Université de Gabés
Faculté des Sciences de Gabés

Licence en génie logiciel et systèmes d'information
Niveau: 3eme année
Semestre 1

Développement d'applications réparties

Dhikra KCHAOU
dhikrafsegs@gmail.com

2023-2024

Présentation générale du cours

- **Volume Horaire**
  - 1,5h par semaine COURS
  - 1,5h par semaine TP
- **Prérequis**
  - Programmation Réseaux et Technologies Web

Objectifs

- Étudier les caractéristiques des systèmes répartis et comprendre les principes de base de ce type de systèmes.
- Apprendre les techniques de résolution des problèmes liés à la répartition.
- Apprendre les méthodes et techniques pour la programmation et l'exploitation d'applications réparties.

Plan du cours

Chapitre 1: Introduction aux systèmes répartis

Chapitre 2 : Modèles de communication dans un système réparti

Chapitre 3: Communication via les Sockets TCP/UDP

Chapitre 4: Communication via le Middleware RMI

Chapitre 5: Communication via le Middleware CORBA

Chapitre 6: Middleware orienté messages

Chapitre 7: Problèmes fondamentaux de la répartition

Chapitre 1: Introduction aux systèmes répartis

Plan du chapitre

1. Historique: Évolution des architectures de communication distantes
2. Motivations des systèmes répartis
3. Définition d’un système réparti
4. Exemples des systèmes répartis
5. Intérêts des systèmes répartis
6. Propriétés des systèmes répartis
7. Concepts liés à la répartition

Historique: Évolution des architectures de communication distantes

« *Un **système informatisé** est un ensemble de logiciels et de matériels potentiellement hétérogènes qu'il fallait interconnecter et faire coopérer, …* » (Pollet Yann,2019)

- **Système centralisé**
  - Caractéristiques des Systèmes centralisés
    - Tout est localisé sur la même machine et accessible par le programme
    - Système logiciel s'exécutant sur une seule machine
    - Les applications accèdent localement aux ressources nécessaires (données, code, périphériques, mémoire ...)

Historique: Évolution des architectures de communication distantes

- Motivations technologiques
  - **Évolution de la technologie**
    - Augmentation des performances
    - Diminution des coûts
    - Banalisation des réseaux de télécommunications
    - Performance des réseaux (Débit et fiabilité)
  - **Évolution des besoins**
    - Structure des organisations: communication et partage
    - Accès universel à l'information

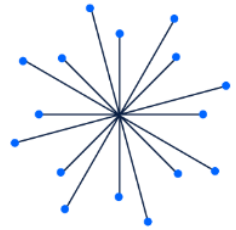
Pourquoi les systèmes répartis??

- Aspects économiques (rapport prix/performance)
- Adaptation de la structure d'un système à celle des applications (géographique ou fonctionnelle)
- Besoin d'intégration (applications existantes)
- Besoin de communication et de partage d'information
- Partage de ressources (programmes, données, services)
- Réalisation de systèmes à grande capacité d'évolution
- Tolérance aux pannes (fiabilité, disponibilité)

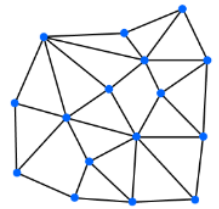
Définition d'un système réparti

- **Définition 1 [Tanenbaum, 2012]:** un système réparti est un ensemble d'ordinateurs indépendants sans mémoire partagée qui apparait à un utilisateur comme un système unique et cohérent:
  - Les machines sont autonomes
  - Les utilisateurs ont l'impression d'utiliser un seul système (la transparence)
- **Définition 2:** Ensemble d'ordinateurs indépendants connectés en réseau et communiquent via ce réseau. Cet ensemble apparaît du point de vue de l'utilisateur comme une seule entité.
- **Définition 3 [Coulouris et al., 2012]:** Un système réparti est un ensemble de machines autonomes connectées par un réseau, et équipées d'un logiciel dédié à la coordination d'activités du système ainsi qu'au partage de ses ressources.

