Université d'Oum el Bouaghi
Département Math & Informatique
3 ISIL

Module : Système d'information Distribué
DATE : 21/01/2024 9 :00 – 10 :30
@.CHORFI

CONTROLE SESSION NORMALE & CORRIGE TYPE

Exo1 : (7pts)

1. **Quel est le type de communication utilisé dans les systèmes distribués? (1p)**

Dans les systèmes distribués, on utilise généralement la communication inter-processus (IPC) comme type de communication. Cette communication peut se faire à travers des mécanismes tels que les appels de procédure à distance (RPC), les messages.

2. **Quelle est la caractéristique principale qui distingue un système distribué d'un système centralisé? (1p)**

Contrairement à un système centralisé, où toutes les fonctions sont regroupées sur une seule machine ou un emplacement central, un système distribué répartit la charge de travail et les responsabilités entre plusieurs unités de traitement indépendantes interconnectées.

3. **Quel est l'avantage de la réplication dans un système distribué? (1p)**

- **Disponibilité améliorée :** En répliquant les données ou les services sur plusieurs nœuds, le système distribué peut continuer à fonctionner même si certains nœuds rencontrent des problèmes ou deviennent indisponibles. Les utilisateurs peuvent accéder aux données ou aux services à partir de n'importe quelle réplique disponible, améliorant ainsi la disponibilité globale du système.
- **Tolérance aux pannes :** En cas de défaillance d'un nœud ou d'une partie du système, les autres répliques peuvent prendre le relais, assurant ainsi une continuité de service. Cela renforce la résilience du système face aux pannes matérielles, logicielles ou réseau.
- **Performance accrue :** La réplication peut également contribuer à améliorer les performances en permettant la répartition de la charge entre les différentes répliques. Les requêtes peuvent être traitées en parallèle, ce qui réduit la congestion et améliore les temps de réponse.
- **Équilibrage de charge :** La réplication permet de distribuer la charge de travail entre les différentes répliques, évitant ainsi la surcharge d'un nœud spécifique. Cela contribue à maintenir des performances équilibrées dans l'ensemble du système.
- **Réduction des temps de latence :** En plaçant des répliques des données ou des services plus près des utilisateurs finaux, on peut réduire les temps de latence, améliorant ainsi l'expérience utilisateur.

4. **Quel est l'objectif d'un service de nommage dans un système distribué ? (1p)**

L'objectif principal d'un service de nommage dans un système distribué est de fournir un mécanisme permettant d'associer des noms compréhensibles pour les programmeurs à des entités spécifiques au services ou des ressources.

5. **Quelle est la différence entre la communication synchrone et asynchrone dans un environnement distribué ? (1p)**

- **Communication synchrone :**
  a. Dans une communication synchrone, les processus sont étroitement synchronisés et dépendent d'une horloge commune ou d'un mécanisme de synchronisation.
  b. L'émetteur envoie un message et attend la réponse immédiate du destinataire avant de poursuivre son exécution.
  c. Cette approche est souvent plus simple à concevoir et à comprendre, mais elle peut entraîner des temps d'attente si le destinataire est occupé ou si des retards se produisent dans le réseau.
- **Communication asynchrone :**
  d. Dans une communication asynchrone, les processus fonctionnent de manière indépendante, sans avoir besoin de se synchroniser strictement.
  e. L'émetteur envoie un message sans attendre immédiatement une réponse. Il peut continuer son exécution sans bloquer son fonctionnement pour la réponse du destinataire.
  f. Le destinataire peut traiter le message et répondre en fonction de sa disponibilité, ce qui permet une meilleure utilisation des ressources et une réduction des temps d'attente potentiels.

6. **Comment les bases de données distribuées fonctionnent-elles et quels sont leurs avantages par rapport aux bases de données centralisées ? (1p)**

Les bases de données distribuées sont conçues pour stocker et gérer des données sur plusieurs nœuds ou serveurs interconnectés au sein d'un réseau. Voici comment elles fonctionnent

- **Répartition des données :** Les données sont réparties sur plusieurs nœuds du réseau plutôt que d'être centralisées sur un seul serveur.
- **Répartition des requêtes :** Les requêtes peuvent être distribuées entre les différents nœuds pour permettre un traitement parallèle et améliorer les performances.
- **Coordination :** Un système de coordination est nécessaire pour gérer la cohérence des données entre les différents nœuds. Cela peut être réalisé à l'aide de protocoles de consensus ou d'autres mécanismes de gestion de la cohérence.

- **Réplication :** La réplication de données sur plusieurs nœuds contribue à la tolérance aux pannes et à l'amélioration de la disponibilité. Cependant, la gestion de la cohérence entre les répliques est un défi.

