Systèmes Répartis

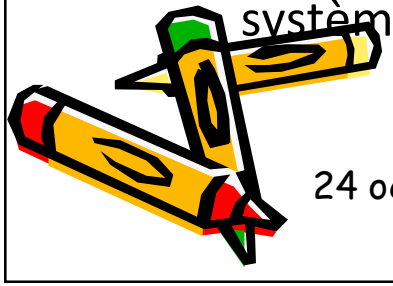
Ridha Azizi

Professeur Technologue

Iset de Sousse

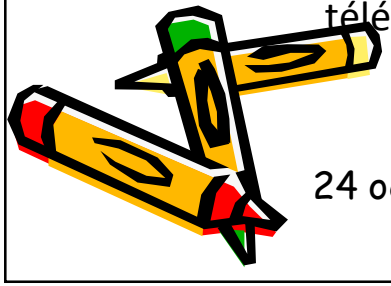
Introduction aux systèmes répartis

- Les ordinateurs ont subi des changements incroyables depuis leur mise en opération vers1945:
  - Plus en plus de puissance,
  - Coût de fabrication a constamment.
- Les appareils subissent des changements constants et de plus en plus rapides tant du point de vue logiciel que matériel.
- Depuis très peu de temps, nous retrouvons sur le marché des systèmes multiprocesseurs, des systèmes d'exploitation pour le traitement parallèle et des réseaux puissants d'interconnexion.
- C'est là l'importance de prendre brièvement connaissance avec le système d'exploitation de demain.

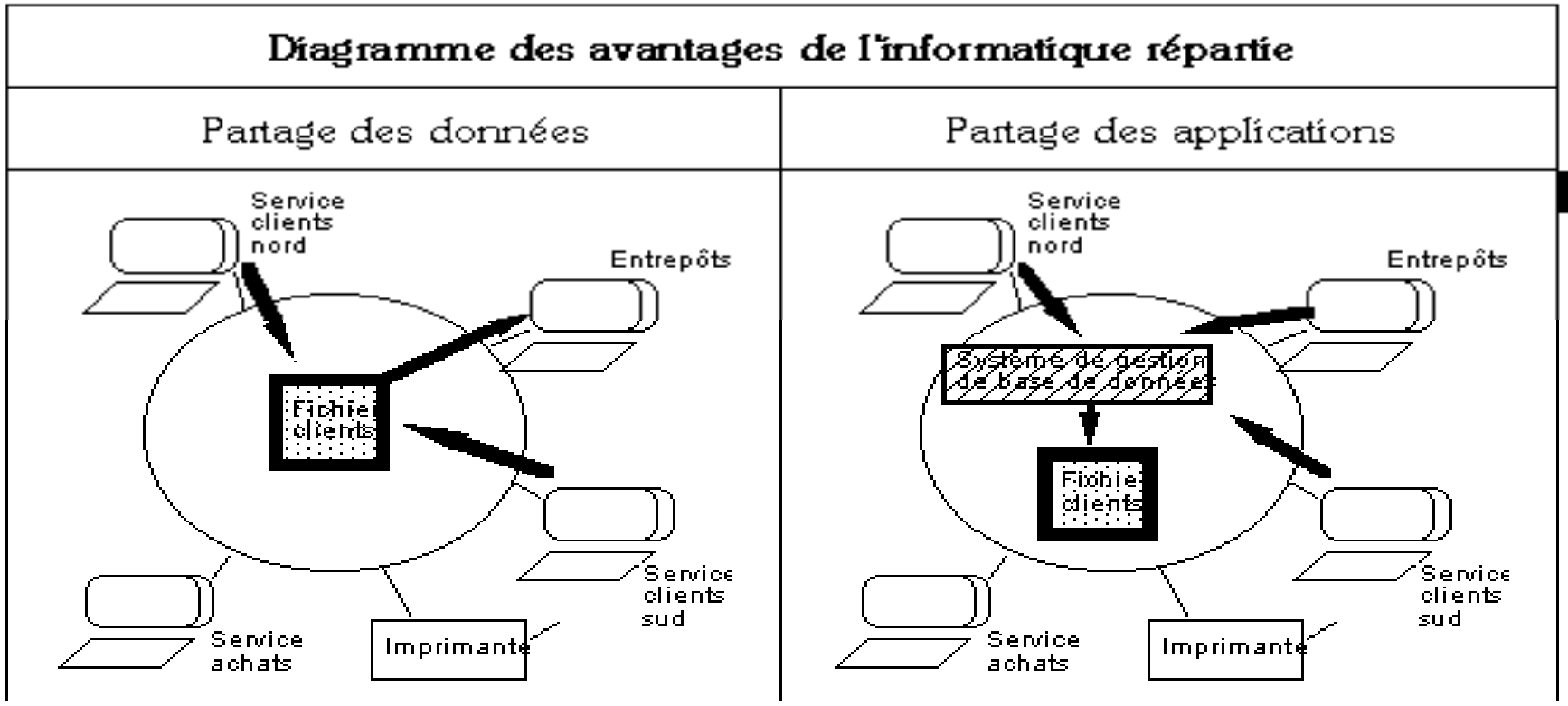


Introduction aux systèmes répartis

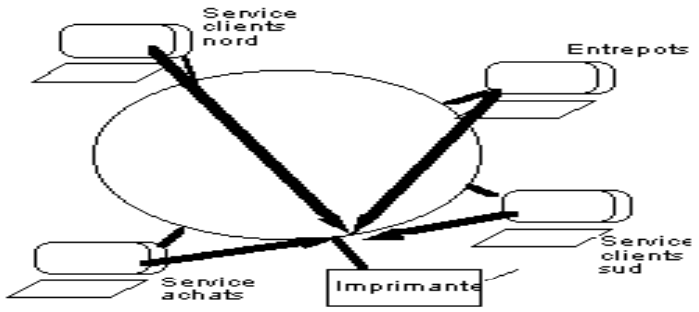
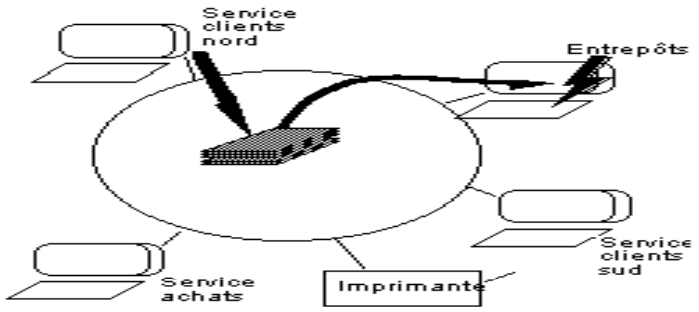
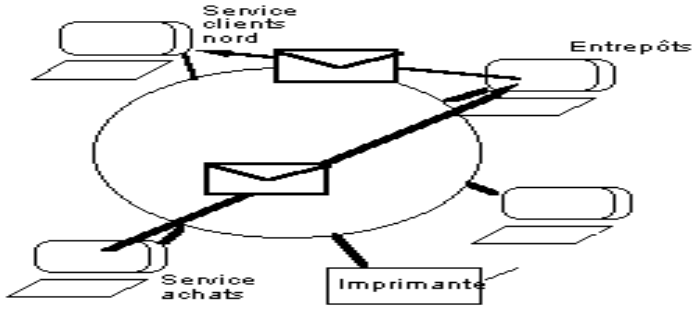
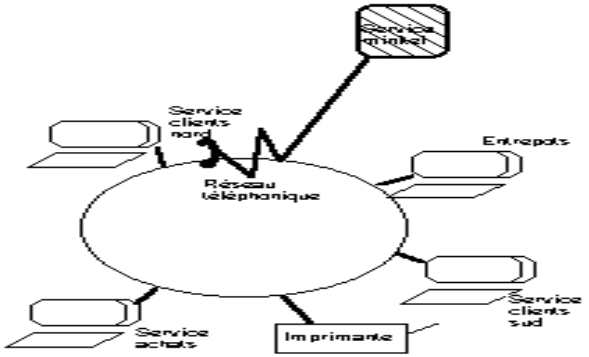
- Nous sommes déjà entrés quelque peu dans l'informatique répartie qui elle nous amènera vers l'informatique distribuée.
  - L'informatique répartie s'oppose à la fois à l'informatique centralisée, celle des gros ordinateurs, et à l'informatique individuelle, celle des micro-ordinateurs. Elle pallie certains désavantages de cette dernière par:
  - Le partage des données grâce à un accès individuel, en lecture, par le réseau à des fichiers communs situés sur un disque quelconque ainsi que par le transfert de fichiers d'un disque à un autre.
  - Le partage des applications. Pour l'exploitation individuelle, par le réseau, d'un seul logiciel de base de données sur le disque d'une des machines connectées.
  - Le partage des ressources : chaque utilisateur connecté peut utiliser une même imprimante.
  - Les communications: envoi par le réseau de courrier dans une boîte aux lettres électronique à un ou plusieurs utilisateurs connectés. Accès par le réseau téléphonique à des services d'informations : annuaires, banques de données, etc.



