Introduction aux systèmes répartis

Babahenini Mohamed Chaouki

Université de Biskra

WEBLIOGRAPHIE

- Durant la préparation de ce cours j'ai eu recours aux supports disponibles sur le WEB et notamment au cours des personnes suivantes:
  - ***Gérard FLORIN****: professeur en informatique au CNAM de Paris.*
  - ***Michel RIVEILL*** *: professeur d'informatique à l'Ecole Polytechnique de l'Université de Nice*
  - ***Yahya SLIMANI*** *: professeur en informatique à la Faculté des Sciences de Tunis*
  - **Sacha KRAKOWIAK***: professeur émérite en informatique à l'université Joseph Fourier, Grenoble*
  - **Eric CARIOU:** *Maître de conférences au département informatique de l'université de Pau*

Système centralisé

- Tout est localisé sur la même machine
  - 1 processeur : une horloge commune
  - 1 mémoire centrale : un espace d'adressage commun
  - 1 système d'exploitation
    - Gestion centralisée des ressources
    - Etat global du système facilement reconstituable
- Accès local aux ressources

Emergence du réparti

- Evolution technologique
  - **machines**
    - de plus en plus **performantes** avec une baisse des prix
  - **Équipement réseau**
    - de plus en plus **rapides**
- Les progrès technologiques des ordinateurs et le haut débit permettent aux différentes applications sur des ordinateurs distincts (et distants) de coopérer pour effectuer des tâches coordonnées.

Pourquoi un système réparti ?

1. Partage des ressources (données, applications, périphériques chers). Optimisation de leur utilisation.
2. Tolérance aux pannes (fiabilité, disponibilité).
3. Adaptation de la structure d'un système à celle des applications (géographique ou fonctionnelle).
4. Interconnexion de machines dédiées (ex : MVS-CICS + PC windows).
5. Besoin de communication et de partage d'information.
6. Concurrence, parallélisme ==⇒efficacité.
7. Prix des processeurs de petite puissance inférieur à ceux de grande puissance ==⇒raisons économiques.
8. Flexibilité, facilité d'extension du système (matériels, logiciels). Sauvegarde de l'existant.

Définitions d'un SD

- "*Un ensemble de machines 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. »*

- "*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 »*

Définitions d'un SD

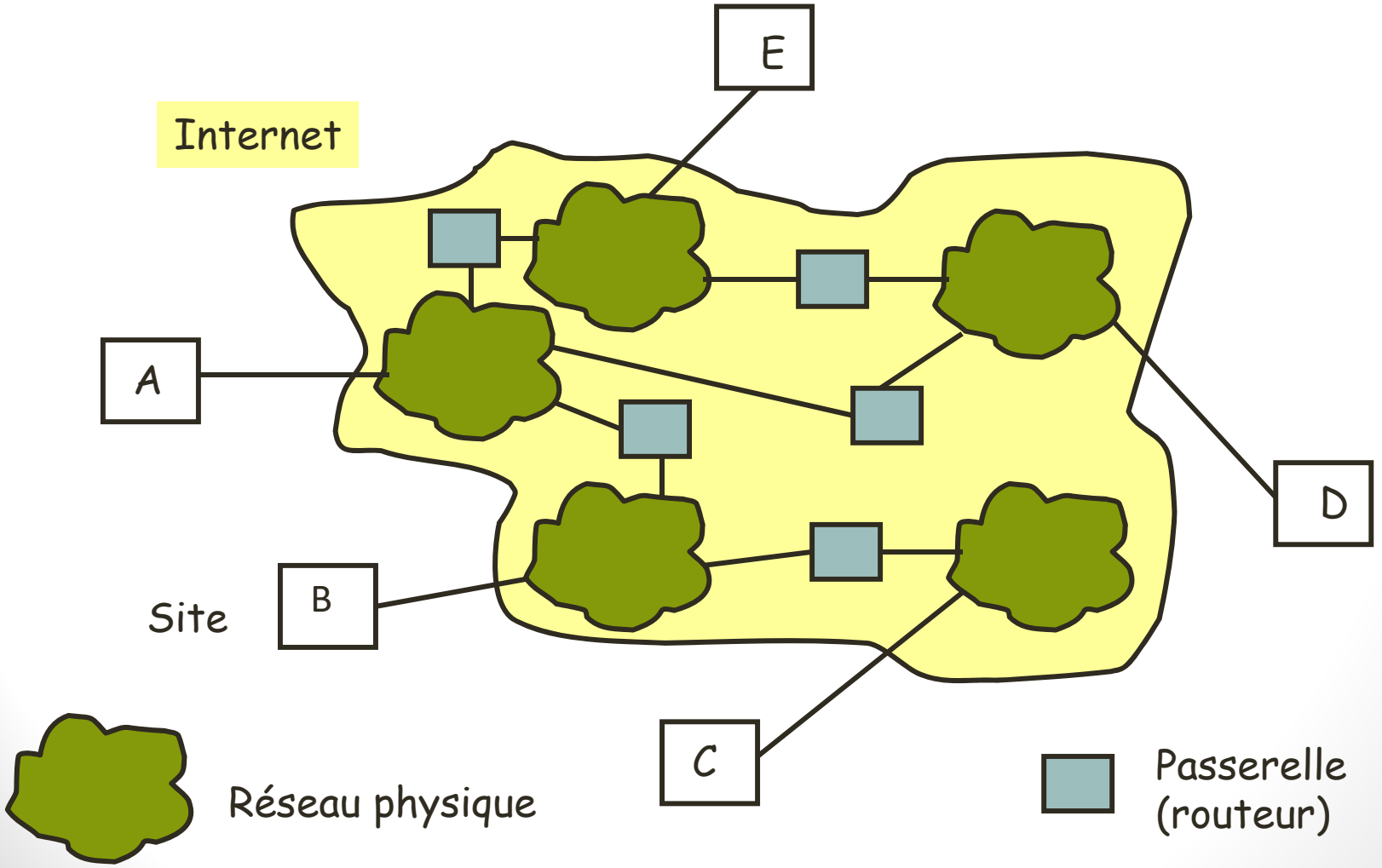
- Types de ressources
  - Calcul
  - Stockage
  - Electronique
    - capteur, satellites, scanneurs, …
- Architecture
  - Plusieurs processeurs ➔ plusieurs horloges
  - Plusieurs mémoires ➔ pas de mémoire partagée
  - Réseau d'interconnexion et de communication

