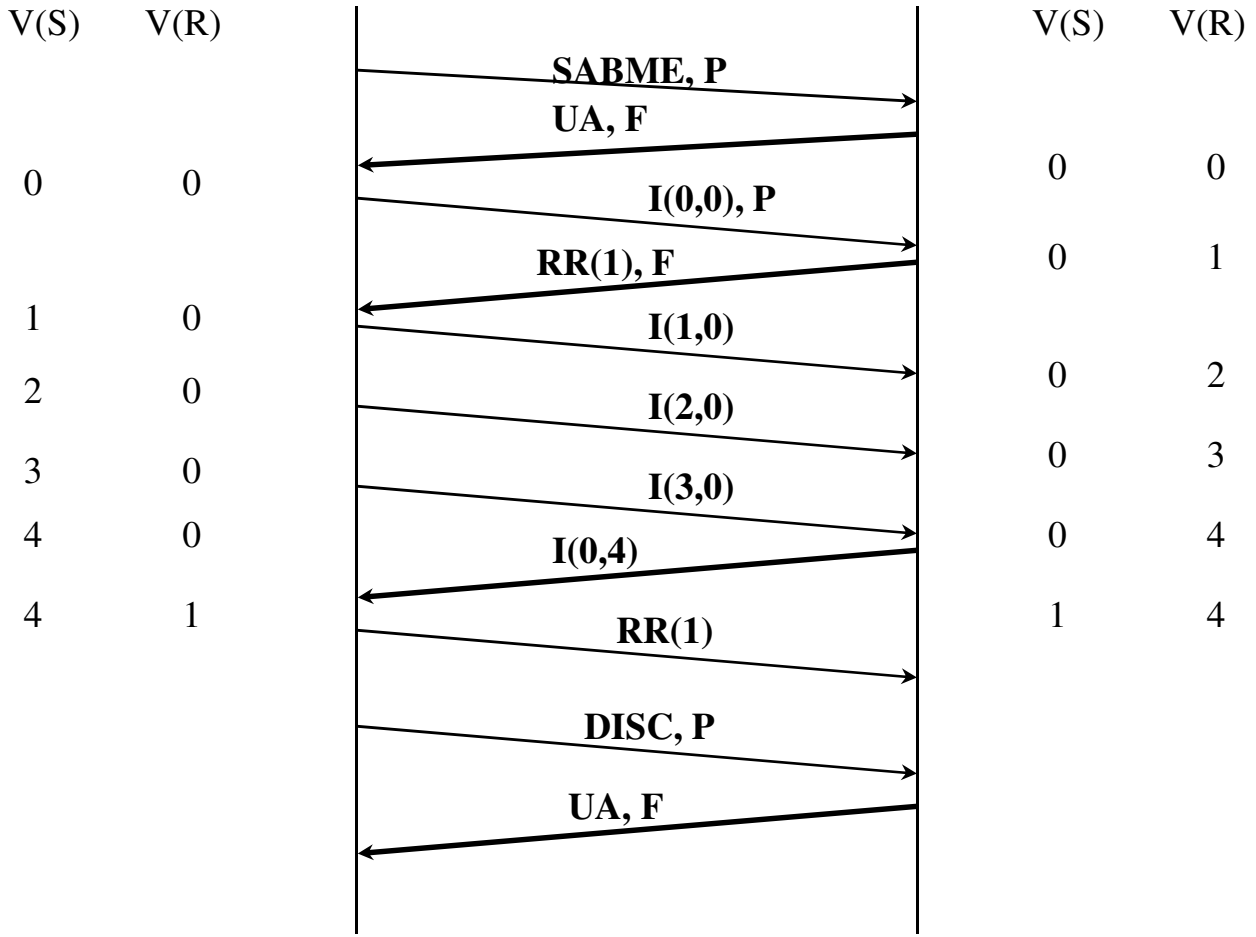
Automate HDLC (2)

