UNIVERSITÀ DEGLI STUDI DELLA CALABRIA - FACOLTÀ DI INGEGNERIA
CORSO DI LAUREA SPECIALISTICA IN INGEGNERIA INFORMATICA

Esame di SISTEMI DISTRIBUITI

APPELLO DEL 25 GIUGNO 2004 – TEMPO A DISPOSIZIONE: 2.5 ORE

Si consideri un sistema di *garbage collection* distribuita strutturato come segue. Su ogni nodo è presente un oggetto ***ReferenceHandler*** che si occupa di gestire i riferimenti agli oggetti presenti sullo stesso nodo e di tenere aggiornati i processi locali sull'esistenza degli oggetti remoti da essi riferiti. Tale oggetto esporta:

1. un metodo *oggettoLocaleCreato(String idOggetto)* (invocato da processi locali) che riceve l'identificatore di un oggetto locale e memorizza l'identità del processo che ha creato tale oggetto.
2. un metodo *oggettoLocaleDistrutto(String idOggetto)* (invocato da processi locali) che riceve l'identificatore di un oggetto locale che è stato distrutto da un processo locale e memorizza tale informazione.
3. un metodo *oggettoRemotoDistrutto(String idOggetto)* (invocato da ***ReferenceHandler*** remoti) che riceve l'identificatore di un oggetto remoto che è stato distrutto da un processo remoto ed informa i processi locali che ancora riferiscono tale oggetto.
4. un metodo *aggiungiRiferimento(String idOggetto, String idNodo)* (invocato da processi locali e remoti) che riceve gli identificatori di un oggetto locale e di un nodo e memorizza il fatto che un nuovo riferimento a tale oggetto è stato creato da un processo presente su quel nodo.
5. un metodo *rimuoviRiferimento(String idOggetto, String idNodo)* (invocato da processi locali e remoti) che riceve gli identificatori di un oggetto locale *objID* e di un nodo e memorizza il fatto che un riferimento ad *idOggetto* è stato rimosso da un processo presente su *idNodo*. Qualora non esista più nessun riferimento ad *idOggetto*, il ***ReferenceHandler*** informa il processo locale che lo ha creato della possibilità di distruggerlo.
6. un metodo *riferimentiPendenti()*, che restituisce tutte le coppie <*idOggetto*,*idNodo*> dove *idOggetto* è l'identificatore di un oggetto locale già distrutto e *idNodo* è l'identificatore di un nodo che mantiene ancora un riferimento a tale oggetto.
7. un metodo *ripulisci()* che, per ogni oggetto locale già distrutto e verso cui ancora esistono riferimenti, informa i ***ReferenceHandler*** dei nodi che riferiscono tale oggetto del fatto che non è più possibile accedervi.

I ***ReferenceHandler*** prevedono un meccanismo di callback verso i processi locali. Ogni processo locale implementa a tal fine un'interfaccia ***Processo*** con due metodi, *rimuoviRiferimenti(String idOggetto)* e *distruggiOggetto(String idOggetto)*, utili a ricevere dal ***ReferenceHandler*** le informazioni di cui ai punti 3 e 5 (non è necessario fornire la definizione Java di tale interfaccia).

Si progetti il sistema descritto e lo si implementi utilizzando i meccanismi RMI di Java (incluso un semplice processo che usi le funzionalità descritte).

Nota: La struttura dei ***ReferenceHandler*** considerata non sempre evita che un oggetto venga distrutto prima che *tutti* i riferimenti ad esso siano stati rimossi. Si pensi ad esempio al caso in cui un processo *A* riceve da un processo *B* un riferimento ad un oggetto *o*, ed il processo *B* comunica, al ***ReferenceHandler*** del nodo su cui è presente *o*, la rimozione del riferimento ad *o* prima che il processo *A* abbia comunicato l'esistenza del nuovo riferimento. Per evitare tali inconsistenze, è possibile prevedere nei ***ReferenceHandler*** un metodo *riferimentoPassato(String idOggetto,String idNodo)*, che riceve l'identificatore di un oggetto *idOggetto* e l'identificatore di un nodo *idNodo* e memorizza il fatto che un riferimento ad *idOggetto* è stato passato ad un processo presente sul nodo *idNodo.*