Définitions d'un SD

- **Fonctionnement collaboratif**: des traitements coopérants sur des données réparties pour réaliser une tâche commune
  - La **coopération** entre les processus correspond à la communication entre eux et la synchronisation de leurs évolutions.
  - La **répartition** peut concerner les données comme les traitements (tâches).

Système réparti

- Fondé sur un réseau longue distance



Quelques domaines d'application des systèmes répartis

- CFAO, Ingénierie simultanée
  - Coopération d'équipes pour la conception d'un produit
  - Production coopérative de documents
  - Partage cohérent d'information
- Gestion intégrée des informations d'une entreprise
  - Intégration de l'existant
- Contrôle et organisation d'activités en temps réel
- Centres de documentation, bibliothèques
  - Recherche, navigation, visualisation multimédia
- Systèmes d'aide à la formation

Caractéristiques d'un SD

1. Centralisée = 1 horloge. Ordre total.

2. Répartie = plusieurs horloges non synchronisées + communications **asynchrones** ==⇒plus d'**état global** facilement calculable et présence d'**indéterminisme logique**.

- Absence d'état global
  - pas de référence spatiale commune à cause de l'absence (dans la majorité des cas) d'une mémoire partagée
  - Pas de référence temporelle commune à cause de l'existence de plusieurs processeurs ayant chacun sa propre horloge
- Existence d'un réseau
  - non géré par le système d'exploitation
  - le comportement du système réparti dépend de l'état du réseau

Exemples de systèmes distribués

- Serveur de fichiers
  - Accès aux fichiers de l'utilisateur quelque soit la machine utilisée
  - Machines du département informatique
    - Clients : scinfeXXX
    - Un serveur de fichier
    - Sur toutes les machines : /home/durand est le « home directory » de l'utilisateur *durand*
    - Physiquement : fichiers se trouvent uniquement sur le serveur
    - Virtuellement : accès à ces fichiers à partir de n'importe quelle machine cliente en faisant « croire » que ces fichiers sont stockés localement
    - Arborescence de fichiers Unix : arborescence unique avec
      - Répertoires physiquement locaux
      - Répertoires distants montés via le protocole NFS (Network File System)

Exemples de systèmes distribués

- Serveur de fichier (suite)
  - Intérêts
    - Accès aux fichiers à partir de n'importe quelle machine
    - Système de sauvegarde associé à ce serveur
    - Transparent pour l'utilisateur
  - Inconvénients
  - Si réseau ou le serveur plante : plus d'accès aux fichiers

