

## Note sullo svolgimento della prova

- Non è consentito l'uso di vecchi testi d'esame, libri, o appunti.
- Utilizzare NetBeans o Notepad++ come editor, e finestre DOS per compilare ed eseguire i programmi.
- La cartella C:\prg contiene il JDK da utilizzare e la documentazione delle Java API. Eventuali file ausiliari preparati dal docente vanno scaricati da [elearn.ing.unipi.it](http://elearn.ing.unipi.it)
- I file sorgenti relativi all'esercizio 1 devono essere posti nella cartella C:\prg\esercizio1 mentre quelli relativi all'esercizio 2 devono essere posti nella cartella C:\prg\esercizio2.
- Al termine della prova, dopo la sottomissione degli elaborati su [elearn.ing.unipi.it](http://elearn.ing.unipi.it), il docente mostrerà una possibile soluzione. Dopo aver visto la soluzione, gli studenti avranno la possibilità di riprendere il proprio elaborato.

## Esercizio 1

Per fare una pizza sono necessari tre ingredienti: pomodoro, mozzarella e pasta. Intorno a un tavolo ci sono tre pizzaioli. Ogni pizzaiolo ha una riserva infinita di uno solo dei tre ingredienti: il primo di pomodoro, il secondo di mozzarella, il terzo di pasta. Al tavolo è presente anche un fornitore dotato di una riserva infinita di tutti e tre gli ingredienti. Il fornitore permette ai pizzaioli di fare le pizze mettendo sul tavolo due dei tre ingredienti. Più in dettaglio: il fornitore sceglie due ingredienti a caso; il pizzaiolo che ha il terzo ingrediente preleva quelli sul tavolo e fa una pizza. Il fornitore sceglie altri due ingredienti a caso e il ciclo si ripete.

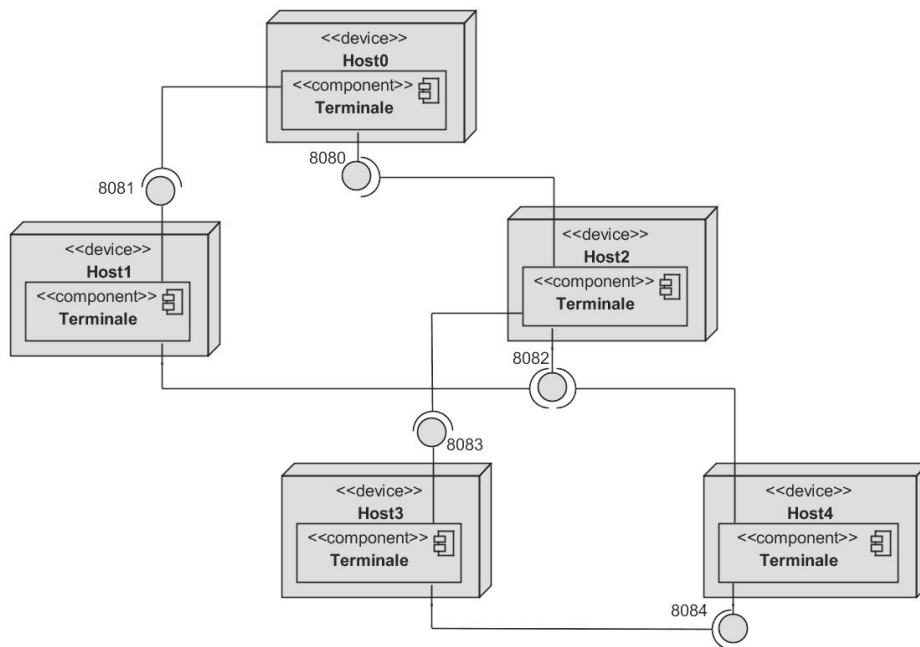
Scrivere le classi necessaria a realizzare quanto descritto in precedenza evitando attese attive.

## Esercizio 2

Un sistema di *environmental crowdsensing* consente il rilevamento di un parametro ambientale (es. il livello di inquinamento dell'aria) attraverso misure provenienti da terminali mobili distribuiti nel territorio. Per maggiore affidabilità ci sono molti terminali per ogni area, in comunicazione secondo una configurazione ad albero binario. I terminali sono interrogati in ordine anticipato (radice, ramo sinistro, ramo destro), e così via in profondità finché il valor medio dei campioni rilevati, arrotondato al valore intero più vicino, non risulta identico tra due successivi campioni. A quel punto tale valor medio arrotondato diventa il parametro rilevato, e quindi la visita dell'albero binario completerà il livello corrente, ritornerà al livello superiore, e via via fino alla radice. Una nuova richiesta di calcolo è elaborata solo quando la richiesta corrente è ritornata alla radice.

Si realizzi un' applicazione Java distribuita, composta dalle classi *Parametro* e *Terminale*. La classe *Parametro* mantiene ed elabora i campioni per ottenere il parametro. La classe *Terminale* svolge le operazioni di trasmissione in formato XML di istanze di *Parametro*, in accordo al protocollo di cui sopra.