+ Voici quelques paramètres
  - Temporisateur T1:
    + déclenchée lors de l'émission de chaque trame avec P (commande)
    + Son expiration signifie l'échec de la commande
    + Doit être désarmé à la réception de la réponse (F)
  - Longueur max d'une trame: N1
  - Nombre max. de retransmission : N2
  - Fenêtre d'anticipation : W
+ Ces paramètres sont à configurer en fonction du contexte et de l'application

Scénario : commentaire

+ Représentation :
  - Deux entités de protocoles
  - Un « milieu » entre deux barres (verticales **OU** horizontales)
    + Milieu = canal reliant les deux entités
+ Communications à trois phases
  - Etablissement de communication
  - Transfert de données
  - Libération
+ V(S), V(R) : **état du lien (*bidirectionnel*)**
  - V(S) = numéro de la prochaine trame à émettre
  - V(R) = numéro de la prochaine trame attendue
+ Tampon d'émission, Temporisateur
+ Couche sup.

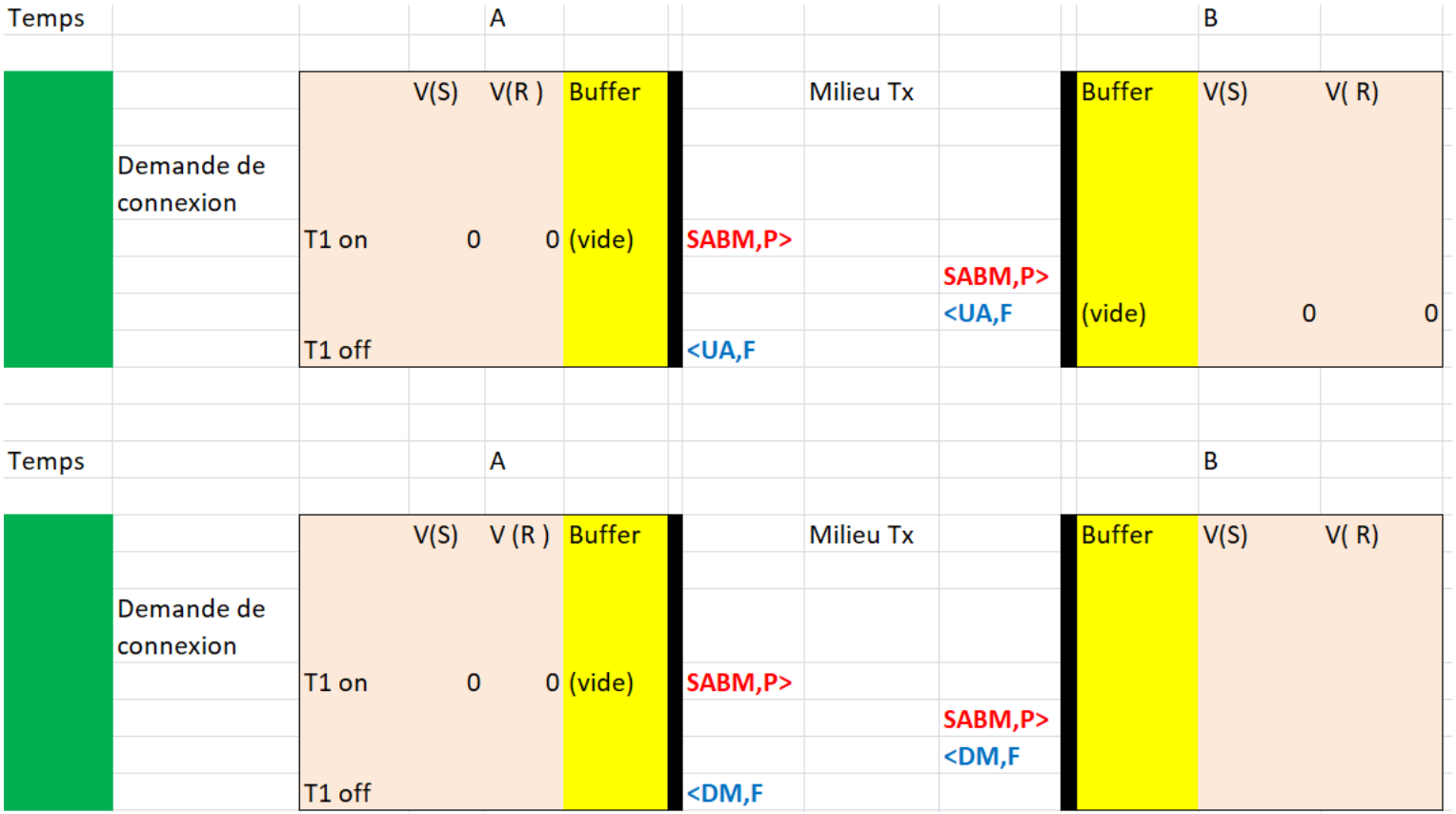
Scenario: présentation classique



Etablissement de connexion

+ Côté Demandeur (A)
  - Emission SABM, P (réponse attendue) et armer T1
  - Réponses valables
    + UA, F: connexion acceptée
    + DM, F: refus
    + Action annexe si réponse : arrêter T1
  - Si T1 expire, relancer l'opération (un total de N2 fois)
  - Data peut être émis après la ***réception*** UA
+ Côté opposé (B)
  - Selon la situation, UA ou DM
  - Si UA (acceptation), il faut créer au préalable
    + Un tampon (mis à vide)
    + Le couple <V(R)=0, V(S)=0>
  - Data peut être émis après l'***émission*** de UA

Etablissement de connexion



Les 2 scenarii: UA (OK) et DM (refus)

Libération

+ L'entité qui veut mettre fin à la communication
  - Emission DISC,P
+ L'autre entité répond par UA,F
+ L'option P n'est pas utile:
  - On est dans une logique de rupture (on attend rien de l'autre)
+ Cette phase est absolument exigée pour libérer des ressources consacrées à une communication
  - Tampon
  - Indicateurs et paramètre
