PCD Assignment #03

Esercizio #1 - *Collective mistermind*

Obiettivo del progetto è la realizzazione di un programma ad attori ispirato ad una rivisitazione del gioco *Mastermind* (https://it.wikipedia.org/wiki/Mastermind).

Il programma consiste in N attori (dove N è un parametro) che fungono da giocatori (*player)* del gioco e un attore che funge da arbitro (*arbiter*). Ogni player P_i (i in [0..N-1]) ad inizio gioco sceglie a caso un numero di M cifre (M è un parametro del programma). Questo numero costituisce il suo numero segreto, che gli altri player devono scoprire. Obiettivo di ogni player è scoprire il numero secreto di tutti gli altri giocatori. Vince il player che indovina per primo il numero segreto di tutti gli altri.

Il dinamica del gioco procede per turni, coordinati dall'arbitro. A ogni turno, ogni player può eseguire un tentativo (*guess*) per indovinare il numero di un altro specifico player. Il criterio con cui il player ad ogni turno sceglie l'attore a cui fare il guess e la scelta del guess stesso non sono specificati. L'attore che riceve il guess invia una risposta, codificata in 2 numeri: numero delle cifre indovinate al posto giusto, numero di cifre indovinate ma non al posto giusto. Tale risposta viene inviata non solo all'attore che ha fatto il guess, ma a tutti i player. A ogni turno, i player eseguono il proprio tentativo secondo un ordine stabilito dall'arbitro (ordine casuale). Ad esempio, se l'ordine è 1 2 0, allora per quel turno potrà fare il proprio tentativo prima P_1, poi P_2 e infine P_0. L'ordine cambia a ogni turno. Un player ha un tempo limitato per eseguire il proprio tentativo *(guess timeout)*, misurato dall'arbitro, scaduto il quale il player salta il turno (e il suo guess non viene considerato).

Quando un player - dopo aver eseguito un tentativo - ritiene di aver scoperto tutti i numeri, allora può inviare una dichiarazione di vittoria all'arbitro, con l'elenco dei numeri. L'arbitro procederà a verificare, chiedendo a tutti i player se la risposta è corretta, relativamente al proprio numero. In tal caso, la partita termina e viene dichiarato dall'arbitro il vincitore. In caso contrario, il giocatore viene escluso.

Il programma deve essere dotato di una GUI con cui:

- Scegliere il numero N di attori e il numero M di cifre.
- Visualizzare informazioni sullo stato del gioco (a scelta dello studente).

- Far partire una nuova partita e interrompere (facendola terminare) una partita in corso.

FACOLTATIVO

- Includere fra i giocatori anche un giocatore umano, che possa partecipare mediante un'apposita GUI
- E' possibile considerare una variazione del gioco in cui la risposta ad un guess da parte di un player è inviata al solo player che ha fatto il guess

Si chiede di progettare e sviluppare il programma usando un approccio ad attori, nel concentrato (non distribuito).

Esercizio #2 - *Distributed Collaborative Puzzle* [TBC]

Fra i sorgenti del corso è fornito un programma che implementa un puzzle semplificato per bimbi, pensato per un solo giocatore. Scelta un'immagine di dimensioni WxH, il puzzle è rappresentato da una matrice NxM di tessere, dove le tessere hanno dimensione W/M e H/N. L'immagine all'inizio del gioco ha le tessere disposte casualmente, non nel posto giusto. Il giocatore mediante una semplice GUI può scambiare la posizione di due tessere, selezionandole in sequenza.

Realizzare una versione aperta collaborativa multi-player, che consenta a un insieme dinamico ed eterogeneo di utenti in rete di partecipare alla risoluzione del puzzle. Ogni utente che partecipa a una sessione di gioco deve "vedere" anche gli altri utenti (ad esempio il loro puntatore, o altra forma a scelta) e come evolve lo stato del puzzle considerando le azioni di tutti. Quando il puzzle è risolto, la sessione finisce e tutti i partecipanti devono vedere un messaggio corrispondente (esempio: "Risolto!").

Il sistema distribuito deve essere progettato a servizi, in modo che esista - come entry point - un *puzzle service* con una web API funzionale alla partecipazione dinamica di utenti al gioco. Gli utenti in linea di principio potrebbero usare tecnologie client eterogenee. Non ci sono vincoli sulle specifiche tecnologie e modelli di programmazione da usare.

CONSEGNA

La consegna consiste in una cartella “Assignment-03” compressa (formato zip) contenente due directory es1 e es2, una per ogni esercizio. Ognuna delle due sottocartelle deve contenere:

- cartella src con i sorgenti del programma
- cartella doc che contiene una breve relazione in PDF (report.pdf) che include una sintetica analisi del problema, la descrizione della soluzione proposta e i test effettuati per valutare il comportamento dei programmi.