Pràctica 7. Nivell d'aplicació.
Pràctica de Sockets.
Laboratori d'Aplicacions i Serveis Telemàtics

Josep Cotrina, Marcel Fernandez, Jordi Forga, Juan Luis Gorricho, Francesc Oller

Per un pont poden circular cotxes, que es modelen com a threads, en un sentit o en l'altre però no en ambdós sentits a la vegada. Per això es demana dissenyar una classe `Pont` amb els mètodes `void entrar(boolean sentit)` i `void sortir()`.

Un cotxe quan vol accedir al pont fa una crida al mètode `void entrar(boolean sentit)`, passant com a paràmetre el seu sentit. Si el pont està ocupat per cotxes que circulen en sentit contrari, aquest s'haurà d'aturar. Un cop un cotxe abandona el pont crida el mètode `void sortir()`

Programeu aquest sistema sota el paradigma RPC (Remote Procedure Call) mitjançant pas de missatges (Sockets de Java) i servidor multithreading. La implementació del RPC s'ha de fer seguint el patró stub/skeleton, és a dir, ha d'incloure la classe: stub o representant que fa servir el client i les classes: skeleton/worker (ajudant) que fa servir el servidor

Aspectes a considerar:

- Classe `pont`. El mètode `void entrar(boolean sentit)` té semàntica bloquejant. Els cotxes s'han d'aturar si el pont està ocupat i el sentit no és el seu. El mètode `void sortir()` no ho és.
- No cal tractar cap aspecte de justícia pel que fa a l'accés al pont.
- El Servidor és multifil. Podeu obrir i tancar la connexió cada vegada que el cotxe vol entrar o sortir del pont, o bé mantenir-la oberta durant les entrades i sortides del cotxe.