Laboratorio
Programmazione concorrente e distribuita

Simone Albertini
12 marzo 2013

Cosa serve

- Linguaggio scelto: Java
- Nessun vincolo su che IDE usare
- Si consiglia di aver studiato il materiale delle lezioni col prof. Gallo.

Obiettivi

- Introdurre le strutture messe a disposizione da Java per la programmazione concorrente
  - Oggi si discuterà di questo
- Svolgere esercitazioni
  - Concorreranno a definire il voto finale dell'esame

Difficoltà prog. concorrente

- Dal punto di vista pratico, queste sono le principali difficoltà:
  - Debug
  - “riproducibilità” di una esecuzione (*nondeterminismo*)
  - *Safety*
  - *Liveness*
    - *Race conditions*
    - *Deadlocks*

Problematiche di debug

- Supponendo più thread in esecuzione...
  - E' lo scheduler del SO che determina l'ordine assoluto di esecuzione delle istruzioni di un processo
  - Dunque spesso si può efficacemente controllare solo il risultato finale (*safety*)
  - … se il programma termina (*liveness*)
  - Ma se ci sono problemi è difficile intervenire

Cosa si tocca con mano oggi

- Threads
- Active/passive objects
- Controllo accesso in sezione critica

- Risolvere e identificare Race Condition

Esercizio

- Semplice esercizio di simulazione

  “Al termine di una elezione, le sezioni territoriali devono inviare al ministero i conteggi dei voti ottenuti dai partiti presso i loro seggi

  Il ministero si occupa di aggregarli. Quando il ministero ha ricevuto i risultati di tutte le sezioni, il programma termina.”

- *https://dl.dropbox.com/u/974341/integrity_check_java.txt*

Come procedere

(1) Identificare le entità significative del problema del problema

(2) Identificare quali oggetti sono passivi e quali attivi

(3) Se necessario schematizzare le relazioni tra gli oggetti su carta

(4) … se riuscite a vederle, identificare già le eventuali Race Conditions...

(5) **Implementare**