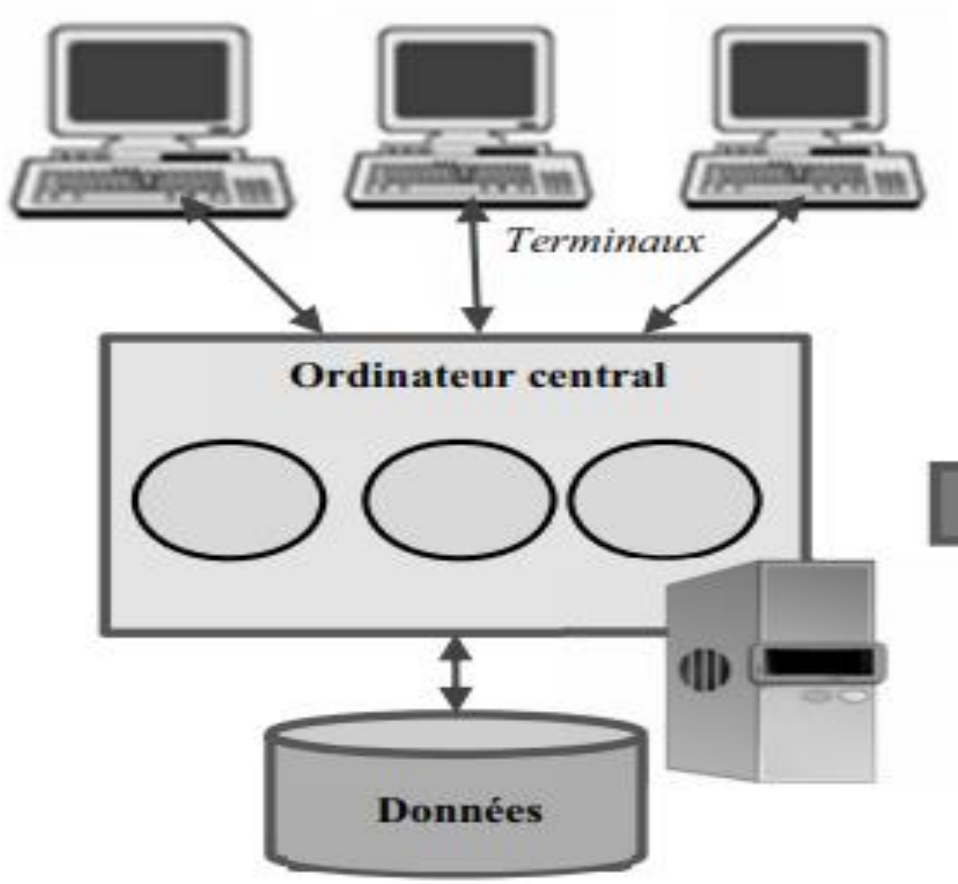
Système centralisé et système réparti



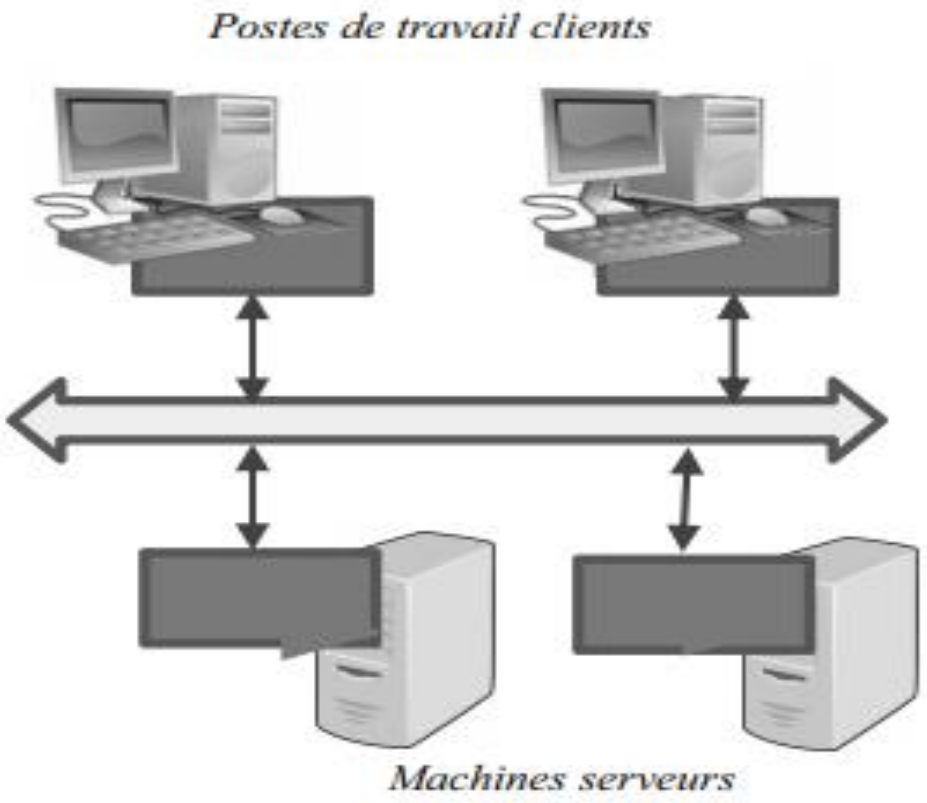
Centralized



Distributed



Système centralisé



Système réparti

Système centralisé VS système réparti

	Système centralisé	Système réparti
Architecture matérielle	Un ou plusieurs processeurs Une horloge commune pour tous les processus Une mémoire centrale commune Un espace d'adressage commun pour les processus	Plusieurs processeurs Multiprocesseurs à mémoire partagée (chaque machine) Pas d'horloges communes Plusieurs mémoires/ pas de mémoire partagée **Réseaux de communication**
SE	Centralisé État: instantané, unique et global (mémoire commune) Évènements totalement ordonnés	Distribué Absence d'état global (plusieurs horloges: événements partiellement ordonnés) Existence d'un réseau: influence le comportement du SE
Ressources	Localisées sur la même machine que les processus qui les utilisent Accessibles localement aux processus	Localisées sur plusieurs machines Les processus accèdent aux ressources locales ou distantes
QOS	Performance faible Sécurité forte Fiabilité forte	Performance forte Sécurité faible Fiabilité forte

Exemples de Systèmes répartis

- **Serveur de fichiers**
  - Accès aux fichiers de l'utilisateur quelque soit la machine utilisée
  - Physiquement : les 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.
- Intérêts
  - Accès aux fichiers à partir de n'importe quelle machine
  - Transparent pour l'utilisateur
- Inconvénients
  - Si réseau ou le serveur plante : plus d'accès aux fichiers

Exemples de Systèmes répartis

- **Serveur Web**
  - Un serveur web auquel se connecte un nombre quelconque de navigateurs web (clients)
  - Accès à distance à l'information
    - **Accès simple**: Serveur renvoie une page HTML statique qu'il stocke localement
    - **Accès dynamique**: 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 répartis

- **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 clientes
    - 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, le système s'arrête

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
- Obtenir un système plus robuste
  - Duplication pour fiabilité : deux serveurs de fichiers dupliqués

Propriétés des systèmes répartis: Hétérogénéité

- L'hétérogénéité concerne:
  - 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

Propriétés des systèmes répartis : Fiabilité

- 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 peuvent planter, engendrant une paralysie partielle ou totale du système
- **Tolérance aux fautes** : Capacité d'un système à gérer et résister à un ensemble de problèmes

