UFR ST Besançon 2011-2012
Master 1 Informatique

Systèmes Client/Serveur
Projet - Jeu Foursight en réseau

---

1 Description du projet

Le but de ce projet est l'implantation de la communication permettant de jouer en réseau au jeu Foursight. Les règles du jeu sont données dans l'énoncé du projet en MOIA. La suite définit les spécifications du travail à réaliser pour la partie communication. Le travail est à remettre en binôme.

Chaque joueur est représenté par un processus écrit en langage C. L'intelligence du jeu Foursight est développée en Prolog et est encapsulée dans un programme Java, appelé moteur IA, qui communique, à travers des sockets, avec le processus joueur. Les parties sont administrées par un serveur arbitre.

1.1 Le joueur

Voici le comportement d'un processus joueur :

1. Il envoie à l'arbitre une requête **IDENTIFICATION** pour s'inscrire.
2. Lorsqu'il est prêt, il envoie une requête **PARTIE** à l'arbitre pour lui demander de jouer. En réponse, l'arbitre lui donne l'identificateur du joueur avec lequel il va jouer et lui indique s'il doit commencer.
3. Lorsque c'est à son tour de jouer, le joueur consulte son moteur IA pour connaître le coup à jouer. Il constitue une requête **COUP** qu'il envoie à l'arbitre, puis il en attend la validation.
4. Il attend le coup de l'adversaire transmis par l'arbitre.
5. Il informe son moteur IA du coup joué par l'adversaire.
6. Si la partie est finie, le joueur retourne en 2, sinon il retourne en 3.

Le joueur peut donc jouer plusieurs parties, réunies dans un tournoi. L'arbitre l'informe de la fin de ses parties (voir le type `PARTIE` ci-dessous).

1.2 Protocole de l'arbitre

Toutes les communications se feront en mode connecté. Vous trouverez sous moodle dans le module SCS, dans la rubrique Projet, le fichier `protocolArbitre.h` qui définit le protocole d'accès à l'arbitre.

Voici les règles retenues pour l'utilisation de ces types :

IDENTIFICATION : le nom à donner est le login LDAP du joueur. Cette requête permet à un joueur de se faire enregistrer dans le tournoi. Il obtient en retour son identifiant dans le tournoi.

PARTIE : l'identificateur est celui du joueur dans le tournoi. Cette requête permet au joueur de demander une nouvelle partie. Si la valeur de `premier` est `VRAI`, il commence à jouer. Sinon il attend le premier coup. S'il a déjà joué toutes ses parties, le booléen `finTournoi` est positionné à `VRAI` et les champs `adversaire` et `premier` ne sont pas renseignés. En fin de tournoi, il doit se déconnecter et finir son exécution ainsi que celle du moteur IA.

COUP : une requête `TypCoupReq` est envoyée à l'arbitre pour jouer un coup. Cette requête donne le type de coup et la position à laquelle la pièce jouée est placée suite à sa dépose. La valeur associée à `propCoup` donne donc le type de coup qui est joué, c'est-à-dire s'il s'agit d'un ajout de pièce (`POSE`), **ou un passe (`PASSE`)**, ou si le coup termine la partie en gagnant (`GAGNE`), en perdant (`PERD`) ou par une égalité (`NULLE`). Lorsque le joueur pose un pion, il doit initialiser la position de la case de la pose (`caseArrivee`) avec les valeurs de type `TypLigne` pour l'indice de la ligne et `TypCol` pour l'indice de la colonne. **Si le coup est un PASSE, il doit initialiser le type de pièce posée à `VIDE`.**

En réponse à la requête, l'arbitre retourne une validation du coup `TypCoupRep` avec une valeur d'erreur initialisée à ERR_OK. Le joueur attend ensuite de recevoir un prochain message de la part de l'arbitre. Il reçoit d'abord un message `TypCoupRep` qui lui dit si le coup joué par l'adversaire est valide, s'il y a triche ou s'il est en `TIMEOUT`. Ensuite, il reçoit le message `TypCoupReq` qui a été envoyé à l'arbitre par l'adversaire. Dès qu'il a reçu le deuxième message, il peut s'adresser à son moteur IA pour lui demander le prochain coup à jouer.

Remarque. ~~Si le coup proposé est `GAGNE`, `PERD` ou `NULLE`~~, ~~le champ `typePiece` est initialisé à `VIDE` et le champ `caseArrivee` à la position (`LI_ZERO`, `CO_ZERO`).~~

1.3 Validation des coups

Quelques précautions sont prises pour éviter les erreurs répétées, voire moins bien intentionnées.

- Si un coup est erroné ou que le joueur annonce qu'il a gagné alors que c'est faux, **ou qu'il annonce un coup passe et qu'il détient encore des pièces blanches**, la partie est perdue. L'arbitre retourne une valeur `TRICHE` dans le champ `validCoup` et en informe simultanément le joueur adverse. Chacun des joueurs doit alors demander une nouvelle partie, le coup invalidé n'est pas transmis à l'adversaire.
- La recherche dans le moteur IA est limitée dans le temps. Le temps d'attente est calculé à partir de l'instant où l'arbitre envoie le coup adverse. Si la limite est dépassée, le coup est considéré comme nul et l'arbitre envoie au joueur et à l'adversaire un message `TypCoupRep` avec une valeur `TIMEOUT` dans le champ `validCoup`. Dans ce cas, la partie est perdue par le joueur. Attention, un joueur peut donc recevoir un `TIMEOUT` alors qu'il est en train de jouer.

Vous devez développer le protocole de communication entre le joueur et son moteur IA. Les messages seront échangés sur des sockets avec un accès en langage Java du côté du moteur IA et un accès en langage C du côté du joueur.

Le joueur prend en paramètre le nom de la machine et le numéro de port du processus arbitre pour s'y connecter.

Remarque sur la programmation Pour que vos programmes soient facilement lisibles, vous respecterez le standard de programmation C.

2 Rapport

Pour expliquer votre réalisation, vous nous remettrez un rapport (5-6 pages) qui contiendra obligatoirement les informations suivantes :

- utilisation du protocole de communication avec l'arbitre,
- la mise en place d'un protocole avec le moteur IA en Java,
- explications sur le code, la structure de votre joueur.

La longueur de ce rapport n'est pas un facteur déterminant, portez plus d'attention à sa clarté et à la structuration de vos explications. Ce rapport est à remettre en même temps avec les sources du projet, au format pdf.

3 Championnat

Afin d'organiser le championnat (11 mai), vous nous fournirez votre joueur en le déposant comme devoir dans le module SCS sous moodle, pour le 9 mai à midi, au plus tard. Le fichier, portant les deux noms du binôme, sera une archive .tar.gz qui contiendra un répertoire, avec le même nom. Dans ce répertoire, mettre tous les fichiers de votre projet (sauf les fichiers temporaires .o, `~`, etc.) et un exécutable appelé `joueur`. Cet exécutable (qui peut être un script de lancement) permet de lancer à la fois votre joueur et le moteur IA. ATTENTION, tout manquement à ces règles peut vous exclure du championnat mais, si vous vous y prenez un peu à l'avance, il sera possible de nous demander une validation (lors de la dernière séance de TP).