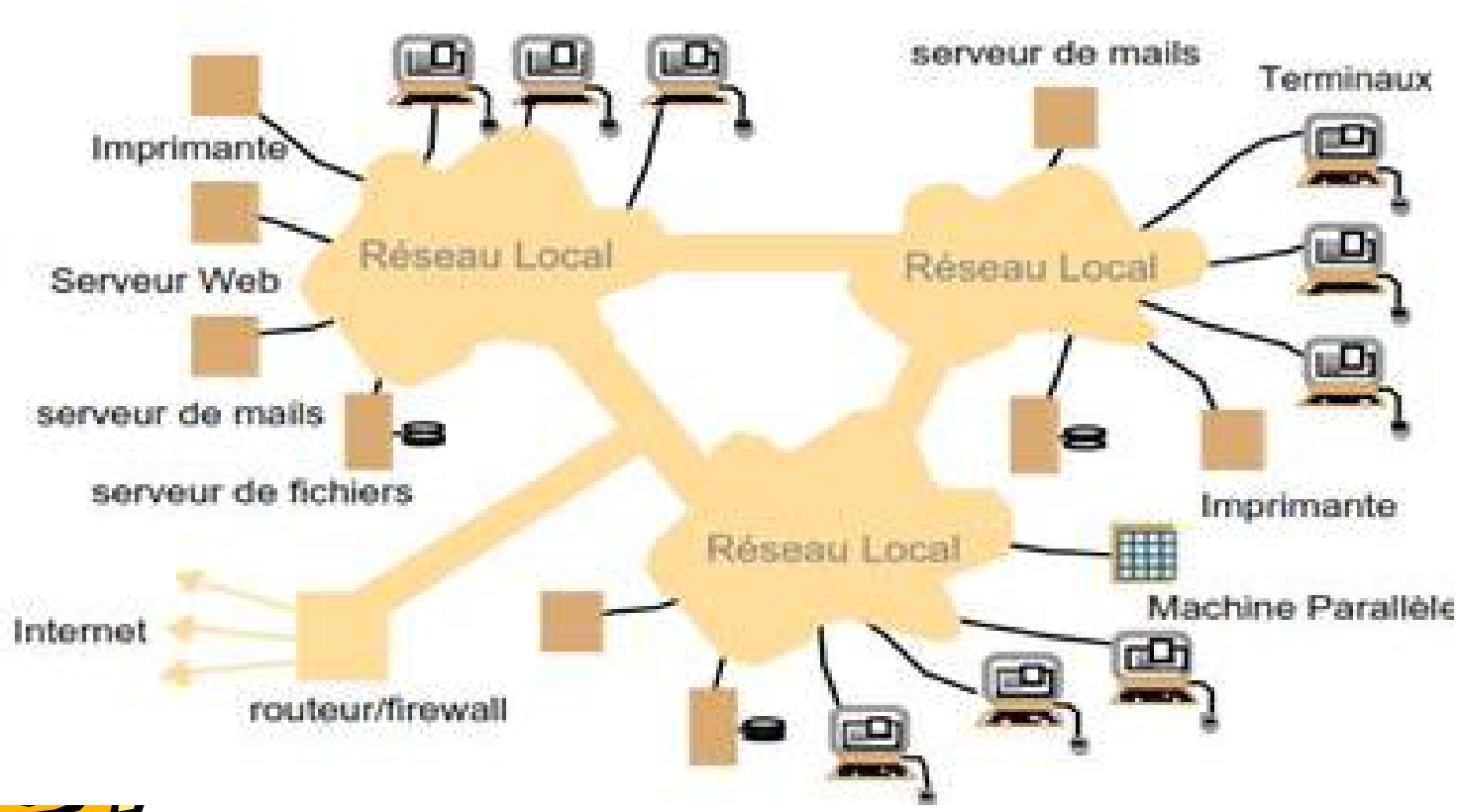
Introduction aux systèmes répartis



Introduction aux systèmes répartis

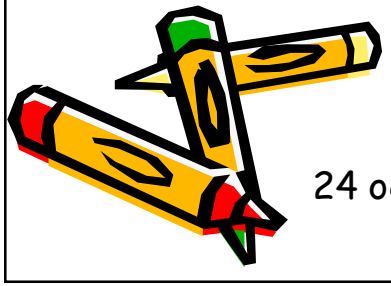
Partage de ressources	Partage du travail
	
Communication par une messagerie électronique	Accès à des informations ou à des services publics
	

Exemple de Système Réparti : Un intranet

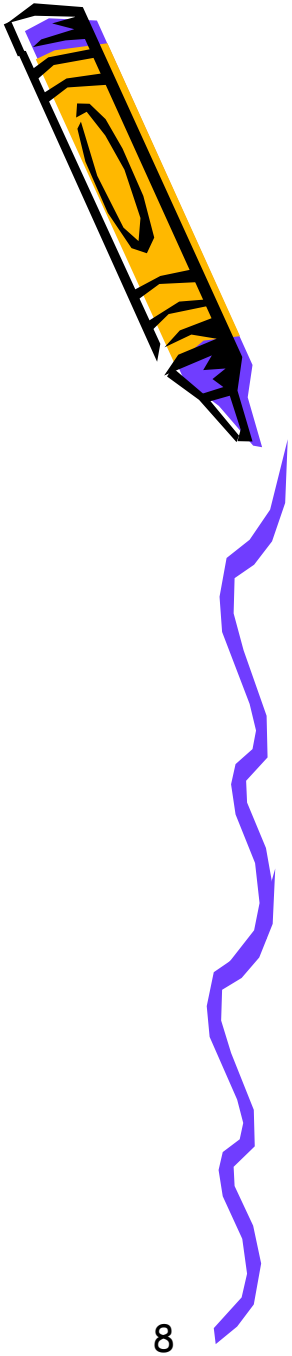


Pourquoi une informatique répartie ?

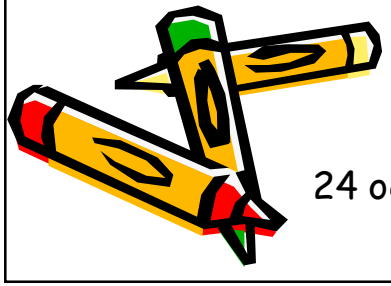
- Raisons budgétaires : économie de logiciels, de matériels
- Raisons intrinsèques : adapter le système à l'application
  - BDD réparties, Web, systèmes bancaires…
- Besoin de partager
  - des informations : fichiers, BDD, messages
  - des ressources : unités de stockage, imprimantes, serveurs
  - Des services
- Accélérer le calcul
  - Parallélisation de tâches
- Alléger la charge : réduire les goulots d'étranglement
- Augmenter la fiabilité : duplication de machines, de données ⇒réalisation de systèmes à haute disponibilité



Pourquoi une informatique répartie ?

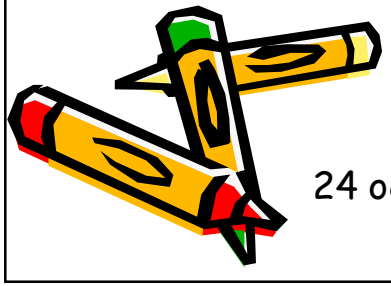


- Qualité de service : diminuer les coûts, les délais, augmenter la disponibilité
- Réaliser des systèmes ouverts, évolutifs : adjonction facile de matériels et logiciels.



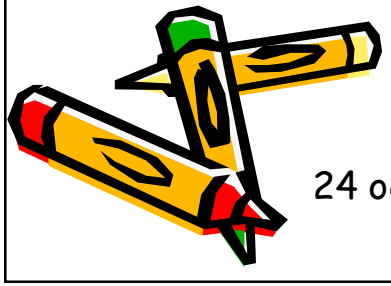
Pourquoi une informatique répartie ?

- La répartition est un état de fait pour un nombre important d'applications
- Besoins propres des applications
  - Intégration d'applications existantes initialement séparées
  - Intégration massive de ressources
    - Grilles de calcul, gestion de données
  - Pénétration de l'informatique dans des domaines nouveaux d'application
    - Intégration d'objets du monde réel
    - Surveillance et commande d'installations



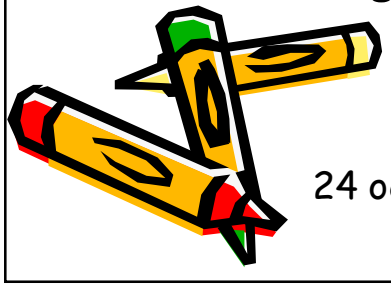
Pourquoi une informatique répartie ?

- Possibilités techniques
  - Coût et performances des machines et des communications
  - Interconnexion généralisée
    - Exemple 1 : interpénétration informatique-télécom-télévision
    - Exemple 2 : Réseaux de capteurs



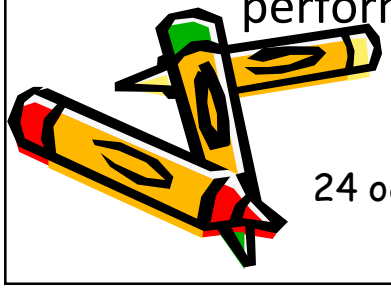
Pourquoi un système réparti ?

- Partage des ressources (données, applications, périphériques chers). Optimisation de leur utilisation (ex : au LIP6, machines et processeurs libres 69% et 93% du temps).
- Tolérance aux pannes (fiabilité, disponibilité).
- Contraintes physiques (ex : avionique, usines automatisées).
- Interconnexion de machines dédiées (ex : MVS-CICS + PC windows).
- Facilite la communication entre utilisateurs.
- Concurrence, parallélisme ==⇒efficacité.
- Prix des processeurs de petite puissance inférieur à ceux de grande
- puissance ==⇒raisons économiques.
- Flexibilité, facilité d'extension du système (matériels, logiciels). Sauvegarde de l'existant.



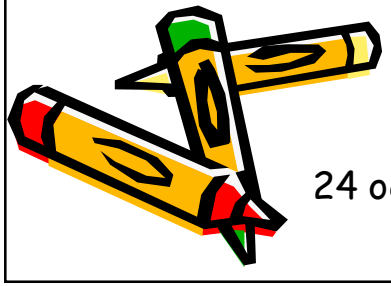
Ce qu'offre un système réparti

- **Transparence à la localisation ⟹désignation.** L'utilisateur ignore la situation géographique des ressources. Transparence à la migration.
- **Transparence d'accès.** L'utilisateur accède à une ressource locale ou distante d'une façon identique.
- **Transparence à l'hétérogénéité**. L'utilisateur n'a pas à se soucier des différences matérielles ou logicielles des ressources qu'il utilise. **Notion d'interopérabilité.**
- **Transparence aux pannes** (réseaux, machines, logiciels). Les pannes et réincarnations sont cachées à l'utilisateur. Transparence à la réplication.
- **Transparence à l'extension des ressources**. Extension ou réduction du système sans occasionner de gène pour l'utilisateur (sauf performance).



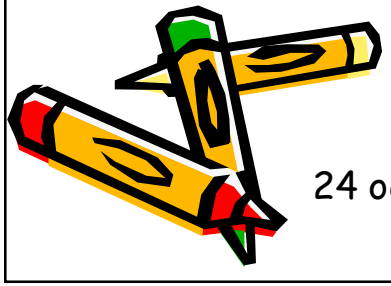
Inconvénients

- Très peu de logiciels existent sur le marché.
- Le réseau peut très vite saturer.
- La sécurisation des données sensibles est compliquée.
- La mise en œuvre est difficile.



Définitions -1-

- Un système à plusieurs processeurs n'est pas forcément un système réparti
- Qu'est-ce qu'un système réparti, distribué, parallèle ?
- Classification de flynn [1972]:
- On différencie les systèmes sur la base du flux d'instructions et de données.
  - SISD : PC monoprocesseur
  - SIMD : machines vectorielles
  - MISD : pipeline
  - MIMD : machines multiprocesseurs faiblement et fortement couplées (systèmes parallèles, systèmes distribués, systèmes d'exploitation réseaux)



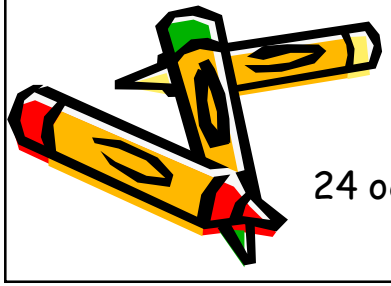
Définitions -1-

- ***MIMD à mémoire partagée***

  Les processeurs ont accès à la mémoire comme un espace d'adressage global. Tout changement dans une case mémoire est vu par les autres CPU. La communication inter-CPU est effectuée via la mémoire globale.

- ***MIMD à mémoire distribuée***

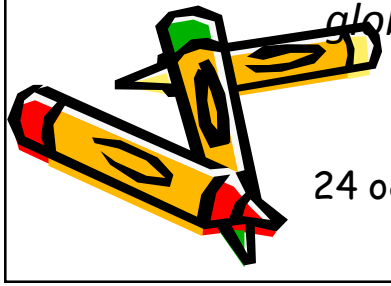
  Chaque CPU a sa propre mémoire et son propre système d'exploitation. Ce second cas de figure nécessite un middleware pour la synchronisation et la communication. Un système MIMD hybride est l'architecture la plus utilisée par les super ordinateurs. Ces systèmes hybrides possèdent l'avantage d'être très extensibles, performants et à faible coût.




QUESTIONS
?

