CPE Lyon - 3ICS - Année 2020/21
Développement informatique – 2
Projet

1 Contexte

l'ICS (institue of Cognitives Sciences) est un laboratoire interdisciplinaire qui intègre l'expertise de chercheurs des Sciences de la Vie (psychologie cognitive, neurosciences) et de médecine (pédo-psychiatrie, neuro-pediatrie) avec celle de chercheurs des Sciences Humaines et Sociales (linguistique computationelle et théorique et philosophie) pour étudier la nature et la spécificité de l'esprit humain.

L'un des doctorants travaille sur la théorie des jeux, à la frontière de la psychologie et des mathématiques, et notamment sur les jeux a somme non constante. Une des illustrations la plus connue de ces jeux est le dilemme du prisonnier :

https://fr.wikipedia.org/wiki/Dilemme_du_prisonnier

Ces deux vidéos devraient vous éclairer sur la théorie des jeux :

https://www.youtube.com/watch?v=StRqGx9ri2I, notamment à partir de 7'15''.

https://www.youtube.com/watch?v=uijJJ2OczNs&t=913s

Le doctorant, qui n'est pas un développeur, a besoin d'accumuler des données expérimentales. Il a besoin que des volontaires jouent l'un contre l'autre un nombre de fois à définir, sans jamais savoir qui sont leurs adversaires.
On définira une partie comme étant un certain nombre de rounds. Un round est défini comme une confrontation trahison-collaboration entre les deux volontaires.

Le doctorant a travaillé avec l'ingénieur d'études du laboratoire pour définir un cahier des charges. L'ingénieur d'études à un planning très chargé, mais il a amorcé les projets (le code du serveur et le code du client). Il a aussi prévu quelques heures pour vous exposer l'état des projets et le principe de fonctionnement (gestion des sockets, gestion des connexions, communication asynchrone, multitâche). Il vous présentera aussi les outils nécessaires pour développer une interface graphique avec GTK pour l'application cliente.

Vous (le chef de projet) avez été contacté pour mettre en place une équipe (2 dev + 2 réseaux) afin de développer les deux applications le plus rapidement possible.

2 Fonctionnalités du système

Le système informatique doit être paramétrable par un non-informaticien et doit permettre d'obtenir et conserver les résultats des multiples expérimentations.
Vous devez tenir compte du fait qu'il peut y avoir plusieurs parties simultanément (deux actuellement, étant tenu compte du matériel dont dispose le laboratoire) et que prochainement le laboratoire disposera encore de quelques ordinateurs supplémentaires.

2.1 Serveur :

Le serveur doit être paramétrable. Le fichier de paramétrage permettra de définir l'adresse IP et le port sur lequel l'application serveur sera à l'écoute.
Le fichier contiendra aussi le paramétrage nécessaire pour définir les réglés du jeu : somme engagée a chaque partie, le nombre de parties et d'autres informations qui vous sembleront nécessaires.

A la fin de la partie, le serveur doit garder dans un fichier (le format est à définir) les choix de chacun des joueurs, ainsi que le temps de décision et les montants accumulés ou perdus.

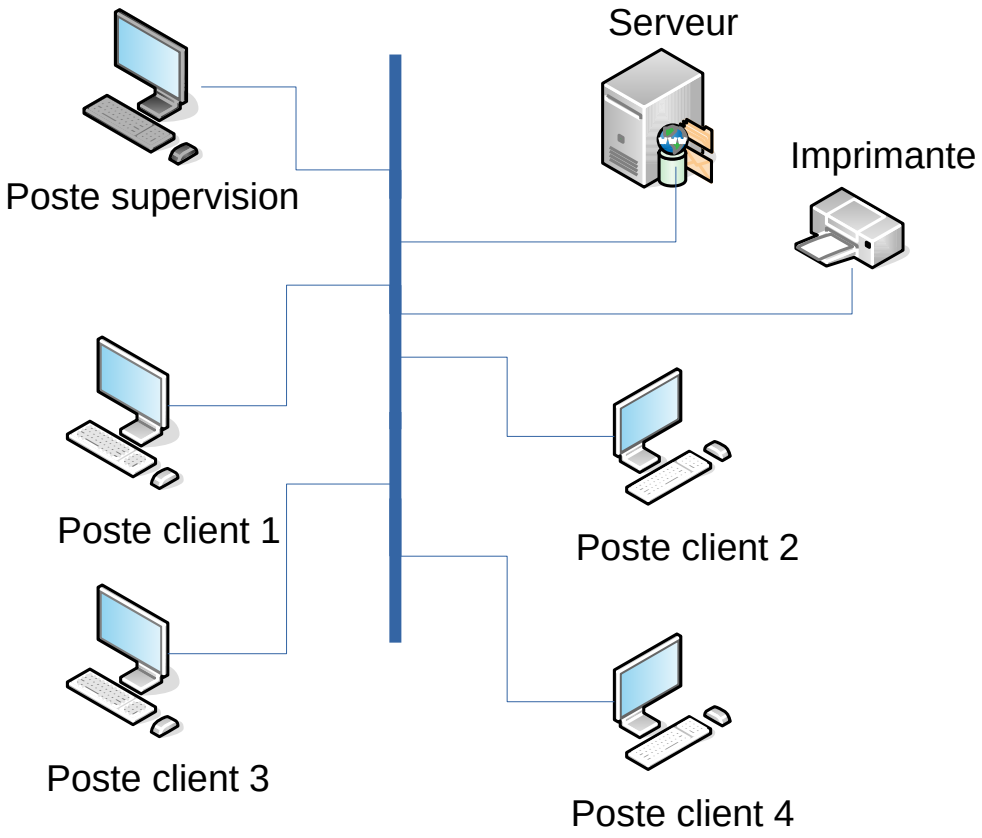
2.2 Client

Le fichier de paramétrage du client doit contenir l'adresse IP du serveur ainsi que le port et tout autre paramétrage qui vous sembleront pertinents.