**Attenersi esattamente a quanto espresso dalle seguenti figure.** Fig.1 mostra una configurazione di test con cinque terminali, con rispettive porte di ascolto. Fig.2 mostra i file dell'applicazione. Fig.3 presenta le classi da realizzare (in grigio), con i relativi campi e i metodi da creare, i package e le classi da importare e usare. Gestire la chiusura di flussi e connessioni aperti. Catturare solo le eccezioni obbligatorie e gestirle semplicemente stampando le relative informazioni. Fig.4 mostra il file di testing `make.bat`, con alcuni casi di test. Si noti che il terminale posto sulla radice si distingue per avere la porta di ascolto 8080, mentre i nodi interni dell'albero con figli si distinguono per avere quattro variabili di ingresso. Fig.5 mostra la sequenza di passi (ad alto livello) che le istanze di *Terminale* svolgono in un caso di test. La logica di *Terminale* e *Parametro* dovrà essere valida a prescindere dai numeri di porta diversi da 8080 e dal numero di terminali.



- Parametro.class
- Terminale.class
- Parametro.java
- Terminale.java
- make.bat

(viene fornito solo make.bat)

Fig.2 - c:\prg\esercizio2

Fig.1 - Cinque istanze di *Terminale* e relative porte di ascolto

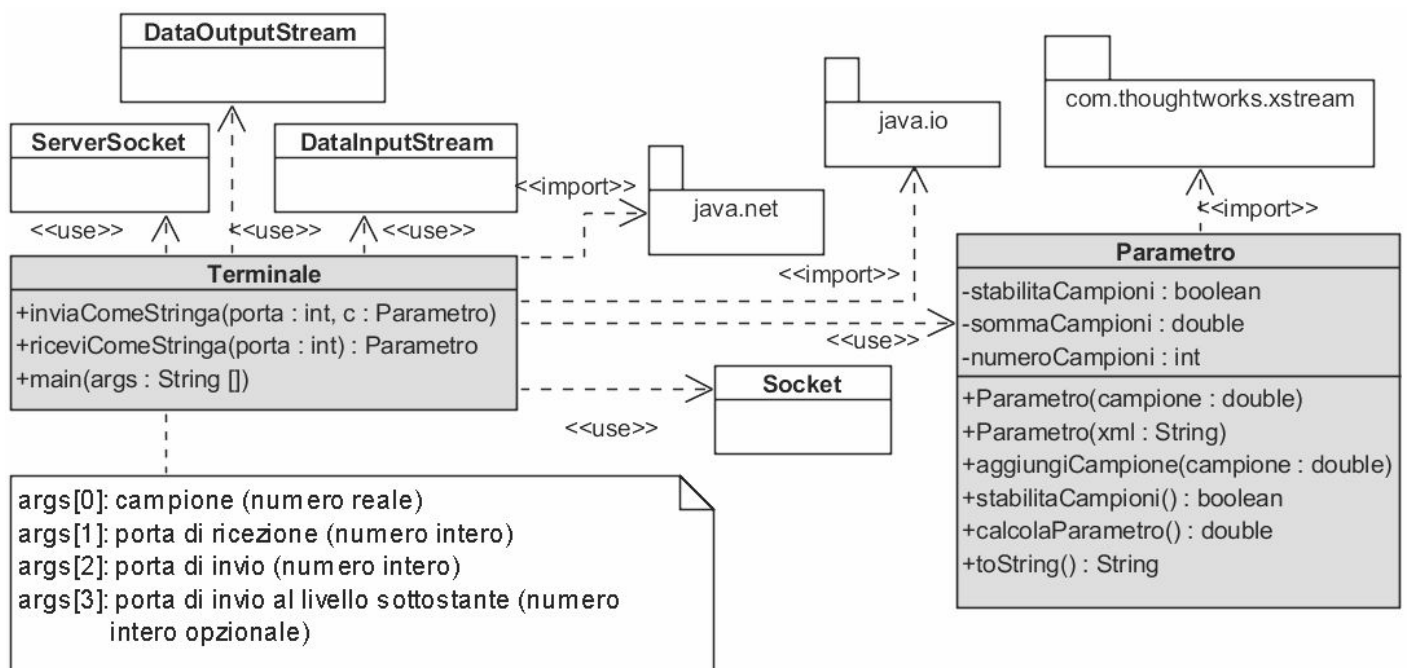


Fig.3 - Classi di cui deve essere composta l'applicazione (in grigio) e classi/package da usare (in bianco)

```

@echo off
c:\prg\jdk8\bin\javac -classpath c:\prg\libs\xstream-1.4.7.jar *.java
pause
set L=c:\prg\libs\
set LIBS=%L%xstream-1.4.7.jar;%L$xmlpull-1.1.3.1.jar;%L\xpp3_min-1.1.4c.jar;
start cmd /k "color 2F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 10.4 8084 8082"
start cmd /k "color 2F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 10.0 8083 8084"