Exemples de systèmes distribués

- Autre exemple de système distribué : Web
  - Un serveur web auquel se connecte un nombre quelconque de navigateurs web (clients)
  - Accès à distance à de l'information
    - Accès simple
      - Serveur renvoie une page HTML statique qu'il stocke localement
    - Traitement plus complexe
      - Serveur interroge une base de données pour générer dynamiquement le contenu de la page
    - Transparent pour l'utilisateur : les informations s'affichent dans son navigateur quelque soit la façon dont le serveur les génère

Exemples de systèmes distribués

- Calculs scientifiques
  - Plusieurs architectures matérielles généralement utilisées
    - Ensemble de machines identiques reliées entre elles par un réseau dédié et très rapide (cluster)
    - Ensemble de machines hétérogènes connectées dans un réseau local ou bien encore par Internet (grille)
  - Principe général
    - Un (ou des) serveur distribue des calculs aux machines clients
    - Un client exécute son calcul puis renvoie le résultat au serveur
  - Avantage
    - Utilisation d'un maximum de ressources de calcul
  - Inconvénient
    - Si réseau ou serveur plante, arrête le système

Ce qu'offre un système réparti

Transparence

- Propriété fondamentale : **Tout cacher à l'utilisateur**
  - La répartition doit être non perceptible : une ressource distante est accédée comme une ressource locale
  - Cacher l'architecture et le fonctionnement du système réparti
- L'ISO définit plusieurs transparences
  - accès, localisation, concurrence, réplication, mobilité, panne, performance, échelle

1. **Transparence à la localisation** ==⇒désignation. L'utilisateur ignore la situation géographique des ressources. Transparence à la migration.

2. **Transparence d'accès**. L'utilisateur accède à une ressource locale ou distante d'une façon identique.

Ce qu'offre un système réparti

3. **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. Sources d'hétérogénéité

- machines (architecture matérielle)
- systèmes d'exploitation
- langages de programmation
- protocoles de communications

4. **Transparence aux pannes** (réseaux, machines, logiciels). Les pannes et réincarnations sont cachées à l'utilisateur. Transparence à la réplication. ➔ Un système réparti doit pouvoir tolérer la panne des machines : (Une machine tombe en panne, Une machine envoie des informations erronées, Une machine n'est plus atteignable (problème réseau)).

5. **Transparence à l'extension des ressources**. Extension ou réduction du système sans occasionner de gène pour l'utilisateur (sauf performance).

Systèmes distribués vs parallèles

- **Systèmes Parallèles**. Une machine multiprocesseurs avec un environnement du type SIMD (tous les processeurs exécutent le même programme et ont une vision uniforme de l'état global du système).

