1. Définitions

« Un système réparti est un ensemble de processeurs (sites/nœuds) interconnecté par un réseau dans lequel chaque processeur a sa propre mémoire et ses propres périphériques ».

« Un OS réparti est un OS qui apparaît aux utilisateurs comme un OS centralisé, mais qui s'exécute sur plusieurs CPU indépendantes et interconnectées ».

« Un système réparti est un système dans lequel une machine dont vous n'avez jamais entendu parler auparavant vous empêche de travailler ».

un système distribué est une collection d'ordinateurs indépendants qui apparaît à ses utilisateurs comme un seul système cohérent.

Il y a donc deux aspects là-dedans :

- hardware les différentes machines qui composent l'ensemble sont autonomes.
- software les utilisateurs ont l'impression d'intéragir avec un et un seul système.

Une première définition : "*un système réparti est un système informatique dans lequel les **ressources** ne sont pas **centralisées***". Ces ressources sont notamment :

- les moyens de stockage (données, fichiers)
- la charge CPU
- les utilisateurs
- les traitements …

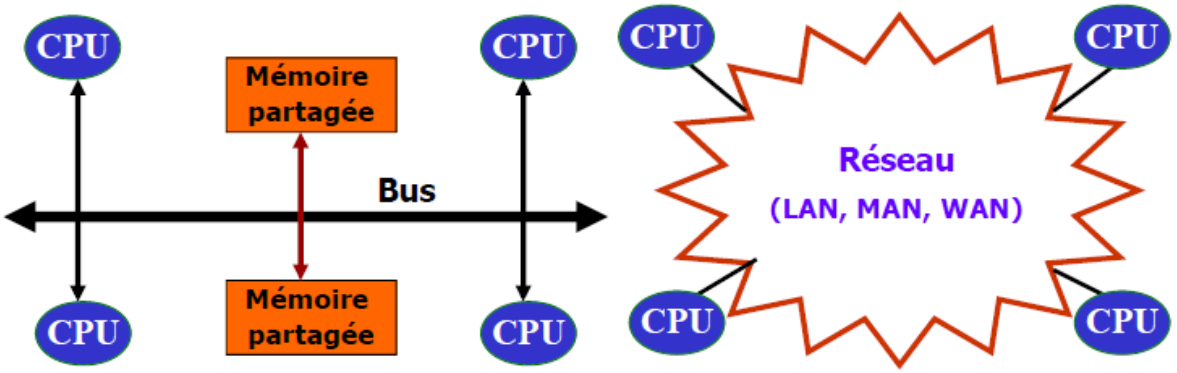
Système réparti ⟺ système distribué

La problématique de la répartition est née avec l'idée de faire communiquer des ordinateurs via un réseau de communication

les ressources (ordinateurs) sont les "nœuds" du modèle et la communication se fait par échange de message

"nœud"→ serveur → ressources → protocole de répartition _

Les middlewares = moyens logiciels de mise en œuvre des applications distribués (RMI, CORBA, …)



- **Systèmes fortement couplés (systèmes parallèles)**
- **Systèmes faiblement couplés (systèmes répartis)**

2. Objectifs de la répartition

– Partage de ressources

– Tolérance aux fautes (disponibilité, fiabilité…)

– Plus de capacités de stockage

– Plus de capacités de traitement

– Répartition géographique de l'environnement du système informatique

– Extensibilité et développement incrémental

– Flexibilité

– Réduction des coûts

Un bon système réparti est un système qui "à l'air centralisé" (paradoxe) Afin de simplifier l'effort de programmation dans une application répartie, il vaut mieux masquer le plus possible les propriétés délicates de la répartition Mais, il est impossible d'être transparent sur tous les aspects

3. Les propriétés de SD :

1. **transparence :**

a) transparence d'accès

b) transparence de localisation

c) transparence du partage

d) transparence de la réplication

e) transparence des fautes

f) transparence de la migration

g) transparence de charge

h) transparence d'échelle

A. transparence d'accès

l'interface d'accès à une ressource ne doit pas dépendre que la ressource soit locale ou distante
la syntaxe utilisée ne doit pas être différente très souvent difficile à mettre en place en fonction de la nature de la ressource
exemple: l'accès un fichier que le fichier soit local ou distant grâce à la transparence obtenue par l'utilisation de NFS
exemple : l'accès à l'interface d'un objet distribué (méthode distante)

B. transparence de localisation

la désignation de la ressource est indépendante de la localisation de cette ressource nécessite un service de nommage global (annuaire)
exemple : utilisation des "Naming services" de la norme CORBA pour les objets distribués
nécessite une connaissance de l'architecture du réseau

C. transparence du partage

les accès concurrents à la ressource sont contrôlés de telle façon que l'intégrité de la ressource est assurée
synchronisation des accès en lecture et écriture de la ressource nécessite de décrire les traitements informatiques sous la forme de "transaction" dont l'atomicité garantit l'intégrité de la données
exemple des transactions dans des bases de données réparties
exemple de l'enchaînement des méthodes synchronisées d'un objet distribué

D. transparence de la réplication

consiste à s'assurer que l'accès à une ressource soit identique quel que soit la forme d'implantation d'une ressource répliquée utilisé pour augmenter la tolérance aux fautes
l'accès à une ressource répliquée est indépendante de l'indisponibilité d'une de ses occurrences
la mise en oeuvre de la réplication est un exercice difficile. Mais facilité par l'usage de mécanismes de réplication intégrés comme ceux des SGBD la réplication permet un routage des transactions (lecture / écriture)

E. transparence des fautes

détection de la défaillance des noeuds du système réparti pallier à cette défaillance d'une manière transparente tolérance aux fautes de l'ensemble des services d'un système réparti ne pas bloquer le fonctionnement global de l'application
la réplication permet une transparence de l'indisponibilité d'une ressource et/ou d'un service

F. transparence de la migration

la propriété de migration consiste à assurer qu'une ressource peut migrer d'un noeud à un autre sans que les usagers s'en aperçoive.

déplacer un service dynamiquement vers un serveur moins chargé mettre en oeuvre des stratégies de régulation de charge sur une architecture répartie

G. transparence de charge

la régulation de la charge de chaque noeud peut permettre une meilleure exploitation du système global et une meilleure satisfaction des utilisateurs.

H. transparence d'échelle

un changement d'échelle d'une application réparti consiste à augmenter le nombre de nœud , l'ajout de noeuds ne doit pas nécessiter l'arrêt du système ce changement d'échelle n'est pas forcement transparent pour les usagers, il faut donc le contrôler et le rendre le plus transparent possible.

2. Fiabilité

– tolérance aux fautes
– évitement de fautes
– détection et recouvrement de fautes

3. **Flexibilité :** facilité d'utilisation, de maintenance, de mise à jour...
4. **Performance** (optimisation de la gestion des ressources)
5. **Hétérogénéité** (du matériel et logiciel)
6. **Sécurité**

4. Domaines d'applications

– Calcul scientifique
– Simulation distribuée
– Multimédia
– Téléconférence
– Travail coopératif
– Robotique
– Télécommunications
– Bourse
– Réalité virtuelle
– Intelligence artificielle, Multi-agents
– Jeux en réseaux