ESERCITAZIONE TIGA: NetBeans IDE, Eccezioni, Socket, rete.

Un' azienda che opera nel settore finanziario necessita di eseguire, occasionalmente ma con certezza di integrità, il backup di una quantità limitata di dati. A tale scopo, viene deciso di adoperare tre host, collocati ad almeno 300 km di distanza l'uno dall'altro. Uno degli host é locale, ed i dati vengono inviati attraverso la rete Internet adoperando il protocollo TCP, nella seguente modalità (Fig.1).

L' host che possiede i dati in un file (*Host0*) li invia ad un altro host (*Host1*); questi a sua volta li memorizza in un file, li rilegge e li invia al terzo host (*Host2*), il quale esegue la medesima procedura rinviando i dati all'host di partenza (Host0); infine questi esegue un confronto di uguaglianza dei dati. In tal modo, viene rilevata una qualsiasi alterazione sia sul trasporto che sulla memorizzazione.

Sviluppare un' unica applicazione in Java che, collocata sui diversi host ed eseguibile in due diverse modalità, realizzi il servizio descritto, provvedendo a catturare e gestire le seguenti eccezioni:

`IOException`
`ClassNotFoundException`

visualizzando un messaggio all' utente e garantendo la chiusura dei flussi prima della terminazione.

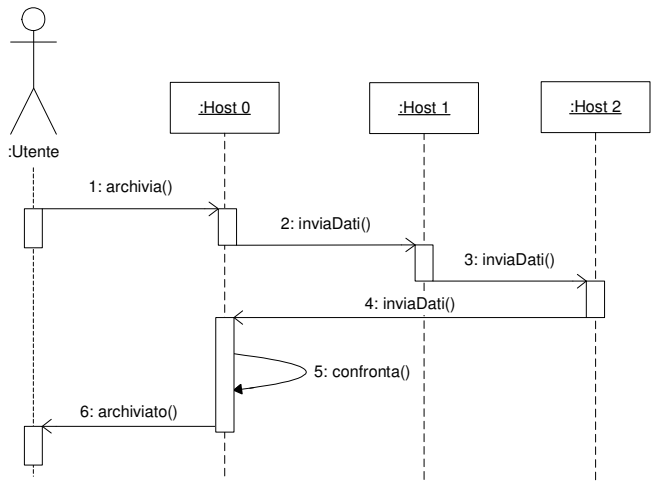

Fig.1 – Diagramma di sequenza del protocollo di backup.

Suggerimenti:

• Supponendo per semplicità che i dati siano costituiti da una sola stringa, si implementi una classe `Host` dal campo stringa e dai seguenti metodi (per gli ultimi due, riusare il codice dei metodi `salva` e `ripristina` del laboratorio su *file*) :

```
public void          inviaDati(String indirizzoIP, int porta)
public String        riceviDati(int porta)
public void          salvaSuFile (String fnome)
public void          leggiDaFile (String fnome)
```

• Si implementi poi una classe `SecureBackup` contenente solo il main, la quale riceve tramite `args` il parametro *modalità* , un intero che indica se l'host deve eseguire le operazioni di *Host0* (cioè creare il file con l'oggetto stringa serializzato, rileggerlo, inviare la stringa e poi aspettare che ritorni dall'ultimo host) oppure di *Host1* (cioè ricevere la stringa, salvarla su file, rileggerla ed inoltrarla all'host successivo). Gli altri parametri sono: 1) il nome del file da creare; 2) l'indirizzo IP dell'host di destinazione; 3) la porta locale di ascolto; 4) la porta remota di ascolto per il server successivo; 5) la stringa da archiviare (per l'*Host0*). Ecco alcuni esempi di esecuzione dell'applicazione:

a) esecuzione su diversi host

SU QUALE HOST	COMANDO DA SHELL	DESCRIZIONE
Host0	`java SecureBackup 0 MioFile.dat 131.121.191.162 8080 8080 CIAO`	Ogni host ha l'indirizzo IP dell' host di destinazione ed ascolta sulla porta 8080.
Host1	`java SecureBackup 1 MioFile.dat 131.124.341.111 8080 8080`	
Hostk	`java SecureBackup 1 MioFile.dat 132.114.291.213 8080 8080`	

b) esecuzione in locale

SU QUALE HOST	COMANDO DA SHELL	DESCRIZIONE
Host0	`java SecureBackup 0 MioFile0.dat 132.114.291.213 8082 8080 CIAO`	L'unico Host può essere riferito anche con l'indirizzo IP di loopback 127.0.0.1 oppure con "". Le porte di ascolto devono susseguirsi per Host adiacenti.
Host1	`java SecureBackup 1 MioFile1.dat 132.114.291.213 8080 8081`	
Hostk	`java SecureBackup 1 MioFile2.dat 132.114.291.213 8081 8082`	

N.B.: *Host0* deve essere avviato per ultimo, dopo che gli altri host sono tutti in "ascolto".

Si ricorda, in ambienti Windows©, di digitare `set CLASSPATH=` all' apertura della Shell.