  - Nécessité de traitement

Transfert de données

+ Côté Emetteur (A)
  - Transmission d'une ou plusieurs trame(s) I, avec (I, Ns, Nr)
    + Nombre de trames transmises dépendant de
      - Taille des données soumises ET fenêtre d'anticipation
  - <Ns, Nr> : valeurs du couple <V(S), V(R))
  - Après l'émission de chaque trame
    + Mise en tampon de la trame **ET** Incrémentation de V(S)
  - Si trame émise avec P, armer T1 également.
+ Côté Récepteur (B)
  - A la réception d'une trame (I, Ns, Nr)
    + Vérifier si Ns ==V(R) (numéro reçu vs numéro attendu)
      - Si conforme : accepter **ET** incrémenter V(R)
      - Sinon: demande de retransmission
  - Une réponse immédiate adéquate si trame reçue avec P

Transfert de données (I)

Temps	Couche Sup		A				Milieu Tx		B			Couche sup
			V(S)	V(R)	Buffer			Buffer	V(S)	V(R)		
	Demande de connexion											
		T1 on	0	0	(vide)	SABM,P>						
								SABM,P>				
								<UA,F	(vide)	0	0	
		T1 off				<UA,F						
	Data coupé en		0	0		I,0,0>						
	2 morceaux	T1 on	1	0	I0	I,1,0, P>		I,0,0>		0	0	Remise Da
	Da, Db		2	0	I0, I1			1,1,0, P>		0	1	Remise Db
										0	2	
								<RR,2,F				
		T1, off				<RR,2,F						
			2	0	(vide)							

- RR : acquittement explicite (car une répons est exigée)
- Effet de vidange de tampon côté A : Da et Db sont (définitivement) transmises avec succès et dans l'ordre

Transfert de données (II)

Temps	Couche Sup		A				Milieu Tx		B				Couche sup
			V(S)	V(R)	Buffer				Buffer	V(S)	V(R)		
	Data coupé en		0	0		I,0,0>							
	2 morceaux	T1 on	1	0	I0	I,1,0, P>		I,0,0>		0	0		Remise Da
	Da, Db		2	0	I0, I1			1,1,0, P>		0	1		Remise Db
										0	2		
													Data en 3 morceaux
								<I,0, 2,F		0	2		Ea, Eb, Ec
	Remise Ea	T1, off				<I,0, 2,F		<I,1, 2	I0	1	2		
	Remise Eb		2	1	(vide)	<I,1, 2		<I,2, 2, P	I0, I1	2	2	T1 on	
	Remise Ec		2	2		<I,2, 2, P			I0, I1, I2	3	2		
			2	3		RR,3,F>							
								RR,3,F>				T1 off	
									(vide)	3	2		