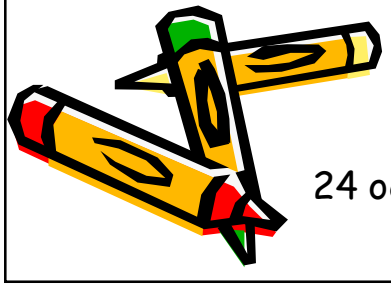
Définitions -2-

- Définition d'un système réparti
  - Ensemble composé d'éléments reliés par un système de communication; les éléments ont des fonctions de traitement (processeurs), de stockage (mémoire), de relation avec le monde extérieur (capteurs, actionneurs)
  - Les différents éléments du système ne fonctionnent pas indépendamment mais collaborent à une ou plusieurs tâches communes.
  - Conséquence : une partie au moins de l'état global du système est partagée entre plusieurs éléments (sinon, on aurait un fonctionnement indépendant)
  - *De manière plus précise : toute expression de la spécification du système fait intervenir plusieurs éléments (exemple : préserver un invariant global, mettre des interfaces en correspondance, etc.)*



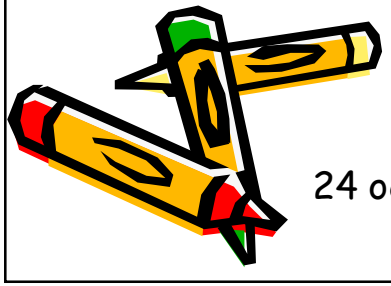
Définitions -2-

- Un système réparti est un ensemble de machines autonomes connectées par un réseau, et équipées d'un logiciel dédié à la coordination des activités du système ainsi qu'au partage de ses ressources." Coulouris et al. [COU 94].
- Un système réparti est un système qui s'exécute sur un ensemble de machines sans mémoire partagée, mais que pourtant l'utilisateur voit comme une seule et unique machine." Tanenbaum [TAN 94].
- Systèmes fortement et faiblement couplés.
- Notion d'**image unique du système ⟹système** généralement incomplet.



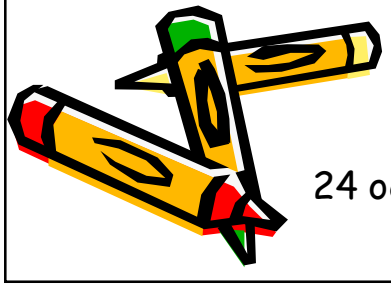
Définitions -2-

- Un système réparti est un ensemble de sites reliés par un réseau, comportant chacun une ou plusieurs machines.
- Un système réparti est un système qui vous empêche de travailler quand une machine dont vous n'avez jamais entendu parler tombe en panne « Lamport »
- Du point de vue utilisateur, un système réparti se comporte comme un système traditionnel, mais s'exécute sur de multiples unités indépendantes « Tanenbaum »



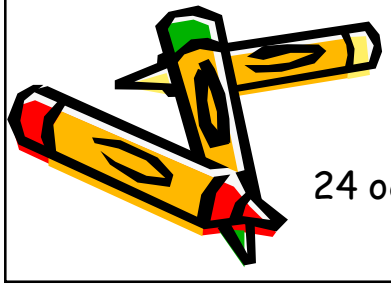
Définitions -2-

- Un système d'exploitation réparti fournit et contrôle l'accès aux ressources du système réparti.
- Un système d'exploitation parallèle contrôle l'exécution de programmes sur une machine parallèle (multiprocesseurs).
- Un système d'exploitation de réseaux fournit une plate forme de machines reliées par un réseau chacune exécutant son propre système d'exploitation.

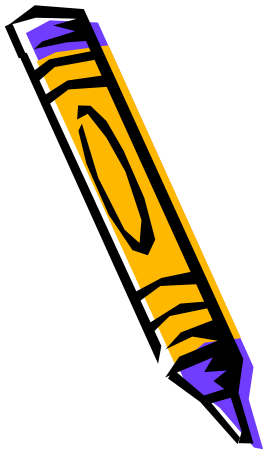


Exemples de système réparti

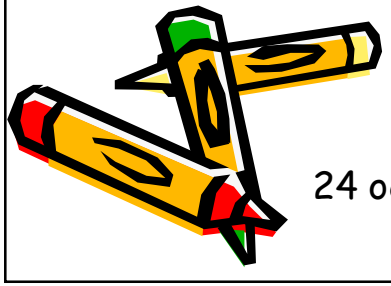
- WWW, FTP, Mail.
- Guichet de banque, agence de voyage.
- Téléphones portables(et bornes).
- Télévision interactive.
- Agents intelligents.
- Robots footballeurs.



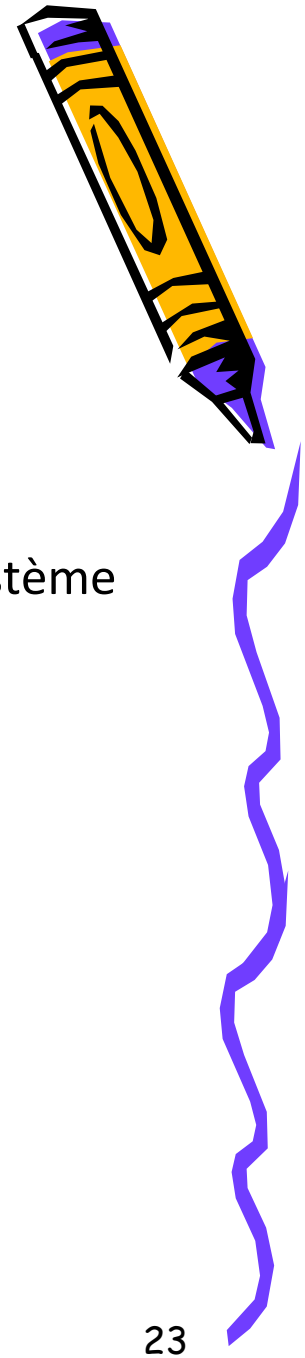
Intérêts des systèmes Répartis



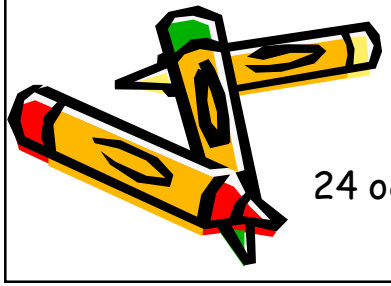
- Utiliser et partager des ressources distantes
- Système de fichiers : utiliser ses fichiers à partir de n'importe quelle machine
- Imprimante : partagée entre toutes les machines
- Optimiser l'utilisation des ressources disponibles
- Calculs scientifiques distribués sur un ensemble de machines
- Système plus robuste
- Duplication pour fiabilité : deux serveurs de fichiers dupliqués, avec sauvegarde
- Plusieurs éléments identiques pour résister à la montée en charge ...



Inconvénients/points faibles

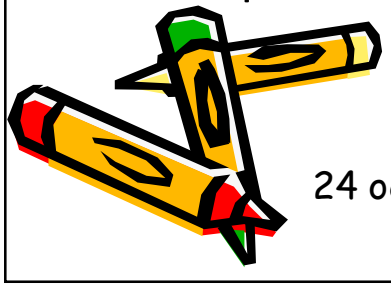


- Si problème au niveau du réseau
  - Le système marche mal ou plus du tout
- Bien souvent, un élément est central au fonctionnement du système : serveur
  - Si serveur plante : plus rien ne fonctionne
  - Goulot potentiel d'étranglement si débit d'information très important
- Sans élément central
  - Gestion du système totalement décentralisée et distribuée
  - Nécessite la mise en place d'algorithmes +/- complexes

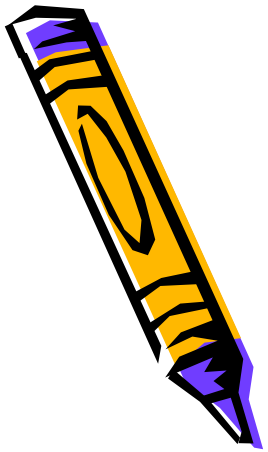


Particularités des systèmes Répartis

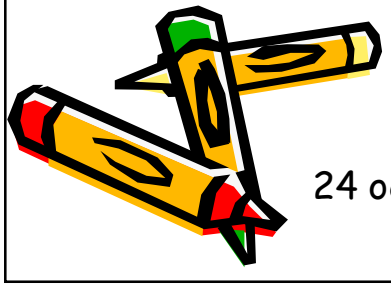
- Système Réparti = éclaté
- Connaissance des éléments formant le système : besoin d'identification et de localisation
- Gestion du déploiement et de la présence d'éléments essentiels
- Communication à distance est centrale
- Techniques et protocoles de communication
- Contraintes du réseau : fiabilité (perte de données) et temps de propagation (dépendant du type de réseau et de sa charge)
- Naturellement concurrent et parallèle
- Chaque élément sur chaque machine est autonome
- Besoin de synchronisation, coordination entre éléments distants et pour l'accès aux ressources (exclusion mutuelle ...)



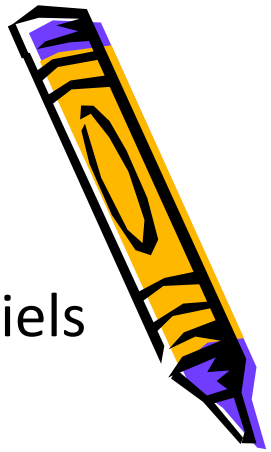
Particularités des systèmes Répartis



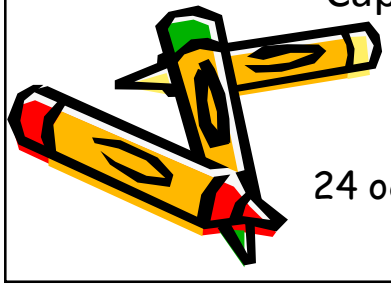
- Hétérogénéité
  - Des machines utilisées (puissance, architecture matérielle...)
  - Des systèmes d'exploitation tournant sur ces machines
  - Des langages de programmation des éléments logiciels formant le système
  - Des réseaux utilisés : impact sur performances, débit, disponibilité ...
    - Réseau local rapide
    - Internet
    - Réseaux sans fil



Fiabilité des systèmes Répartis

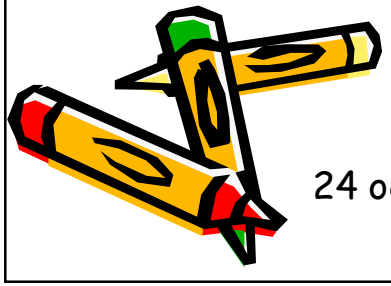


- Nombreux points de pannes ou de problèmes potentiels
  - **Réseau**
    - Une partie du réseau peut-être inaccessible
    - Les temps de communication peuvent varier considérablement selon la charge du réseau
    - Le réseau peut perdre des données transmises
  - **Machine**
    - Une ou plusieurs machines peut planter, engendrant une paralysie partielle ou totale du système
- Peut augmenter la fiabilité par redondance, duplication de certains éléments
  - Mais rend plus complexe la gestion du système
- Tolérance aux fautes
  - Capacité d'un système à gérer et résister à un ensemble de problèmes