Extensible à un réseau de machines asynchrones fortement couplées

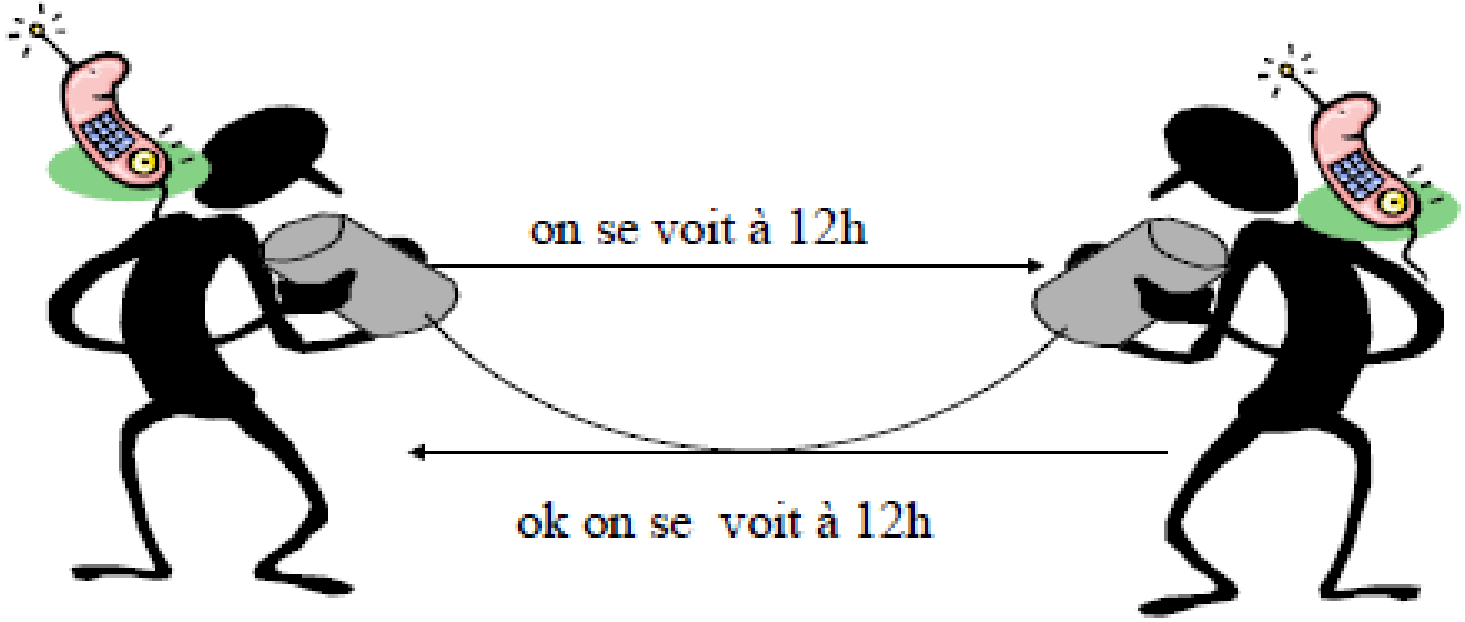
- **Systèmes distribués.** processus indépendants sur des machines distinctes et communiquant par échange de messages asynchrones (en général, des réseaux faiblement couplés).

Où sont les problèmes ?

- Y ’a t-il un algorithme pour détecter la terminaison distribuée d ’une application ?
- Comment modéliser et exprimer les algorithmes distribués ? Comment les prouver ?
- Comment les analyser, calculer leur complexité, les classifier etc…?
- Difficulté de l’algorithmique distribuée / centralisé:
  - Pas de connaissance de l’état global
  - Absence de temps universel (ou horloge globale)
  - Non déterminisme (lié souvent au problème du synchronisme)
  - Et surtout pas de modèle « universel » et standard pour l’algorithmique distribuée

Modèles de communication synchrone /asynchrone

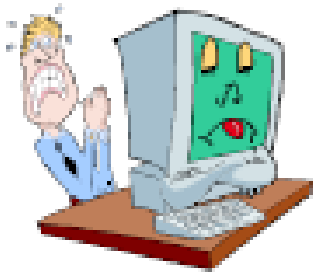
Synchrone



Même notion de temps, transmission instantanée, généralement bornée

- Communication asynchrone:

mail: on se
voit à 12h



10H

10h45

ok

ok

11h:15

est-ce qu'il sait que
j'ai lu sa réponse

ack ok

11h:45

est-ce qu'il sait
que j'ai ack ok

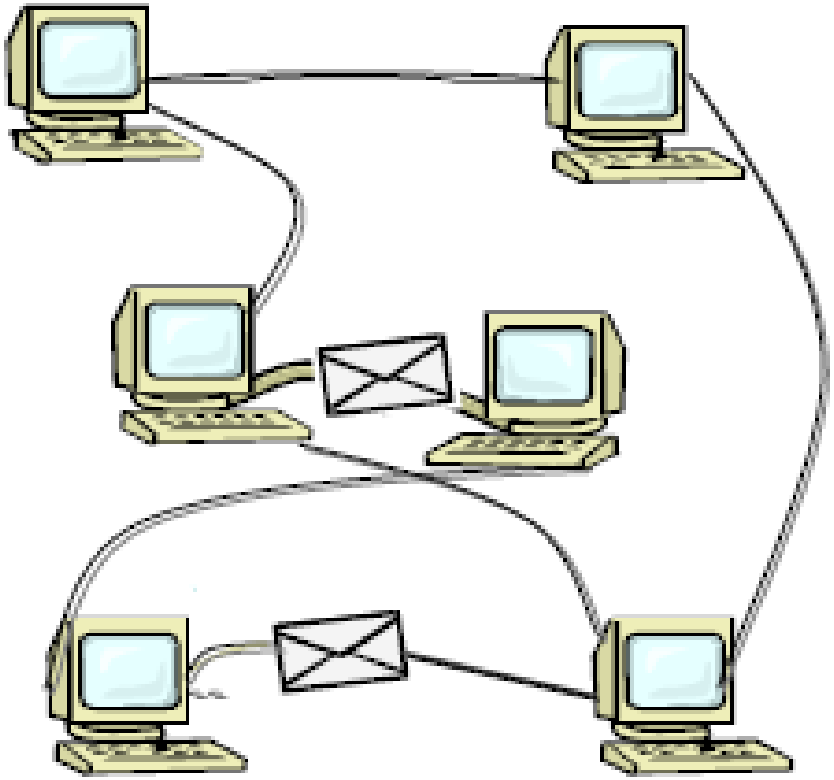
ack ack

12h15

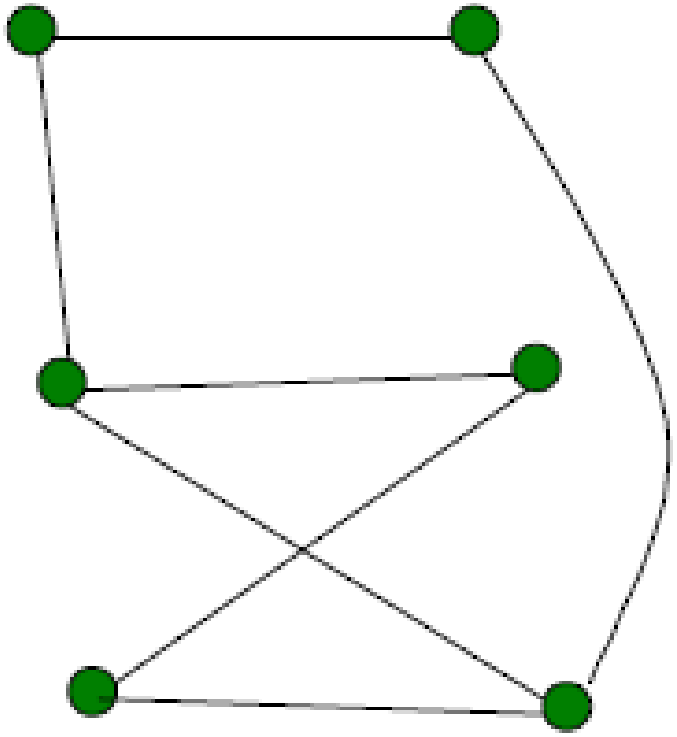
Modélisation d ’un système distribué

- Système distribuée : graphe (non orienté, connexe, simple)
  - sommet : processus
  - arête : canal de communication
- algorithme distribué local : algorithme qui s’exécute sur chaque sommet (en utilisant uniquement le contexte local)