start cmd /k "color 2F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 10.2 8082 8080 8083"
start cmd /k "color 2F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 10.3 8081 8082"
start cmd /k "color 2F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 10.2 8080 8081"
pause
taskkill /f /im "java.exe"
rem campioni stabili a media 10.0 su Host1
pause

start cmd /k "color 3F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 15.4 8084 8082"
start cmd /k "color 3F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 11.0 8083 8084"
start cmd /k "color 3F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 17.8 8082 8080 8083"
start cmd /k "color 3F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 19.3 8081 8082"
start cmd /k "color 3F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 7.5 8080 8081"
pause
taskkill /f /im "java.exe"
rem campioni stabili a media 14.0 su Host4
pause

start cmd /k "color 4F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 30.4 8084 8082"
start cmd /k "color 4F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 50.0 8083 8084"
start cmd /k "color 4F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 17.8 8082 8080 8083"
start cmd /k "color 4F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 19.3 8081 8082"
start cmd /k "color 4F && c:\prg\jdk8\bin\java -classpath %LIBS% Terminale 7.5 8080 8081"
pause
taskkill /f /im "java.exe"
rem i campioni a media 25.0 su Host0, ma non stabili
pause

```

Fig.4 - File make.bat, da non modificare.

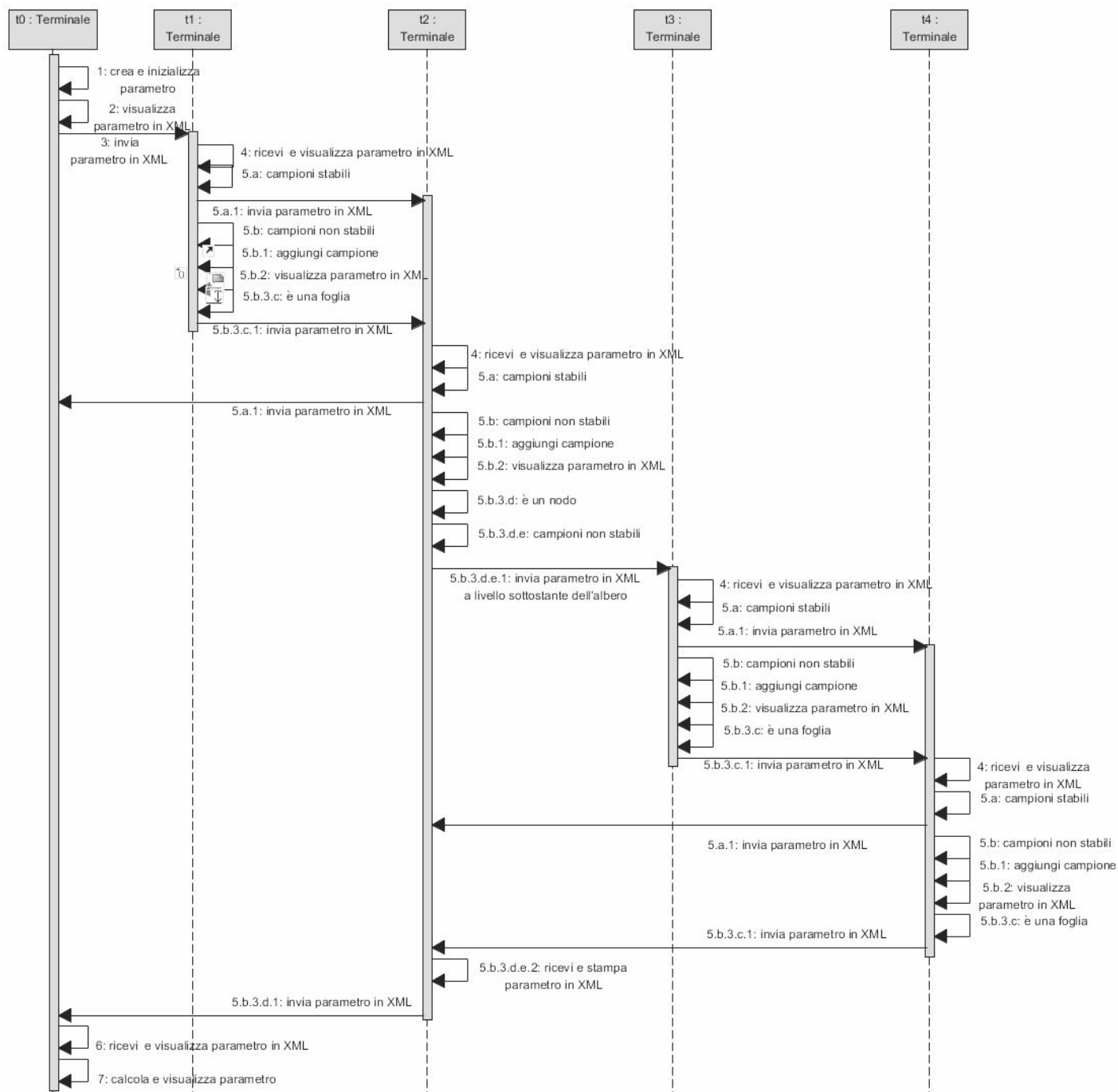


Fig.5 - Sequenza di passi che le istanze di *Terminale* devono svolgere nel un caso di test. Per brevità non viene rappresentata la classe *Parametro*.