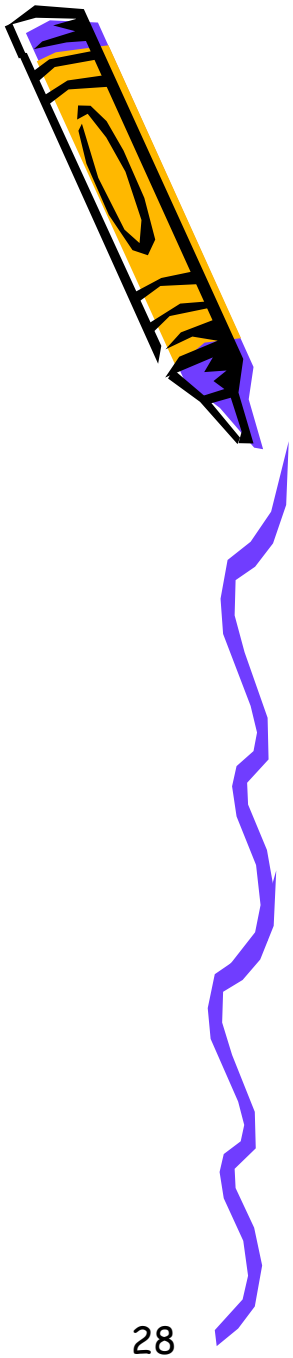


Sécurité des systèmes Répartis

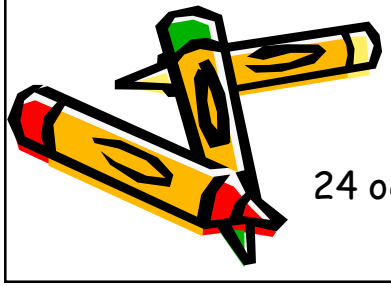
- Nature d'un système distribué fait qu'il est beaucoup plus sujet à des attaques
- Communication à travers le réseau peuvent être interceptées
- On ne connaît pas toujours bien un élément distant avec qui on communique
- Solutions
  - Connexion sécurisée par authentification avec les éléments distants
  - Cryptage des messages circulant sur le réseau



Sécurité et authentification

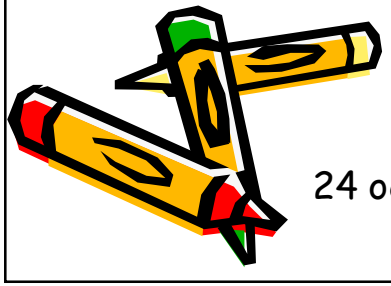


- Confidentialité.
- Intégrité :
  - Droits d'accès, Pare-Feu.
- Authentification :
  - Identification des applications partenaires.
  - Non-répudiation.
  - Messages authentifiés.
- Combien de personnes utilisent l'application, qui sont-ils ?
  - Nécessité de se protéger contre les intrusions.
  - Nécessité de stocker les accès des clients dans des fichiers journaux



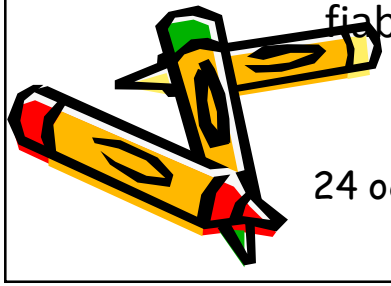
Transparences

- Fait pour une fonctionnalité, un élément d'être invisible ou caché à l'utilisateur ou un autre élément formant le système Réparti
  - Devrait plutôt parler d'opacité dans certains cas ...
- But est de cacher l'architecture, le fonctionnement de l'application ou du système distribué pour apparaître à l'utilisateur comme une application unique cohérente
- L'ISO définit plusieurs transparences (norme RM-ODP)
  - Accès, localisation, concurrence, réplication, mobilité, panne, performance, échelle

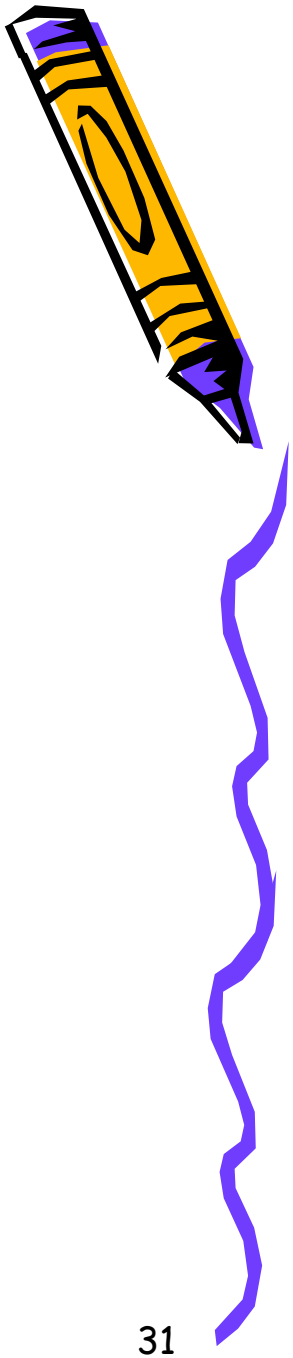


Transparences

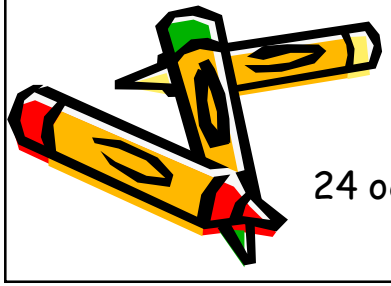
- Transparence d'accès
  - Accès à des ressources distantes aussi facilement que localement
  - Accès aux données indépendamment de leur format de représentation
- Transparence de localisation
  - Accès aux éléments/ressources indépendamment de leur localisation
- Transparence de concurrence
  - Exécution possible de plusieurs processus en parallèle avec utilisation de ressources partagées
- Transparence de réplication
  - Possibilité de dupliquer certains éléments/ressources pour augmenter la fiabilité



Transparences

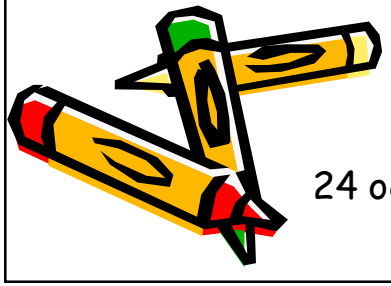


- Transparence de mobilité
  - Possibilité de déplacer des éléments/ressources
- Transparence de panne
  - Doit supporter qu'un ou plusieurs éléments tombe en Panne
- Transparence de performance
  - Possibilité de reconfigurer le système pour en augmenter les performances
- Transparence d'échelle
  - Doit supporter l'augmentation de la taille du système (nombre d'éléments, de ressources ...)



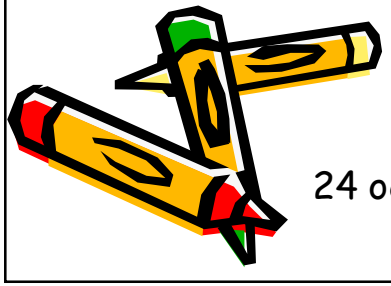
Transparences

- Un système donné va offrir un certain nombre de transparences
  - Souvent au minimum transparences de localisation, d'accès et de concurrence
- Système distribué ouvert
  - Peut être étendu en nombre d'éléments matériels le constituant
  - Possibilité d'ajouts de nouveaux services ou de réimplémentation de services existants au niveau logiciel
    - Fonctionnement se base sur des interfaces d'interactions clairement définies



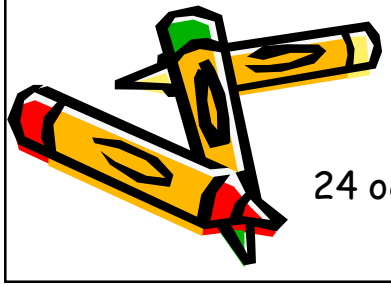
Communication

- Problème : communication inter-processus
  - dans un système centralisé : communiquer par la mémoire partagée
  - dans un système distribué : en général, l'absence de la mémoire partagée, on doit communiquer par messages avec des protocoles
- Protocole
  - ensemble de règles qui gère les communication inter-processus dans un système distribué
  - il détermine un l'accord sur la façon dont les communications doivent d'effectuer

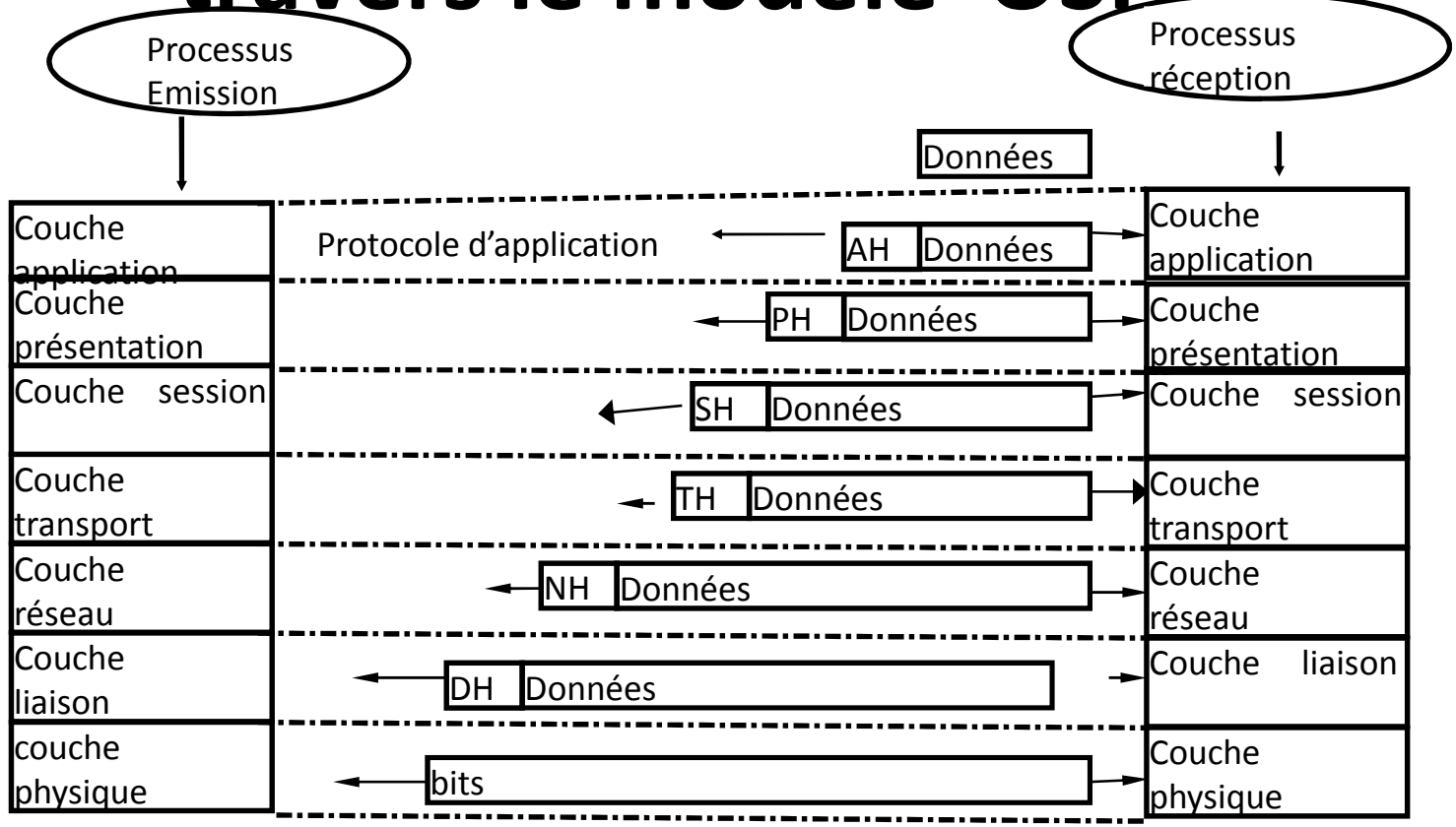


Communication

- Système Réparti
  - Ensemble d'entités logicielles communiquant entre-elles
- Entités logicielles s'exécutent sur des machines reliées entre elles par un réseau
- Communication entre entités logicielles
  - Le plus basique : directement en appelant les services des couches TCP ou UDP
  - Plus haut niveau : définition de couches offrant des services plus complexes
    - Couche réalisée en s'appuyant sur les couches TCP/UDP
    - Exemple de service : appel d'une procédure chez une entité distante
    - Notion de middleware (intergiciel)