Avantages par rapport aux bases de données centralisées : **(1p)**

- **Évolutivité horizontale :** Les bases de données distribuées peuvent être facilement étendues en ajoutant de nouveaux nœuds, ce qui permet une évolutivité horizontale. Cela est particulièrement utile pour gérer de grandes quantités de données ou une charge de requêtes élevée.
- **Tolérance aux pannes :** En distribuant les données sur plusieurs nœuds, les bases de données distribuées sont plus résilientes aux pannes. Si un nœud échoue, les autres peuvent continuer à fonctionner normalement.
- **Performances améliorées :** La répartition des requêtes et le traitement parallèle contribuent à des performances globales améliorées, surtout dans le cas de charges de travail massivement parallèles.
- **Localisation des données :** En distribuant les données géographiquement, les bases de données distribuées peuvent réduire les temps de latence en plaçant les données plus près des utilisateurs ou des applications.
- **Économies de coûts :** La distribution des données sur plusieurs serveurs permet une meilleure utilisation des ressources et peut être plus rentable que l'achat et la maintenance d'un serveur central puissant.

Exo2 : **(6pts)**

Imaginez que vous êtes un développeur chargé de concevoir une application collaborative permettant à une équipe de travailler sur un même projet de code source. **Q_{01}** : Proposerez un diagramme de cas d'utilisation incluant les principales fonctionnalités de cette application. **(4p)**

---

1. Gestion des Utilisateurs :

- Créez une fonctionnalité d'inscription et de connexion pour les utilisateurs.

- Mettez en place un système de rôles (administrateur, développeur, observateur) pour différencier les autorisations.

2. Création de Projets :

- Permettez aux utilisateurs de créer un nouveau projet de code source.

- Intégrez la possibilité d'ajouter une description, des objectifs et des langages de programmation associés au projet.

3. Édition Collaborative :

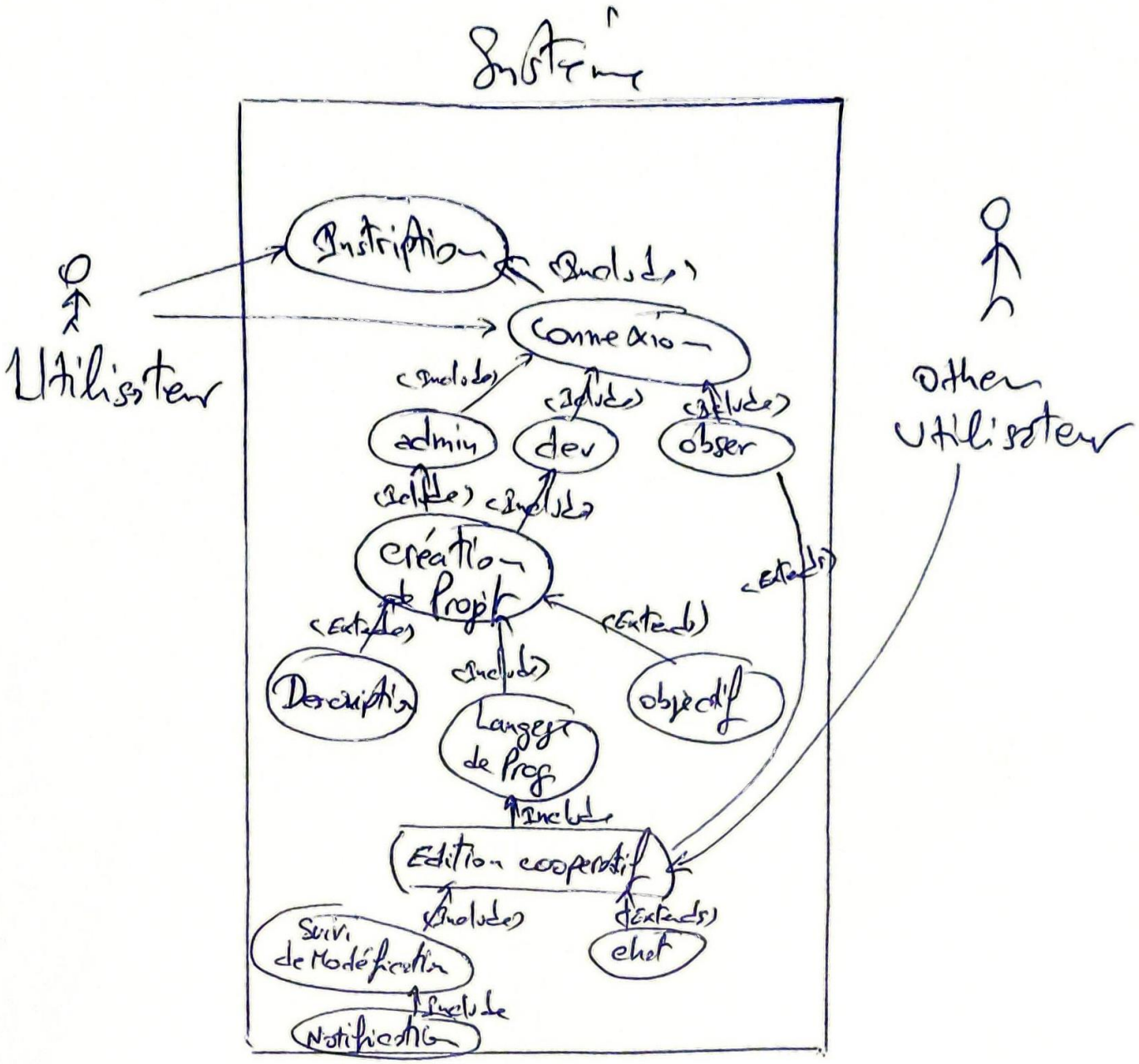
- Implémentez un éditeur de code collaboratif en temps réel permettant à plusieurs développeurs de travailler simultanément sur le même fichier.