Propriétés des systèmes répartis : sécurité

- Un système distribué est beaucoup plus sujet à des attaques
  - Communications 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

Propriétés des systèmes répartis : Transparence

La propriété fondamentale d'un système réparti

- La transparence concerne une fonctionnalité, un élément qui doit être invisible ou caché à l'utilisateur ou un autre élément formant le système distribué.
- Le 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 niveaux de transparences (norme RM-ODP): Accès, localisation, concurrence, réplication, mobilité, panne, performance, échelle…

Propriétés des systèmes répartis : Transparence

- **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

Propriétés des systèmes répartis : Transparence

- **Transparence de mobilité**
  - Possibilité de déplacer des éléments/ressources
- **Transparence de panne**
  - Doit supporter qu'un ou plusieurs éléments tombe en panne et ces pannes sont cachés à l'utilisateur
- **Transparence de performance**
  - Possibilité de reconfigurer le système pour en augmenter les performances
- **Transparence d'échelle**
  - Capacité d'augmenter la taille du système (nombre d'éléments, de ressources ...)

Concepts liés à la répartition

- Distinction entre “système” et “application”:
  - **Système :** gestion des ressources communes et de l’infrastructure, lié de manière étroite au matériel sous-jacent
    - Exemples: Système d’exploitation, Système de communication
    - Caractéristiques: cachent la complexité du matériel et des communications, et fournissent des services communs de plus haut niveau d’abstraction
  - **Application :** réponse à un problème spécifique, fourniture de services aux utilisateurs (qui peuvent être d’autres applications).
    - Une application utilise les services généraux fournis par le système

Distinction entre “système réparti” et “application répartie”

- **Système Réparti:** est un ensemble de systèmes calculatoires autonomes (exp: ordinateur, serveur, terminal, etc.) sans mémoire physique commune qui communiquent à travers un réseau quelconque.

- **Application répartie**: est un ensemble de processus qui tournent sur un système réparti afin de fournir ou utiliser un service déterminé

Application répartie = traitements coopérants sur des données réparties

Services et Interfaces

- **Un service** est un comportement **défini par contrat**, qui peut être implémenté et fourni par un composant pour être utilisé par un autre composant, sur la base exclusive du contrat.