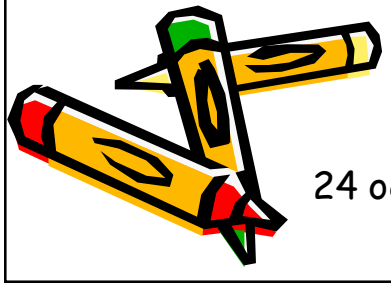


Communication de données a travers le modèle OSI

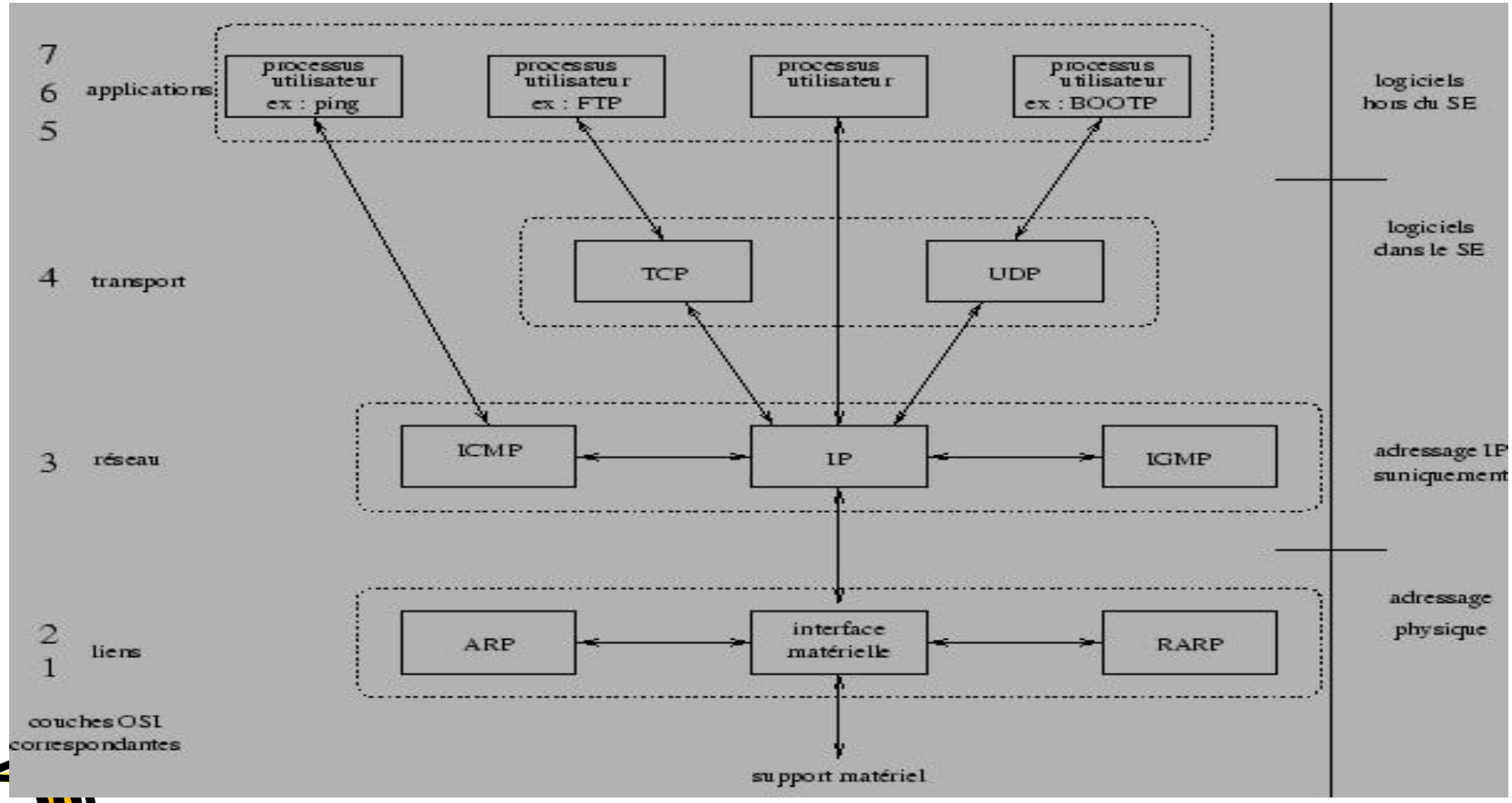


Rappels des protocoles OSI

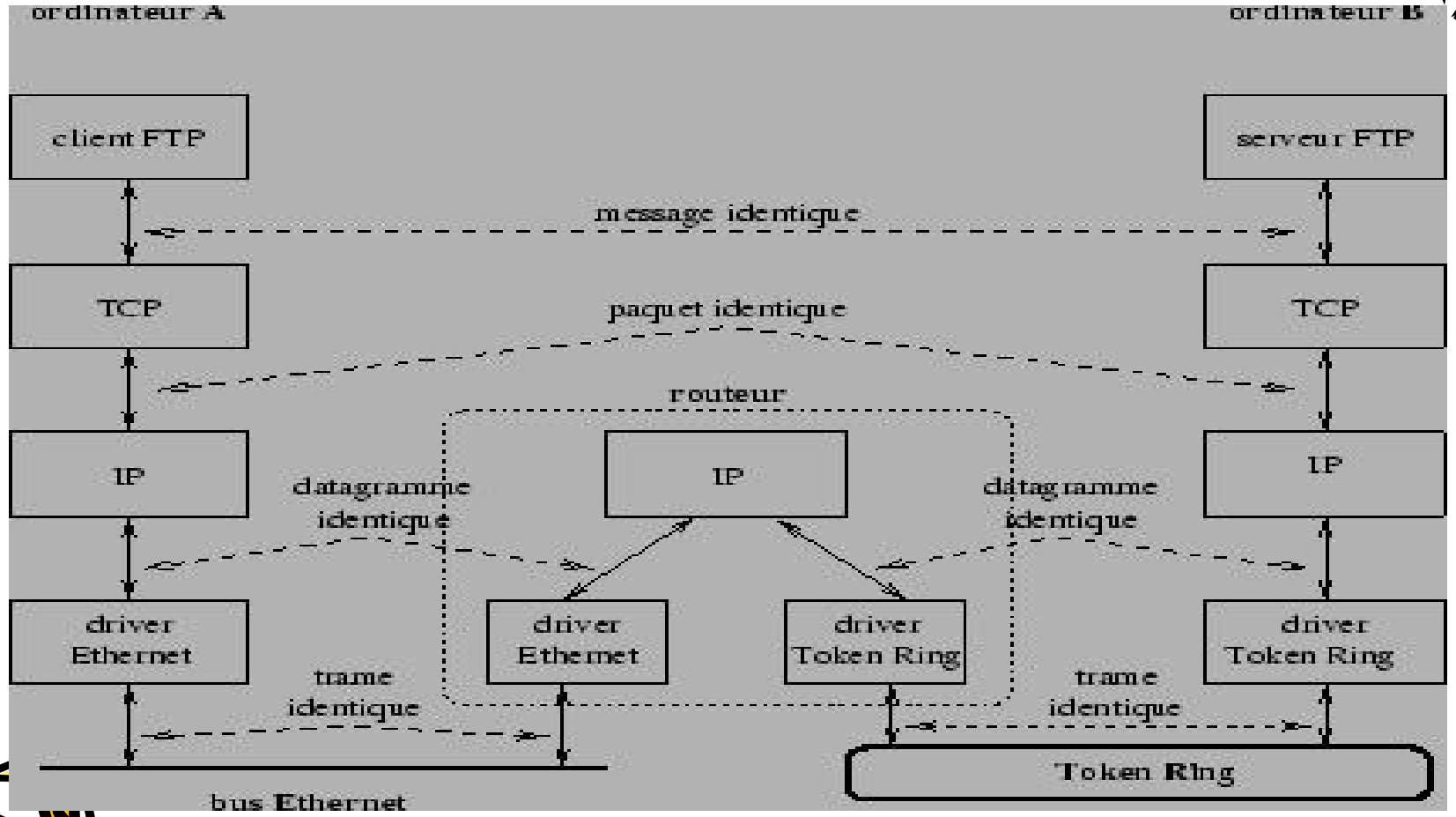
- Couche physique : assurer la transmission : connecteurs physique, codages physiques adaptation au mode de transmission, ...
- Couche de liaison : assurer la transmission sans erreur entre deux extrémistes d'une liaison directe
- Couche réseau : assurer le routage des messages sur les réseaux d'interconnexion (gestion de congestion, choix de chemins, ...
- Couche transport : assurer une transmission fiable entre l'expéditeur et le destinataire contrôle de perte des paquets, l'ordre des paquets, ...
- Couche de session : extension de la couche transport : mécanismes de synchronisation de reprise sur panes, ... très peu utilisée en pratique !!!
- -Couche de présentation : format de données, adaptation au terminal
- - Couche d'application: protocoles applicatifs : courrier électronique (X400), ftp, telnet,...