- Exemple **Piggy backing**: **B vers A** avec **I,0,2, F**
- Par contre: **A vers B**: ACK spécifique avec **RR,3,F**

Reprise sur Perte avec T1

Temps	Couche Sup		A				Milieu Tx		B			Couche sup
			V(S)	V(R)	Buffer				Buffer	V(S)	V(R)	
	Data coupé en		0	0		I,0,0>						
	2 morceaux	T1 on	1	0	I0	I,1,0, P>		I,0,0>		0	0	Remise Da
	Da, Db						perte I,1,0			0	1	
		...										
		T1, TimeOut										
		T1 on	1	0	I0	I,1,0, P>						
			2	0	I0, I1			1,1,0, P>		0	1	Remise Db
										0	2	
								<RR,2,F				
		T1, off				<RR,2,F						
			2	0	(vide)							

- Reprise I1 (la dernière), grâce à T1

Reprise sur Perte avec REJ

Temps	Couche Sup		V(S)	V(R)	Buffer		Milieu Tx		Buffer	V(S)	V(R)		Couche sup
				A						B			
	Data coupé en		0	0		I,0,0>							
	2 morceaux	T1 on	1	0	I0	I,1,0, P>		I,0,0>		0	0		Remise Da
	Da, Db		2	0	I0, I1			1,1,0, P>		0	1		Remise Db
										0	2		
													Data en 3 morceaux
								<I,0, 2,F		0	2		Ea, Eb, Ec
	Remise Ea	T1, off				<I,0, 2,F		<I,1, 2	I0	1	2		
			2	1	(vide)		*perte*	<I,2, 2, P	I0, I1	2	2	T1 on	
			2	1		<I,2, 2, P			I0, I1, I2	3	2		
						REJ,1,F>							
								REJ, 1,F>	I1, I2	3	2	T1,off	
								<I,1,2					
	Remise Eb		2	1		<I,1,2		<I,2,2,P					
	Remise Ec		2	2		<I,2,2,P							
			2	3		RR,3,F>							
								RR,3,F>				T1 off	
									(vide)	3	2		

Remarquer la prise en compte de l'ACK de I0 par REJ,1

Pause/Reprise: RNR/RR

		V(S)	V(R)	Buffer			Buffer	V(S)	V(R)	
Demande de connexion										
	T1 on	0	0	(vide)	SABM,P>					
						SABM,P>				
						<UA,F	(vide)	0	0	
	T1 off				<UA,F					
Data coupé en 2 morceaux Da, Db		0	0		I,0,0>					
	T1 on	1	0	I0	I,1,0, P>		I,0,0>	0	0	~~Remise Da~~
		2	0	I0, I1			I,1,0, P>	0	1	~~Remise Db~~
								0	2	
						<RNR,2,F				
	T1 off				<RNR,2,F					Remise Da
		2	0	(vide)						Remise Db
Data Dc						<RR,2				
					<RR,2					
		2	0		I,2,0,P>					
	T1 on	3	0	I2			I,2,0,P>	0	3	Remise Dc
							<RR,3,F			
	T1 off	3	0		<RR,3,F					
		3	0	vide						

- *Pause (RNR) réclamée par le récepteur à la réception de Db*
- *Reprise (RR) signifiée à la remise de Db à la couche sup.*
- *Libération non représentée*

Autres fonctionnalités utiles

+ Contrôle de flux
  - RNR,Nr: bloquer la suite de la transmission
    + Souvent parce qu'il y a un problème de place pour réception
    + Equivalent de « PAUSE »
    + Donne ACK des trames reçus jusqu'à Nr-1
      - Nr = état V(R) du moment
  - Reprise avec RR, Nrb
    + Equivalent de « PLAY »
    + Donne ACK des trames reçus jusqu'à Nrb-1
      - Nr = état V(R) du moment
+ FRMR
  - Rejet d'une trame bien reçue mais sémantiquement erronée:
    + Taille dépasse N1
    + N(R) non conforme
    + Champ de Contrôle non valide (par exemple SREJ non supporté)
    + Etc.