  Un service est accessible via une ou plusieurs interfaces

- **Une interface** décrit l'interaction entre le demandeur et le fournisseur du service
- La fourniture d'un service met en jeu deux interfaces:
  - Interface requise (côté client)
  - Interface fournie (côté serveur)

Exemple : Agence de voyage

- Un produit «voyage» est la combinaison de plusieurs produits:
  - Gestion de réservation de billets de transport
  - Gestion de réservation des hôtels
  - Gestion de réservation de voitures de location
  - …
- Le résultat des informations récupérées auprès de différents fournisseurs
  - Compagnies aériennes
  - Chaînes hôtelières
  - Agences de location de voiture
  - …

Exemple : Agence de voyage

Agence de Voyage
(plateforme technique 1)

Application Web

Compagnies aériennes
(plateforme technique 2)

Réservations

Tarifs

Promotions

Compagnies de location de voitures
(plateforme technique 3)

Chaîne hôtelière
(plateforme technique 4)

Compagnies d'assurance
(plateforme technique 5)

Programmation classique VS programmation répartie

- La plupart des applications réparties sont de type **client/serveur**: le client demande des services à un serveur
- En **programmation « classique »** lorsque qu'un programme a besoin d'un service, il appelle une fonction d'une librairie, une méthode d'un objet, etc.
- **En programmation répartie**, l'appel de fonction / procédure /méthode peut se faire à distance
  - Proposer des méthodes / concepts / outils permettant de simplifier le développement d'application réseau client/serveur, en essayant de s'abstraire de l'aspect « distant ».

Programmation classique VS programmation répartie

Programmation classique	Programmation répartie
L'utilisateur du service et le fournisseur de service se trouvent sur la même machine: ➢ Même OS ➢ Même espace mémoire ➢ Même capacité de calcul CPU ➢ Pas de problème de transport ➢ Disponibilité du service assuré (tant que l'on a accès à la librairie)	L'utilisateur et le fournisseur de service ne se trouvent pas sur la même machine: deux machines différentes (sans compter celles traversées): ➢ OS différents ➢ Espace mémoire non unitaire ("passer un pointeur comme argument"?) ➢ Problème de transport : pare-feu (firewall),réseau, etc. ➢ Retrouver le service ? où se trouve t-il ? qui le propose?

Programmation classique VS programmation répartie

Programmation classique	Programmation répartie
Un même langage: ➢ Même paradigme de programmation ➢ Même représentation des types de base ➢ Même représentation de l'information composite	Deux langages: ➢ Représentation des types de base et de l'information composite pouvant être différente ➢ Association des paramètres effectifs aux paramètres formels ? ➢ Comment gérer les différents types de passage de paramètre ? ➢ Paradigmes de programmation différents : qu'est ce que c'est un objet pour un langage procédural ? Comment gérer les erreurs ?

Programmation “classique” vs. Informatique Répartie: Solutions

- Séparer la spécification/conception de l’implantation
  - Utilisation d’un langage propre à la spécification/conception
  - Utilisation de “traducteurs” vers le langage cible en distinguant le client du serveur
- Utiliser un langage de représentation de l’information
  - Langage de représentation indépendant du langage de programmation
  - Pour chaque langage de programmation, définir un ensemble d’opérations pour “sérialiser” ces types (prédéfinis ou utilisateur)

Programmation “classique” vs. Informatique Répartie: Solutions

- Utiliser un protocole de transport
  - Comment spécifier le service demandé
  - Comment associer les paramètres effectifs aux paramètres formels
  - Comment transmettre les erreurs
- Définir la gestion du service
  - Utiliser un mécanisme permettant d’identifier la librairie (au sens large) qui fournit le service
  - Utiliser un mécanisme qui permet d’activer le service si besoin