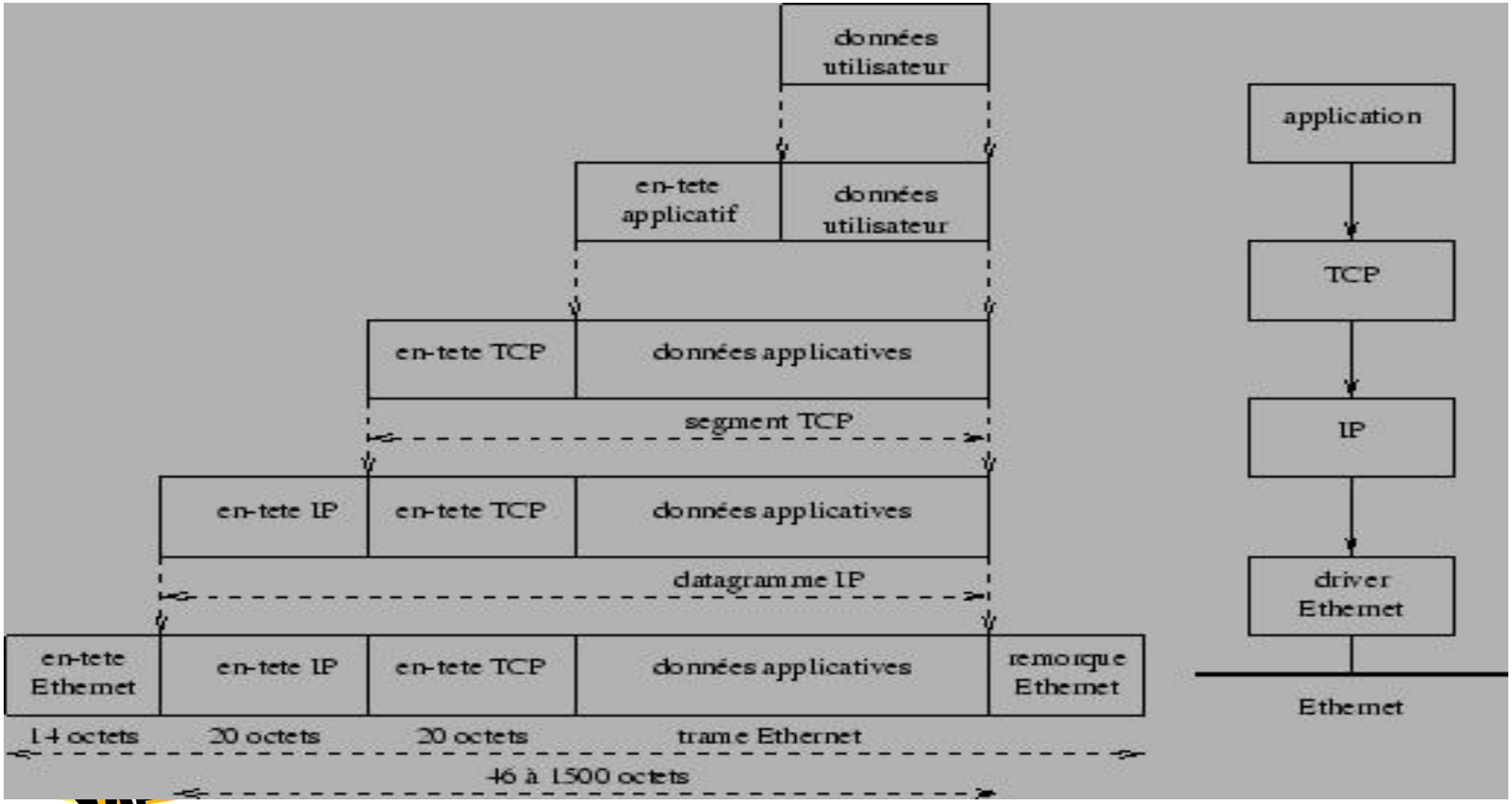
Architecture des protocoles TCP/IP



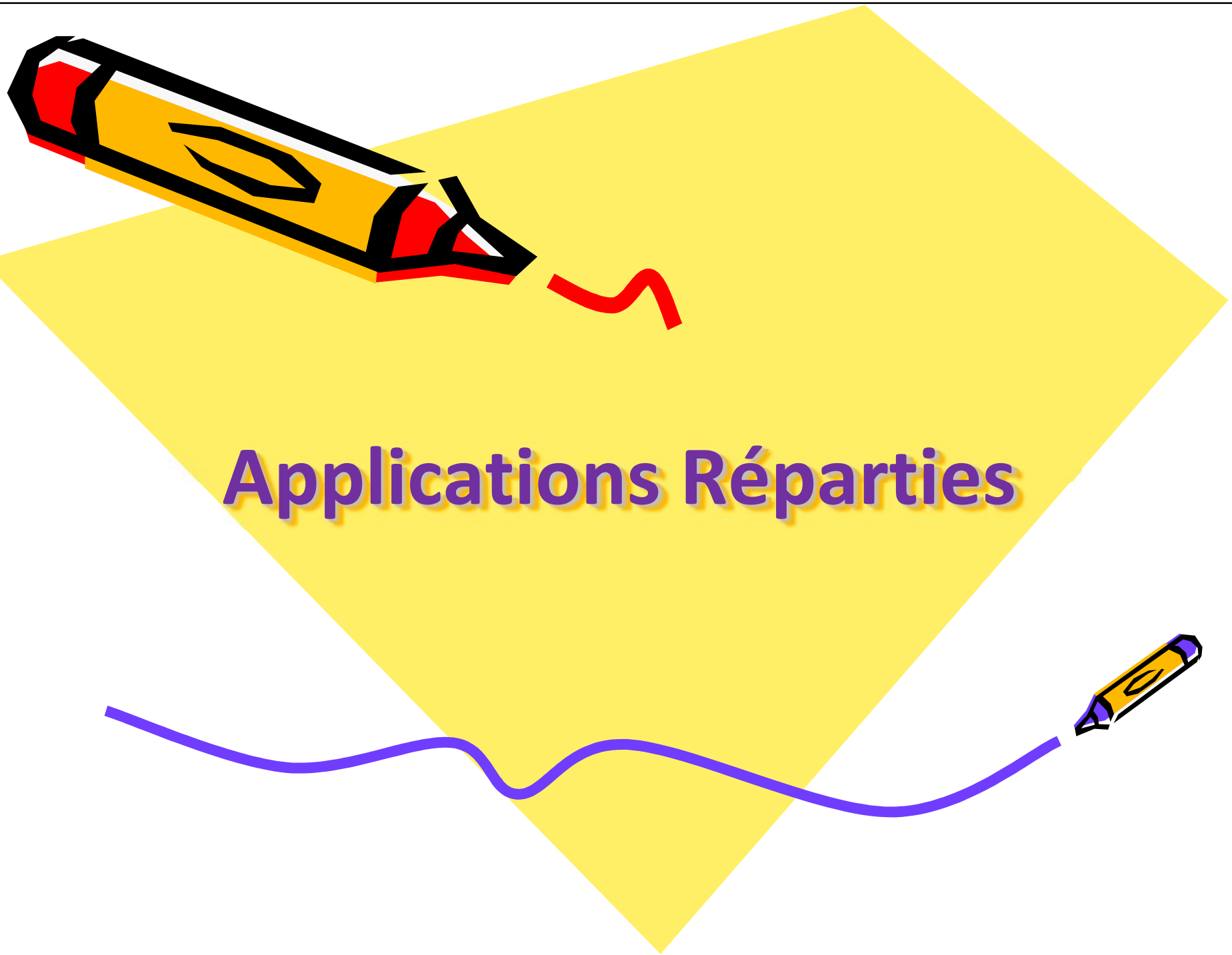
Interconnexion de deux réseaux



Encapsulation des données par la pile des protocoles TCP/IP

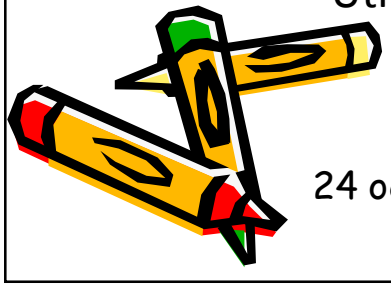



QUESTIONS
?


Applications Réparties

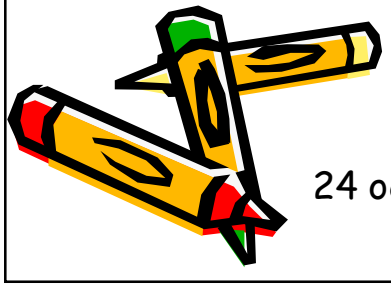
Système *vs* application

- **Système : Gestion des ressources communes et de l'infrastructure, lié de manière étroite au matériel sous-jacent**
  - **Système d'exploitation :** gestion de chaque élément
  - **Système de communication :** échange d'information entre les éléments
  - **Caractéristiques communes :** cachent la complexité du matériel et des communications, fournissent des services communs de plus haut niveau d'abstraction
- **Application : réponse à un problème spécifique, fourniture** de services à ses utilisateurs (qui peuvent être d'autres applications).
  - Utilise les services généraux fournis par le système

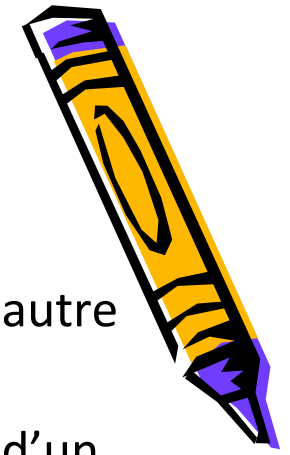


Modèles d'exécution

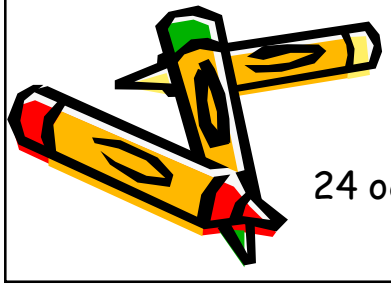
- Modèle Client-Serveur
- Modèle de communication par messages
- Modèle de communication par événements
- Modèle à base de composants
- Modèle à base d'agents mobiles
- Modèles à mémoires partagées
- Modèle orienté service



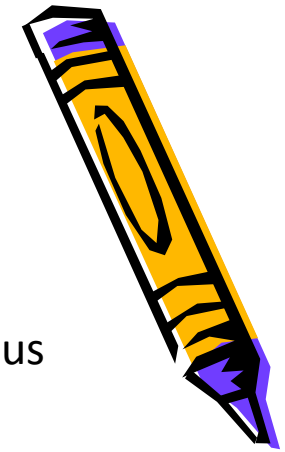
Modèle Client-Serveur 1/5



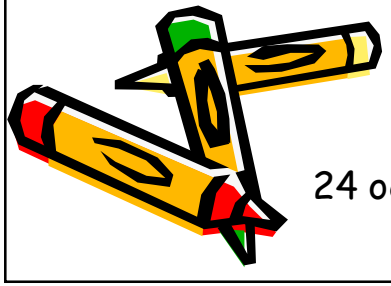
- **Client :** processus demandant l'exécution d'une opération à un autre processus
- **Serveur :** processus accomplissant une opération sur demande d'un client, et lui transmettant le résultat
- **Modèle Requête/Réponse :**
  - **Requête :** message transmis par un client à un serveur décrivant l'opération à exécuter
  - **Réponse** : message transmis par un serveur à un client suite à l'exécution d'une opération, contenant le résultat de l'opération
- Mode de communication **synchrone**
  - Le client envoie la requête et attend la réponse.
  - Emission **bloquante**



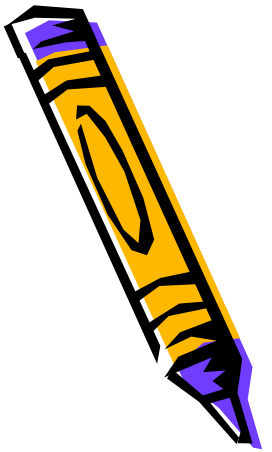
Le modèle Client-Serveur (2/5)



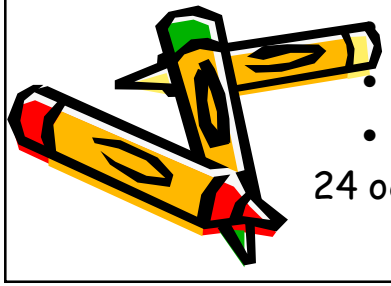
- 2 rôles distincts
  - Client : demande que des requêtes ou des services lui soient rendus
  - Serveur : répond aux requêtes des clients
- Interaction
  - Message du client vers le serveur pour faire une requête
  - Exécution d'un traitement par le serveur pour répondre à la requête
  - Message du serveur vers le client avec le résultat de la requête
- Exemple : serveur Web
  - Client : navigateur Web de l'utilisateur
  - Requêtes : récupérer le contenu d'une page HTML gérée ou générée par le serveur



