Système de Discussion Distribué

Préliminaires

Certes nous utilisons Teams pour dialoguer entre nous, mais le dialogue oral ne laisse pas de trace et dans le chat de discussion Teams mélange toutes les conversations. Donc nous pourrions disposer d'une application simple et focalisée sur les conversations textuelle de type Desktop. Cette application se doit d'être facile à utiliser pour faciliter les échanges entre les personnes et uniquement en mode texte pour laisser de la souplesse dans le temps de réponse et offrir un outil simple.

Pour concevoir ce système, nous faisons appel à votre entreprise informatique, EnStar.com

Principe général

Chaque personne possède un ordinateur personnel qui lors du démarrage exécute l'application EnstarDesktop. Chaque personne peut donc échanger avec ses collèges identifiés dans l'application. Les personnes sont identifiées dans cette application.

Au minimum, chaque personne peut et veut :

- s'identifier au démarrage de l'application (login, password)
- engager un échange en direct avec un autre collègue
- déclarer un échange pour plusieurs collègues
- effectuer un échange à plusieurs collègues
- rechercher un collègue et sa présence
- etc…

Au niveau de l'entreprise EnStar, l'administrateur de EnstarDesktop veut :

- s'identifier au démarrage de l'application (login, password)
- déclarer les utilisateurs de LampaulSkype à partir d'une ressource externe à l'application, type fichier ou base de données.
- Avoir les utilisateurs en cours d'échange
- Etc…

Bien sûr vous pouvez étendre la liste de base des fonctionnalités sur ce logiciel EnstarDesktop en intégrant par exemple :

- déclarer un sujet d'intérêt pour créer un échange sur une durée plus longue
- s'abonner à un sujet d'intérêt et échanger sur ce sujet
- Avoir la liste des sujets d'intérêt déclarés.
- Garder un historique des échanges (exemple, un fichier par échange)
    - Une extension possible : pour préserver la confidentialité de l'historique, vous pouvez le chiffrer

Méthode pour la réalisation du projet

Objectifs :

- Développer une application Java en intégrant des motifs de conception
- Appliquer une méthodologie Agile par équipe de 4 étudiants en intégrant des tests automatisés

Compte tenu des objectifs fixés, le code source du projet qui sera rendu à la fin devra IMPERATIVEMENT intégrer les motifs de conception suivants :

- Le serveur TCP concurrent
- Le sujet-observateur de l'API Java ou votre propre implantation du pattern
- Le motif délégation
- Le motif stratégie
- Le composant-composite
- Tout autre motif utilisé dont vous auriez besoin.

Les interfaces graphiques seront réalisées en JavaFX.

Pour satisfaire l'objectif lié à l'application de la méthode agile, il faudra :

- Créer des groupes de 4. 2 étudiants se chargeront de la partie serveur du système de surveillance et 2 autres étudiants des deux clients nécessaires pour la partie utilisateur.
- A la première séance, il faudra identifier les user stories qui seront couvertes par l'application.
    - Commencer avec un nombre restreint de User Stories et étendre le nombre ensuite
- Tout le long du projet vous devrez tenir à jour un diagramme de classe UML et votre JavaDoc.
- **Méthode de développement à utiliser :**
    - **1 sprint = 1 séance + 1 semaine de travail**
    - **La séance de TD sera surement trop courte mais le modèle agile reste valide**
    - **Sélection d'un scénario(s) opérationnel(s) = User Story**
    - **Définition de la structure des classes (CRC et design pattern si besoin)**
    - **Définition des tests associés avec JUNIT**
    - **Exécution pour la validation du besoin et application des scénarios de tests**
    - **Refactoring du code en appliquant SOLID et design pattern**
    - **Exécution et validation des tests**
- Chaque semaine, vous devrez déposer sur Moodle:
    - Une ou plusieurs ***user-stories*** sélectionnées pour définir le sprint dans un court fichier texte (ce fichier contiendra une courte revue du sprint effectué)
    - Une photo des **cartes CRC** effectuées dans le sprint (si besoin)
    - Les **tests** associés à ce sprint (dont le fichier XML exporté par JUnit qui contient la trace des tests effectués)
    - Une implantation **exécutable** de l'application relative au sprint
    - Votre diagramme de classe et votre documentation actuelle