Fenêtre d'anticipation et **au delà**

+ Fenêtre d'anticipation
  - Nombre de trames pouvant être transmis sans attendre ACK
  - Il faut : suffisamment de places dans buffer
  - Il faut aussi : pas de confusion possible
+ Cas HDLC: numérotation module 8
  - On peut numéroter 0, 1, … 7
  - MAIS: Fenêtre d'anticipation MAX: **7** (8-1)
+ Si c'était 8, il y aurait confusion
  - Cf diagramme suivant
+ Leçon **fondamentale**
  - Protocole = **automate** dans un **système distribué**
  - Pas de vision globale (« *vision agnostique* »)
  - Fonctionne avec les **informations reçues** uniquement

Situation : A transmet 8 trames à B

+ Mauvaise approche (confusion)
  - Fenêtre d'anticipation : 8
  - Transmission des trames I0, … I7 ((*avec probablement P*)
  - Scenario 1
    + Les 8 trames ont été reçues
    + B répond avec RR,**0**
  - Scenario 2
    + **Aucune trame n'a été reçue**
    + B, de son côté, décide librement d'envoyer un RR, avec **RR,0**
+ Bonne approche
  - Fenêtre d'anticipation : 7
  - Transmission des trames I0, … I6 (*avec probablement P*)
  - Même en cas de perte totale, il n'y aurait pas de confusion
  - I7 sera transmise après

Implémentation

+ Protocole = automate =Algorithme
+ Protocole = programme (souvent codé en C/C++)
+ Comme tout programme, ça se définit avec
  - Input, Output
  - Variable (*attributs*)
    + Cas HDLC: Tampon, <V(S), V(R)>, les paramètres, etc
  - Fonctions (*méthodes*)
    + Cas HDLC: établissement de connexion, Go-back N, etc
+ Code = Algorithme
+ Input/Output ( )
  - L'ordre reçue de la couche supérieur
    + Connexion, transmission des data, etc.
  - PDU reçues d'en-face véhiculant l'ordre d'en-face
  - ***(Ordre = primitive de service)***

HDLC: connexion

+ Cas de l'établissement de connexion: A vers B
+ Côté initiateur (A)
  - Input: demande de connexion
  - Output: SABM
  - Action 1: Créer puis envoyer une trame SABM
  - Action 2 : Armer T1
+ Côté B
  - Si trame reçue = SABM
  - Examiner possibilité d'acceptation
    + Conditions générales
    + Possibilité de créer le contexte (Tampon, V(S), V(R), etc)
  - Emission de UA ou DM selon le cas

HDLC: transfert data

+ Cas de transfert de A vers B
  - avec Data correspondant à deux trames I
+ Côté initiateur (A)
  - Vérifie si place dans tampon avant l'émission de chaque I
  - En cas de blocage: envoie RR,Nr, P pour provoquer un ACK
  - En cas de place: émission I, avec V(S), V(R) (*et P*)
    + actualise V(S), met la trame dans le tampon
+ Côté B
  - A l'arrivée d'une trame I,Ns, Nr
  - Vérifie Ns==V(R)
    + Si OK: actualise V(R) et rémet la datz reçue
    + Si Problème: émettre REJ, Nr avec Nr=V (R)

Point-à-Point vs bout-en-bout

+ Communication orienté connexion
  - « liaison » entre deux entités
  - niveau 2 ou 4 ou niveau applicatif
+ Niveau 2 : point-à-point
  - Liaison dont les caractéristiques sont (quasiment) figées
  - Courte distance
  - Exemple : HDLC
+ Niveau 4 : bout-en-bout
  - Souvent, les caractéristiques de la « liaison » sont fluctuante
  - Peut emprunter un long parcours
  - Exemple : TCP