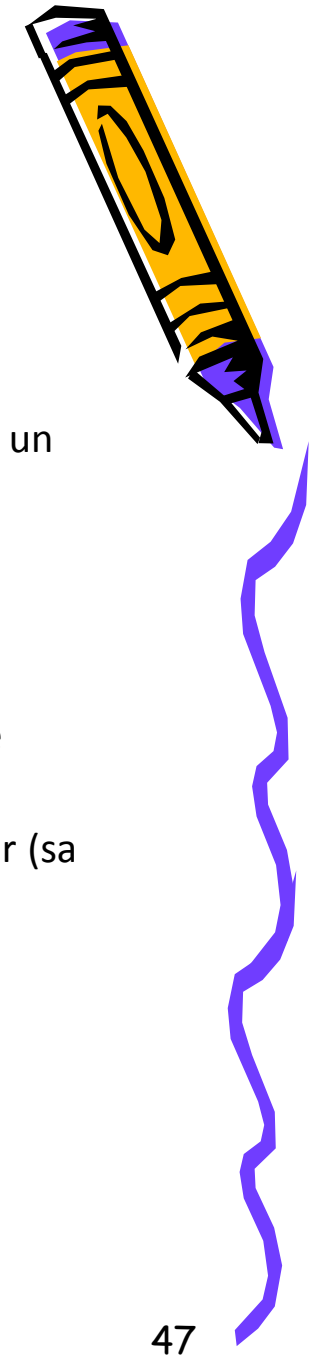
Le modèle Client-Serveur (3/5)



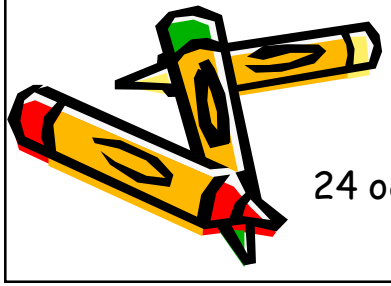
- Coté serveur :
  - Externalisation de services.
  - Attente de requêtes en provenances de clients puis exécution des requêtes en séquentiel ou en parallèle.
  - Interface : Skeleton
    - reçoit l'appel sous forme de « stream»
    - décapsule les paramètres
    - demande l'exécution
    - renvoi les paramètres (par références) et les résultats
- Coté client :
  - Émission de requêtes puis attente de la réponse.
  - Initiateur du dialogue.
  - Interface : Stub
    - Reçoit l'appel en local
    - encapsule les paramètres
    - attends les résultats du serveur
    - décapsule les résultats
    - redonne la main à la fonction appelante



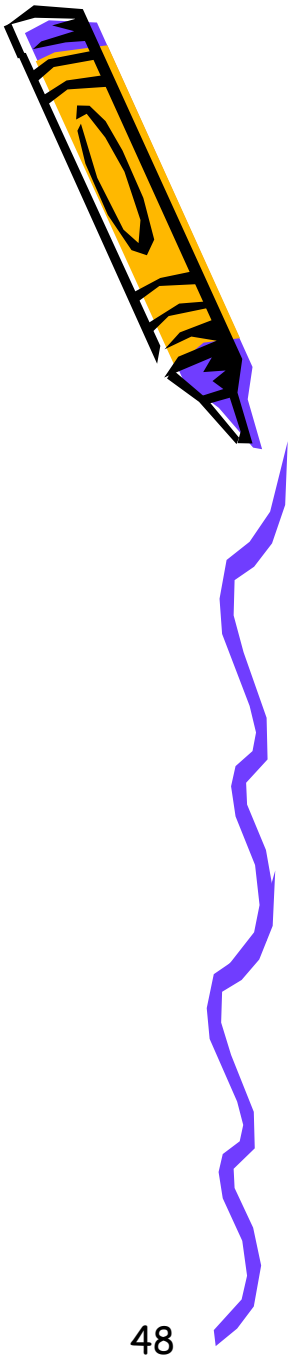
Le modèle Client-Serveur (4/5)



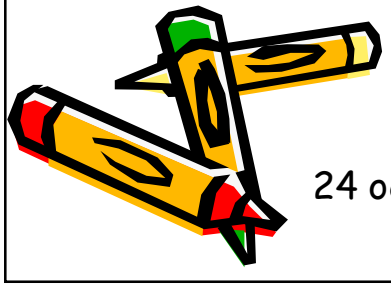
- Modèle le plus répandu
  - Fonctionnement simple
  - Abstraction de l'appel d'un service : proche de l'appel d'une opération sur un élément logiciel
    - Interaction de base en programmation
- Particularités du modèle
  - Liens forts entre le client et le serveur
  - Un client peut aussi jouer le rôle de serveur (et vice-versa) dans une autre interaction
  - Nécessité généralement pour le client de connaître précisément le serveur (sa localisation)
  - Ex : URL du site Web
  - Interaction de type « 1 vers 1 »



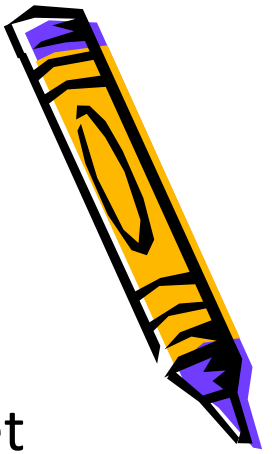
Le modèle Client-Serveur (5/5)



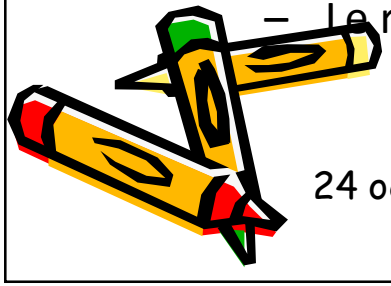
- Client/Serveur« traditionnel » :
  - RPC
- Client/Serveur« à objets » :
  - RMI, CORBA, DCOM
- Client/Serveur« de données » :
  - Requêtes SQL
- Client/Serveur« WEB » :
  - CGI, Servlet, asp, jsp, php,...



Modèle de communication par Messages (1/2)

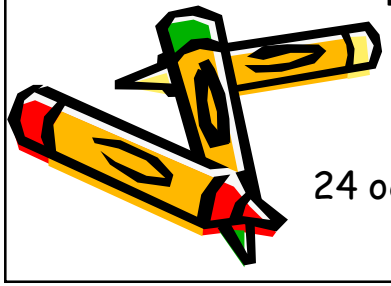


- Une application produit des messages **(producteur)** et une application les consomme **(consommateur)**
  - Le producteur (l'**émetteur**) **et le consommateur (récepteur)** ne communiquent **pas directement entre eux mais utilisent** un objet de communication intermédiaire (**boîte aux lettres** ou **file d'attente**)
- Communication **asynchrone :**
  - Les deux composants n'ont pas besoin d'être connectés en même temps grâce au système de file d'attente
  - Emission **non bloquante:** l'entité émettrice émet son message, et continue son traitement sans attendre que le récepteur confirme l'arrivée du message.
  - Le récepteur récupère les messages **quand il le souhaite.**



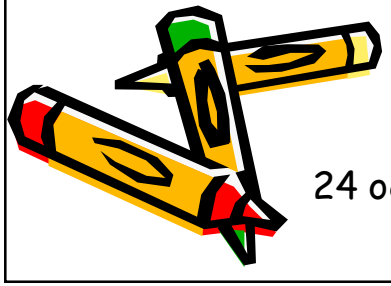
Modèle de communication par Messages (2/2)

- Adapté à un système réparti dont les éléments en interaction sont **faiblement couplés :**
  - éloignement géographique des entités communicantes
  - possibilité de déconnexion temporaire d'un élément
- Deux modèles de communication :
  - **Point à point**:
    - Les messages ne sont lus que par un seul consommateur.
    - Une fois qu'un message est lu, il est retiré de la file d'attente.
  - **Multi-points :**
    - Le message est diffusé à tous les éléments d'une liste de destinataires
- Exemple: messagerie électronique



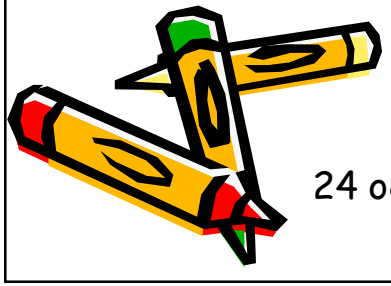
Modèle de communication par événements (1/2)

- Concepts de base
  - Evénement = changement d'état survenant de manière asynchrone (par rapport à ses “clients”)
  - Réaction = exécution d’une opération prédéfinie liée à l’événement
- Mode de communication **asynchrone et anonyme**
  - indépendance entre l’émetteur (producteur) et les destinataires (consommateurs) d’un événement
- Modèle **Publish/Subscribe (par abonnement) :**
  - les applications consommatrices des messages s'abonnent à un **topic (sujet)**
  - Les messages envoyés à ce topic restent dans la file d'attente jusqu'à ce que toutes les applications abonnées aient lu le message

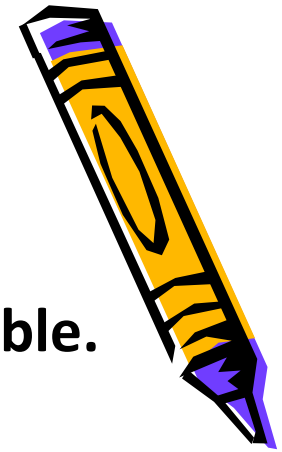


Modèle de communication par événements (2/2)

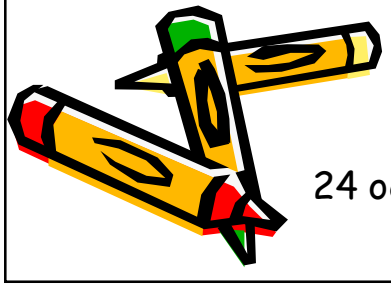
- Deux modes de consommation des messages:
  - **«Mode Pull»** - réception explicite : les clients viennent prendre régulièrement leurs messages
  - **«Mode Push»** - délivrance implicite : une méthode prédéfinie est attachée à chaque type de message et elle est appelée automatiquement à chaque occurrence de l'évènement. la réception d'un événement entraîne l'exécution de la réaction associée
- Exemple : surveillance des systèmes (changement d'états de configuration, alertes, statistiques



Modèle à base de composants

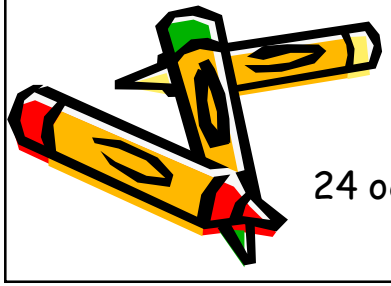


- **Composant : Module logiciel Autonome et Réutilisable.**
- Caractéristiques d'un composant :
  - des entrées/sorties déclarées pour permettre les connexions entre plusieurs composants
  - des propriétés déclarées permettant de configurer le composant
- Simplifier la conception et le développement des applications avec les composants:
  - **Réutilisabilité**
  - **Indépendance**
  - **Autonomie**



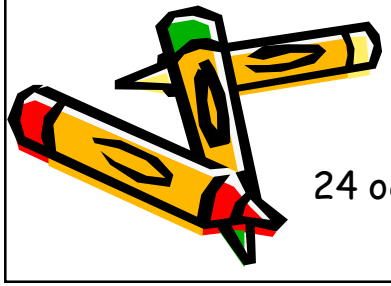
Modèle à base d'agents mobiles

- **Code mobile** : programme se déplaçant d'un site à un autre sur le réseau
- Exemple : les applets = programme exécutable inclut dans une page HTML et qui s'exécute sur le site (machine) qui télécharge la page.
- Avantage de la mobilité :
  - efficacité, privilégie les interactions locales
  - moins de communications effectuées pour les échanges
  - amener le code aux données plutôt que le contraire
- **Agents mobile** : entité logicielle permettant de construire des applications distribuées
  - peut se déplacer d'une machine à une autre sur réseau.
  - s'exécute sur une machine et peut, à tout moment, arrêter son exécution, se déplacer sur une autre machine et reprendre son exécution (à partir de son dernier état)