- Ajoutez un système de suivi des modifications pour identifier les contributions individuelles.

5. Communication en Temps Réel :

- Ajoutez un chat en temps réel pour que les membres de l'équipe puissent discuter pendant le développement.

- Mettez en place des notifications pour informer les utilisateurs des modifications, des commentaires, etc.



Q02 : Comment les développeurs peuvent-ils collaborer sur le code source dans cette application ? (1p)

R2 : Une fois que les utilisateurs se sont inscrits, ils doivent se connecter et sélectionner leur rôle. Seuls les administrateurs et les développeurs peuvent créer des projets, puis choisir le langage de programmation. À ce stade, un espace d'édition de code source collaboratif sera activé.

Q03 : Comment les développeurs peuvent-ils partager des commentaires et des suggestions sur le code source ? (1p)

R3 : Au sein de l'espace d'édition collaborative, les commentaires seront automatiquement partagés avec l'ensemble du groupe. En utilisant le suivi des modifications du code source, nous avons la possibilité d'intégrer un mécanisme de suggestion et de recueillir des idées.

Exo3 : (7pts)

Soit le code source suivant :

```
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.HashMap;
import java.util.Map;
```

```
// Interface pour définir les opérations RMI
public interface CodeShareService extends Remote {                    <---------- 1 (1p)
   void shareCode(String userId, String code) throws RemoteException;
   String getCode(String userId) throws RemoteException;              <---------- 2 (1p)
}

// Implémentation de l'interface RMI
public class CodeShareServiceImpl extends UnicastRemoteObject  implements CodeShareService {  < --- 6  (1p)
   private Map<String, String> userCodeMap = new HashMap<>();

   @Override
   public void shareCode(String userId, String code) throws RemoteException {
      userCodeMap.put(userId, code);
      System.out.println("Code partagé par l'utilisateur " + userId);
   }

   @Override
   public String getCode(String userId) throws RemoteException {
      return userCodeMap.get(userId);
   }

   public static void main(String[] args) {                                <---------- 3 (1p)
      try {
         // Création du service RMI
         CodeShareService codeShareService = new CodeShareServiceImpl();  <---------- 4 (1p)

         // Enregistrement du service RMI sur le registre
         java.rmi.registry.LocateRegistry.createRegistry(1099);
         java.rmi.Naming.rebind("CodeShareService", codeShareService);

         System.out.println("Service RMI prêt à partager le code.");
      }
         e.printStackTrace();  }                                           <---------- 5 (1p)
   }
}

// Client RMI
public class CodeShareClient {
   public static void main(String[] args) {
      try {
         // Recherche du service RMI
         CodeShareService codeShareService = (CodeShareService)
java.rmi.Naming.lookup("rmi://localhost/CodeShareService");

         // Utilisateur partage le code
         codeShareService.shareCode("User1", "public class HelloWorld { ... }");

         // Utilisateur récupère le code partagé par un autre utilisateur
         String sharedCode = codeShareService.getCode("User1");
         System.out.println("Code récupéré : " + sharedCode);
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Q_{01} : à quoi ressemble ce code source ? (1p)

Ce code semble être similaire à une application de messagerie instantanée.
Q_{02} : Identifiez et corrigez cinq erreurs présentes dans ce code source. (5p)
Q_{03} : Proposez le diagramme de classe qui correspond à ce code source. (1p)