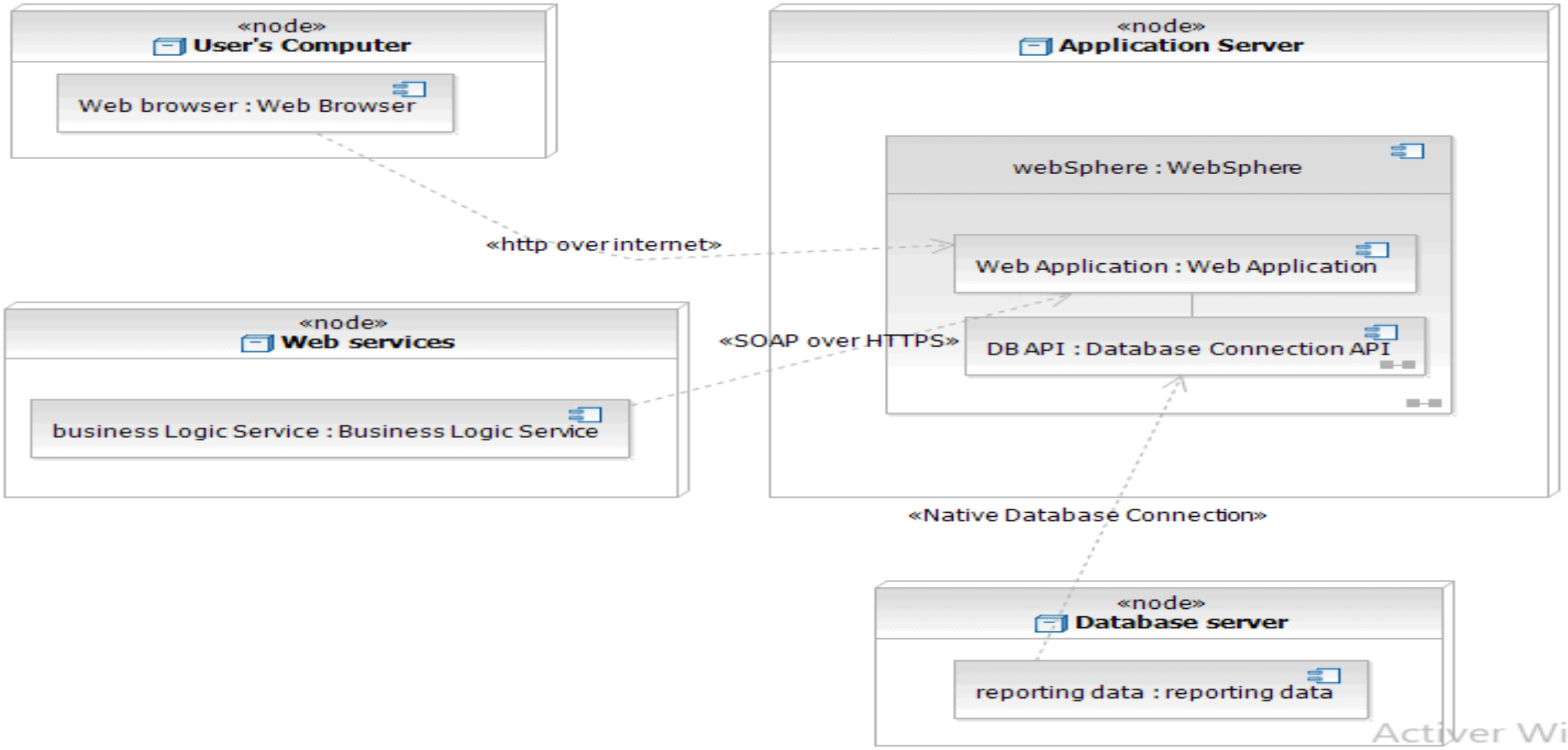
Impact sur le génie logiciel

- La programmation répartie ajoute des degrés de complexité au génie logiciel
  - Réflexion nécessaire sur la topologie de l'application et sur les interfaces entre systèmes
  - la prise en compte de connecteurs externes à identifier pour des applications ouvertes (type SOAP ou REST)
  - La prise en compte d'environnements non homogènes avec des langages différents
  - Des problématiques non fonctionnelles supplémentaires : sécurité et identification, répartition de charge, reprise de panne, persistance des données, reprise sur erreur …

Impact sur le génie logiciel

- **Modification de la conception**
  - Au-delà d'identifier les classes et leurs opérations, il faut identifier la synchronisation de tous ces objets sur le réseau
  - Diagramme de déploiement !!!
- **Modification du processus de développement**
  - Plusieurs nouvelles bibliothèques et API à utiliser
  - La compilation n'est plus la seule étape à prendre en considération
  - Debugging
    - Test de temps de réponse du middleware
    - Tests de charge
    - Tests de synchronisation
    - Tests d'accès concurrent

Exemple d’un diagramme de déploiement



Conclusion

- Caractéristiques d'un système réparti :
  - c'est un ensemble de machines indépendantes (chacune dispose d'un processeur et d'une mémoire),
  - ces machines sont connectées au réseau et communiquent via ce réseau.
  - Les utilisateurs ont l'impression d'utiliser un seul système (transparence)
- Cette communication nécessite:
  - Un modèle de communication (client/serveur, pair à pair, …)
  - Un moyen/technique de communication qui peut être un **logiciel ou autres moyens** (Appel de procédure, intergiciel, envoi de message, …)