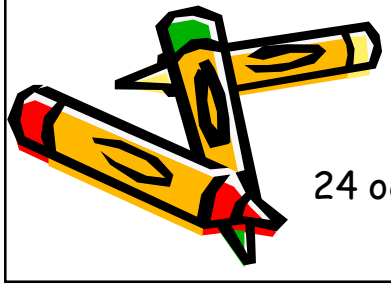
Modèles à Mémoire Partagée

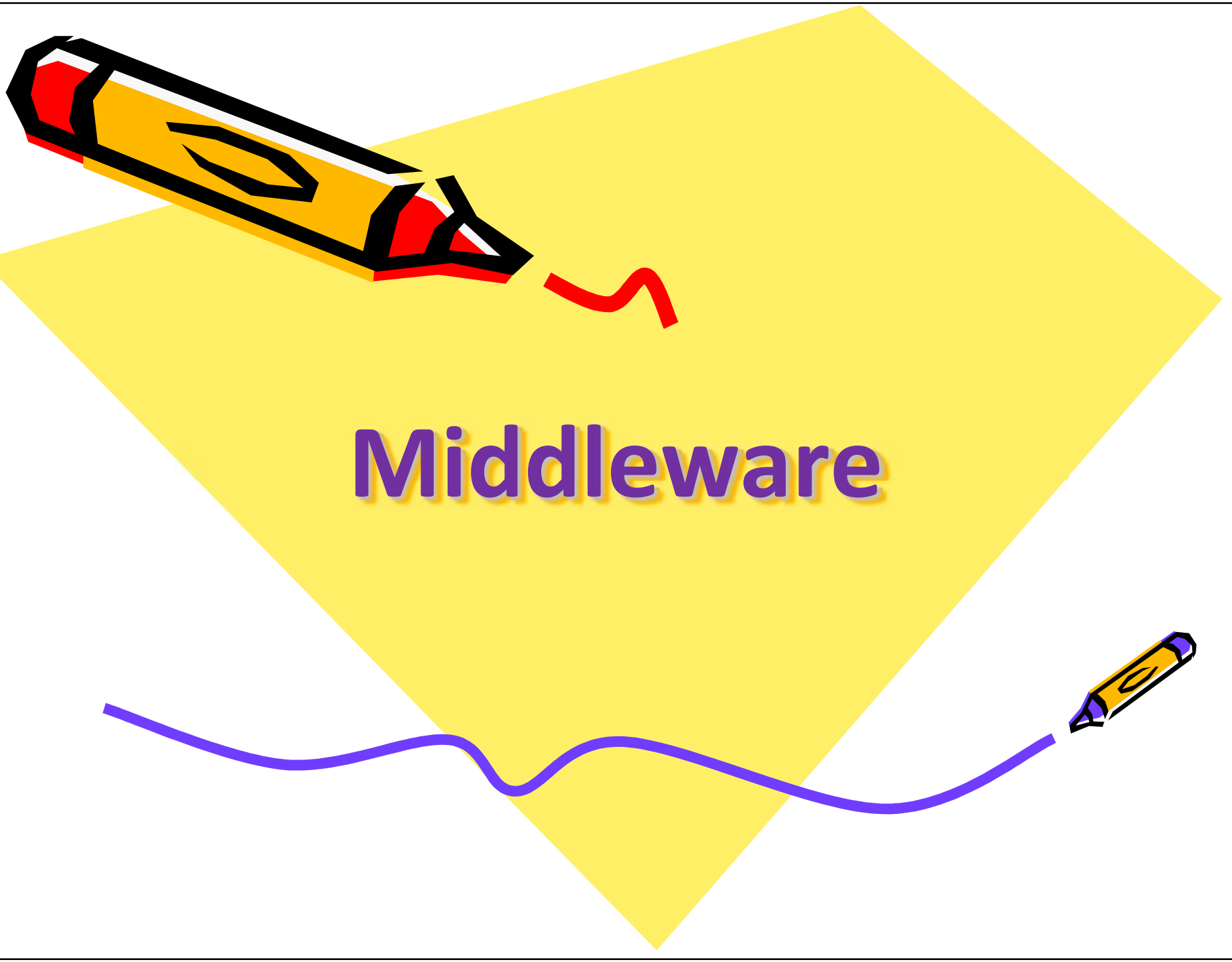
- **Objectif:**
  - Replacer le programmeur dans les conditions d'un système centralisé :
    - utiliser un espace mémoire commun pour les communications
- **Avantages:**
  - transparence de la distribution pour le développeur,
  - efficacité de développement (utilisation des paradigmes usuels de la programmation concurrente).
- **Inconvénient:**
  - Complexité de mise en œuvre efficace d'une mémoire partagée distribuée



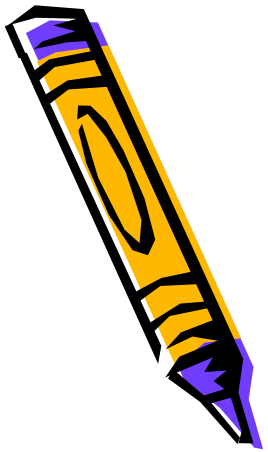
Modèle Orienté Service

- Un modèle d'interaction basé sur la notion de service
- Un service est un composant logiciel exécuté par :
  - Un producteur
  - A l'attention d'un consommateur
- Une nouvelle vision dans la conception des applications réparties

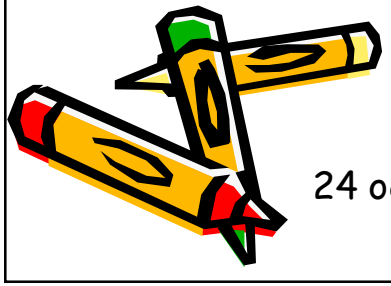



Middleware

Motivations

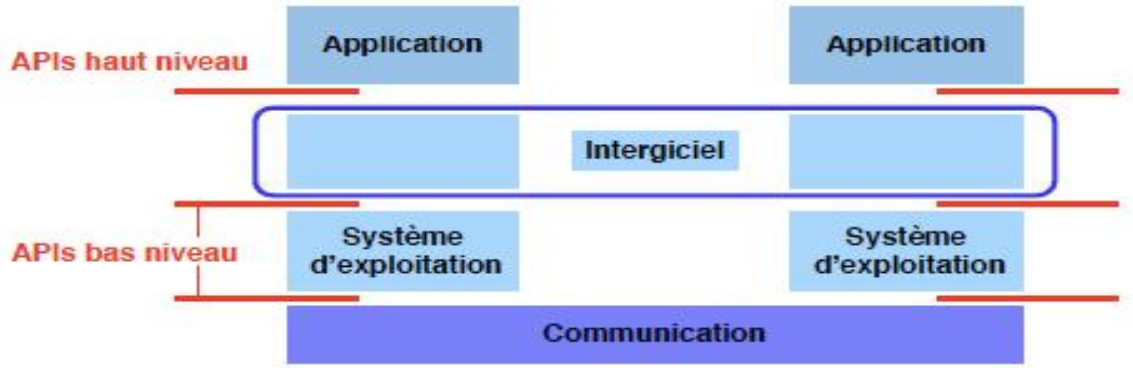


- l'interface fournie par les systèmes d'exploitation et de communication est encore trop complexe pour être utilisée directement par les applications:
  - Hétérogénéité (matérielle et logicielle)
  - Complexité des mécanismes (bas niveau)
  - Nécessité de gérer la répartition
- **Solution**
  - Introduire une couche logicielle **intermédiaire (répartie)** entre les **niveaux bas (systèmes et communication) et le niveau haut (applications) : c'est l'intergiciel (Middleware** en anglais)
  - L'intergiciel joue un rôle analogue à celui d'un "**super système d'exploitation" pour un système réparti**
- Un **middleware ou «!intergiciel!»** permet le dialogue entre les différents composants d'une application répartie
- Représente **l'élément le plus important de tout système réparti**

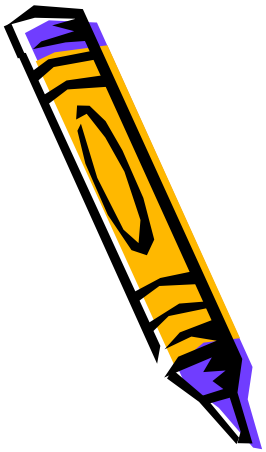


Middleware

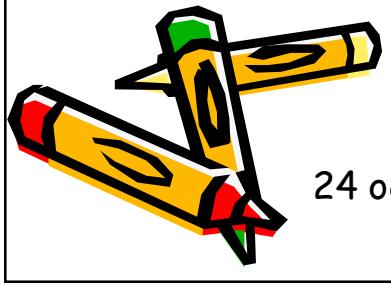
- Middleware ou intergiciel : couche logiciel
  - S'intercale entre le système d'exploitation/réseau et les éléments de l'application distribuée
  - Offre un ou plusieurs services de communication entre les éléments formant l'application ou le système distribué



But et Fonctionnalités d'un Middleware (1/3)

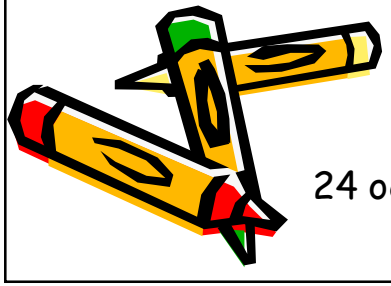


- Gestion de l'hétérogénéité
  - Langage de programmation, systèmes d'exploitation utilisés ...
- Offrir des abstractions de communication de plus haut niveau
  - Appel d'une procédure à distance sur un élément
  - Communication via une mémoire partagée
  - Diffusion d'événements
- Offrir des services de configuration et de gestion du système
  - Service d'annuaire pour connaître les éléments présents
  - Services de persistance, de temps, de transaction, de sécurité ...



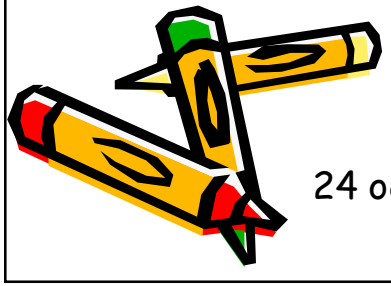
But et Fonctionnalités d'un Middleware (2/3)

- **Masquer l'hétérogénéité** (des machines, systèmes, protocoles de communication)
- **Fournir une API (*Application Programming Interface*)** de haut niveau
  - Permet de masquer la complexité des échanges
  - Facilite le développement d'une application répartie
- Rendre **la répartition** aussi invisible (transparente) que Possible
- Fournir des **services répartis** d'usage courant



But et Fonctionnalités d'un Middleware (3/3)

- **Communication**
  - Permet **la communication** entre machines mettant en œuvre des formats différents de données
  - Prise en charge par la **FAP (Format And Protocol)**
  - **FAP : pilote les échanges à travers le réseau :**
    - Synchronisation des échanges selon un **protocole** de communication
    - nées échangées selon un **format** connu de part et d'autre
- **Nommage**
  - permet d'identifier la machine serveur sur laquelle est localisé le service demandé afin d'en déduire le chemin d'accès.
  - fait, souvent, appel aux services d'un **annuaire.**
- **Sécurité**
  - permet de garantir la confidentialité et la sécurité des données à l'aide de mécanismes d'authentification et de cryptage des informations



FIN Partie 1