An afternoon at the races

Events portray the activities that go by during a typical afternoon at a hippic center, somewhere at the outskirts of Aveiro. There are four main locations: the *track* where the races take place, the *stable* where the horses rest waiting their turn to enter the competition, the *paddock* where the horses are paraded before the spectators, and the *betting center* where the spectators place their bets on the winning horse.

There are three kinds of intervening entities: the pairs *horse* / *jockey* participating in the races, the *spectators* watching the races and placing bets on the horse they hope to win and the *broker* who accepts the bets, pays the dividends in case of victory and manages in a general manner all the operations that take place.

K races are run during the afternoon, each with N competitors. M spectators are present. The activities are organized as described below

- the broker announces the next race;
- the participating horses are paraded at the paddock by the jockeys;
- the spectators, after observing the horses and thinking about their winning chances, go to the betting center to place their bet;
- the race takes place and one or more horses are declared winners;
- when somebody wins, he or she goes to the betting center to collect the gains.

At the end of the afternoon, the spectators meet at the bar to have a drink and talk about the events that took place.

Each race is composed of a sequence of position increments of the intervening horse / jockey pairs according to the following rules

- the track distance for the race k, with $k = 0, 1, \ldots, K-1$, is D_k units of length;
- each horse / jockey C_{nk}, with $n = 0, 1, \ldots, N-1$ and $k = 0, 1, \ldots, K-1$ carries out a single position increment per iteration by moving randomly 1 to P_{nk} length units along its path – the maximum value P_{nk} is specific of a given horse, because they are not all equal, some being more agile and faster than others;
- the horse / jockey pairs move in parallel paths and may be side by side or overtake one another;
- the winner is the pair horse / jockey C_{nk}, with $n = 0, 1, \ldots, N-1$ and $k = 0, 1, \ldots, K-1$, which, after the completion of an iteration and having overtaken the finishing line, has a position with the highest value;
- in case of a draw, all the horse / jockey pairs with the highest position value are declared winners; the dividends to be received are inversely proportional to their number and rounded to unity.

Assume there are five races, each having four competitors and that the number of spectators is also four. Write a simulation of the activities carried out during *an afternoon at the races*. The simulation shall be based on the client-server model, with server replication, where the broker, the horse / jockey pairs and the spectators are the clients and the access to the information sharing regions are the services provided to them by the servers.

The operations that were assigned to activities previously carried out in the information sharing regions (for the already implemented concurrent version), must now be assigned to independent requests performed on the servers through message passing. In each case, a connection has to be established, a request has to be made, waiting for the reply will follow and the connection has to be closed.

One aims for a solution to be written in Java, to be run in Linux under TCP sockets, either in a concentrated manner (on a single platform), or in a distributed fashion (up to 8 different platforms) and to terminate (it must contemplate service shutdown).

A logging file, which describes the evolution of the internal state of the problem in a clear and precise way, must be included.

Guidelines for solution implementation

1. Specify for each representative server of an information sharing region the structure of the messages to be exchanged.
2. Specify the general organization of the servers architecture.
3. Specify the general organization of the clients architecture.
4. Sketch the interaction diagram which describes in a compact, but precise, way the dynamics of your solution. Go back to steps 1, 2 and 3 until you are satisfied the description is correct.
5. Proceed to its coding in Java as specific reference data types.
6. Specify the mapping of the servers and the clients onto multiples nodes of the parallel machine and write the shell scripts which enable the deployment and the execution of the different modules the application is composed of.
7. Validate your solution by taking several runs and checking for each, through the detailed inspection of the logging file, that the output data is indeed correct.