L'interface graphique des clients doit rester très simple : le cobaye ne doit pas être déconcentré par des fioritures : deux boutons cliquables (Trahir, Collaborer), le résultat du round (il doit savoir s'il a gagné ou perdu). Il doit être informé du début de chaque round.

3 Architecture du système informatisé

Le laboratoire dispose de plusieurs ordinateurs connectés entre eux, chacun dans un box afin que les volontaires ne se rencontrent pas, d'un ordinateur pour la supervision, d'un serveur et d'une imprimante en réseau.



3.1 Gestion des données

Le serveur central a la charge de l'ensemble des données. Les données manipulées par le serveur sont de nature persistante, c'est-à-dire qu'elles doivent survivre à l'exécution de l'application

3.2 Gestion des échanges client/serveur

Les échanges entre les clients et le serveur doivent suivre un protocole bien défini pour que le

serveur comprenne les requêtes des clients et pour que les clients comprennent les résultats renvoyés par le serveur. Les échanges doivent être considérés au niveau applicatif et au niveau transport.
Au niveau applicatif, vous devrez définir la structure des messages qui sont échangés. Vous êtes invité à voir la définition d'un RFC. Le RFC pour le protocole HTTP est l'un des plus simple a appréhender (https://tools.ietf.org/html/rfc2616).

Au niveau transport, il est imposé d'utiliser le protocole TCP/IP.

4 Consignes

4.1 Travail de base demandé

Vous disposez d'un squelette de programme pour l'application serveur et un autre pour l'application client. Après lecture des différents documents, il faudra le compléter pour fournir :

Aspect réseau : un client et un serveur en mode TCP (le squelette de base est opérationnel)
Aspect système :

- un serveur parallèle. Le code qui vous est fourni utilise des threads pour pouvoir gérer des connexions en parallèle et faire du multitâche pour le traitement des requêtes.
- un client asynchrone capable de traiter les messages du serveur dès leurs réceptions.

Dans tous les cas, avant de commencer à coder ces questions, faites une analyse des problèmes que vous voulez résoudre et des solutions que vous allez proposer. Si cette analyse est bien faite, le codage sera facile. Une bonne analyse (même sans implémentation) dans le rapport et la soutenance sera fortement valorisée.

Nous vous conseillons de :

– bien respecter le cahier des charges ;
– bien gérer le temps qui vous est imparti ;
– bien discuter dans le quadrinome;
– réfléchir avant de programmer.
– structurer votre code pour le rendre le plus évolutif possible.
– utiliser des assertions
– bien formalisez le protocole (faite des simulations entre vous).
– penser aux évolutions possibles du projet : par exemple l'état du serveur et des clients pourra être supervisé depuis un poste de… supervision.

Tout le code doit bien sur être versionné. Comme toujours vous donnerez un accès à votre dépôt a votre enseignant.
Vous serez très attentif aux commentaires et annotations. Il est impératif de fournir une documentation (Javadoc ou Doxygen).
La présence d'assertion dans le code est un impératif, cependant les tests unitaires ne sont pas nécessaires.

4.2 Support

Vous serez encadré pendant tout le projet. Souvenez-vous cependant que votre enseignant n'est pas un 'degugger'. Vous pouvez le contacter par mail, mais l'enseignant n'est pas 24h/24 devant son ordinateur : prévoyez donc un certain délai dans les réponses. Plus la question est précise (nécessitant donc une réponse courte), plus la réponse sera rapide.

5 Évaluation du travail

5.1 Soutenance

Chaque groupe aura 20 minutes, diapositives à l'appui, pour présenter son travail. Le groupe expliquera brièvement la répartition du travail dans le quadrinôme, l'avancement du projet et les problèmes rencontrés. Il devra aussi expliquer et justifier les choix effectués pour les implémentations. Le jury posera ensuite des questions et demandera éventuellement des démonstrations.

5.2 Rapport

Le quadrinome doit rendre avant la soutenance un rapport de 4 pages (+2 pages d'annexes) maximum donnant :
– l'organisation du travail dans le groupe ;
– la méthodologie utilisée dans le développement ;
– la justification de certains choix et l'évaluation de la pertinence à posteriori;
– l'état courant du projet et ce qu'il reste à réaliser ;
– les difficultés rencontrées ;
– un rapide bilan de ce que vous a apporté ce projet.

Le rapport ne doit pas redonner des informations présentes dans ce document. Il doit contenir suffisamment d'élément pour permettre au doctorant de prendre en main l'application. Le README.md contiendra les éléments nécessaires pour utiliser les programmes, avec les formats des fichiers de paramétrages, les formats des fichiers de données, les commandes nécessaires pour compiler le code source.
Toutes les informations nécessaires à la poursuite du développement devront être stockées dans des fichiers de documentation (répertoire doc) présent dans le projet (description du protocole par exemple)