Représentation : abstraction



un graphe

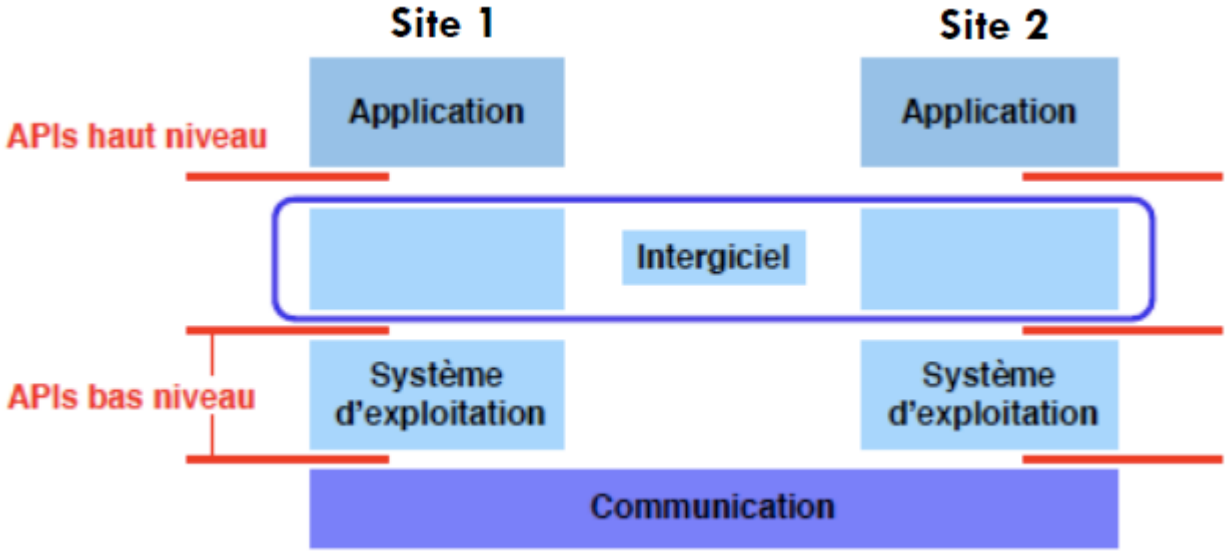


Middleware: Interlogiciel

- Dans un système réparti, 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

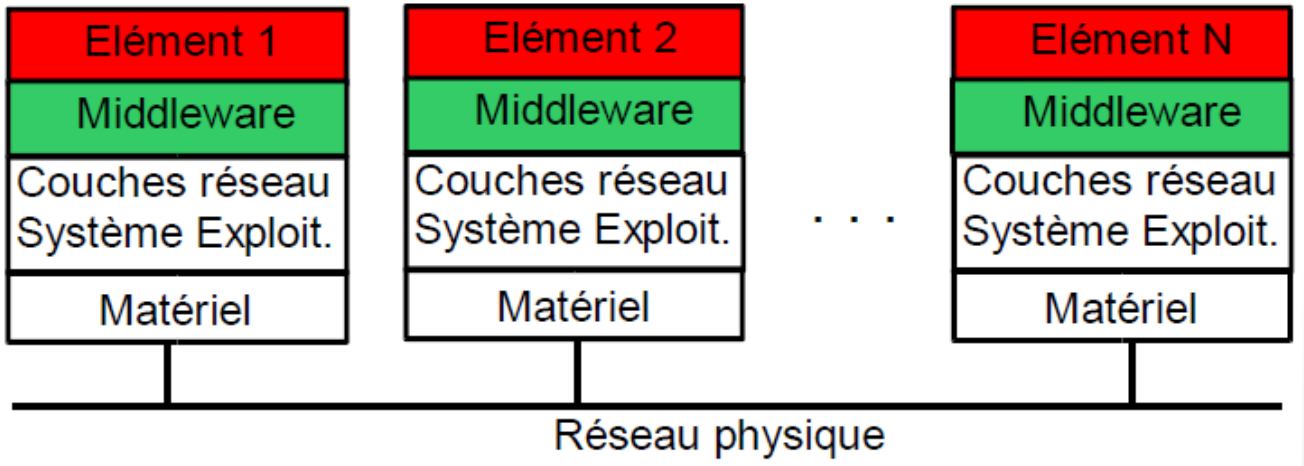
Middleware: Intergiciel

- Un **middleware** ou « **intergiciel** » ou « **élément du milieu** » est l'ensemble des couches réseau et services logiciel qui permettent le **dialogue** entre les différents composants d'une application répartie.
- *Gartner Group* définit le middleware comme une ***interface de communication universelle*** entre processus.
- Représente **l'élément le plus important** de tout système réparti.



Middleware: Intergiciel

- **Fonctions**
  - **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



Middleware: Intergiciel

- **Services du middleware**
  - **Conversion**
    - permet la communication entre machines mettant en œuvre des formats différents de données
    - prise en charge par la FAP (**Format And Protocol**)
  - **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
  - **Communication**
    - permet la transmission des données entre les deux systèmes

Middleware: Communication - Exemple

- Exemple basique de protocole
  - Une entité envoie des données à une deuxième entité
  - La deuxième entité envoie un acquittement pour prévenir qu'elle a bien reçue les données
  - Mais si utilise un réseau non fiable ou aux temps de transmission non bornés
    - Comment gérer la perte d'un paquet de données ?
    - Comment gérer la perte d'un acquittement ?
    - Comment gérer qu'un message peut arriver avant un autre alors qu'il a été émis après ?

Middleware: **Modèles d'interaction**

- Les éléments distribués interagissent, communiquent entre eux selon plusieurs modèles possibles
  - Client/serveur
  - Diffusion de messages
  - Mémoire partagée
  - Pair à pair
  - ...
- Abstraction/primitive de communication basique
  - Envoi de message d'un élément vers un autre élément
  - A partir d'envois de messages, peut construire les protocoles de communication correspondant à un modèle d'interaction

Middleware: **Modèles d'interaction**

- Rôle des messages
  - Données échangées entre les éléments
    - Demande de requête
    - Résultat d'une requête
    - Donnée de toute nature
    - ...
  - Gestion, contrôle des protocoles
    - Acquittement : message bien reçu
    - Synchronisation, coordination ...