Università di Ferrara

Architettura di Reti

Lab 3

Carlo Giannelli
carlo.giannelli@unife.it
http://www.unife.it/scienze/informatica/insegnamenti/architettura-reti/
http://docente.unife.it/carlo.giannelli

Esercizio 1

Si realizzi, utilizzando la Socket API, un'applicazione distribuita Client/Server che permetta a un utente di controllare le spese del proprio conto corrente. Il Client deve presentare la seguente interfaccia:

controllo_conto_corrente server porta

dove "server" e "porta" sono rispettivamente il nome logico dell'host e il numero di porta su cui si trova il Server.

Per prima cosa, il Client si interfaccia con l'utente da cui riceve, via tastiera, una stringa che rappresenta la categoria di spese di cui si vogliono controllare i pagamenti.

Esercizio 1 (cont.)

Il Client dovrà inviare la categoria inserita al Server, che a sua volta dovrà occuparsi di analizzare i dati del conto corrente, ***1) selezionando le spese appartenenti alla categoria richiesta dall'utente e 2) elencandole in ordine decrescente rispetto all'importo***. Infine, il Server dovrà inviare le informazioni al Client, che le stamperà a video.

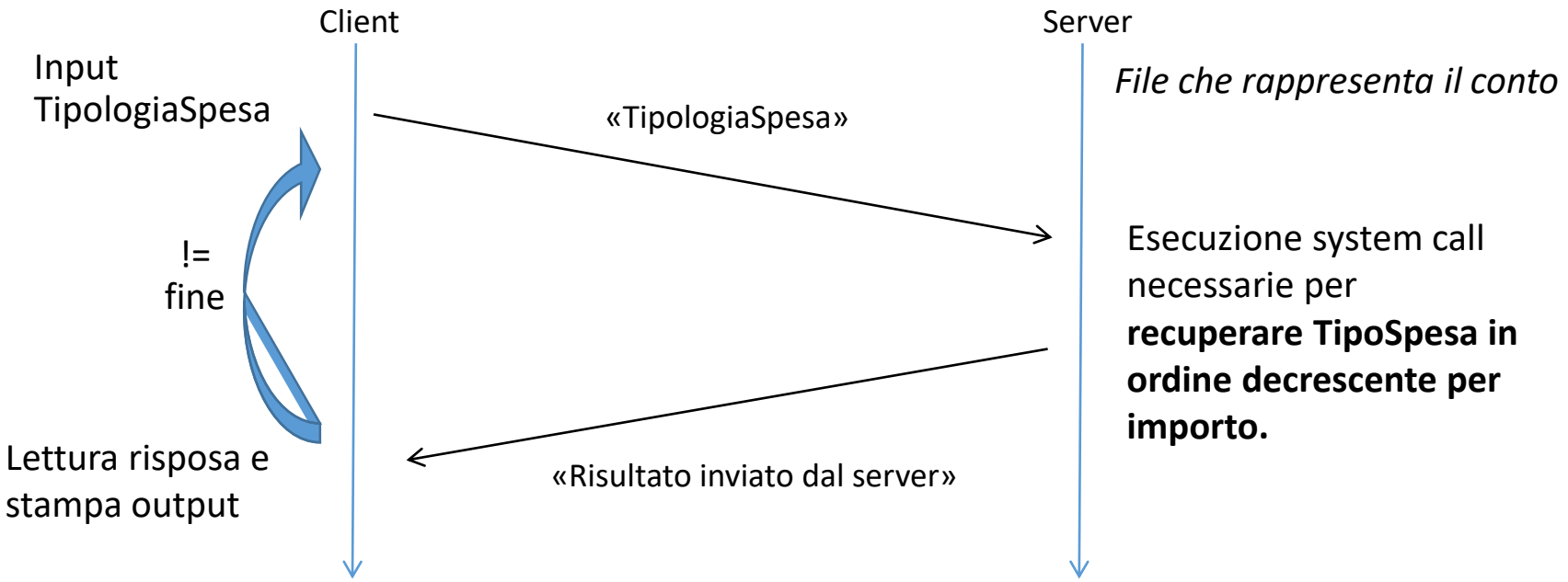
Si supponga che le informazioni sul conto corrente dell'utente siano salvate nel file ***./conto_corrente.txt*** all'interno del filesystem del Server. In particolare, si supponga che ogni riga del file contenga i dati di una singola operazione (**importo, categoria di spesa, data…**).

Al termine di ogni richiesta il Client deve mettersi in attesa della richiesta successiva e terminare qualora l'utente inserisca la stringa "fine". Si realizzi l'applicazione utilizzando il linguaggio Unix/C sia per il Client che per il Server.

conto_corrente.txt

5 spesa
5 altro
5 sale
4 senape
10 spesa
11 soia
34 spesa

Conto Corrente



Attenzione: fork, redirezione, exec (grep e sort), pipe

Esercizio 2

Si realizzi, utilizzando la Socket API, un'applicazione distribuita Client/Server che permetta a un ristoratore di verificare la disponibilità di vini pregiati nella propria cantina. Il Client deve offrire la seguente interfaccia:

verifica_disponibilità_vini server porta

dove "server" e "porta" sono rispettivamente il nome logico dell'host e il numero di porta su cui si trova il Server.

Per prima cosa il Client si interfaccia con l'utente da cui riceve, via tastiera, il nome del vino e l'annata di interesse. Il Client dovrà inviare i dati inseriti al Server, che a sua volta dovrà occuparsi di analizzare i registri del magazzino per reperire le informazioni sui vini richiesti dall'utente. Infine, il Server dovrà inviare le informazioni al Client, che le stamperà a video.

Esercizio 2 (cont.)

Si supponga che le informazioni sul magazzino del ristorante siano salvate nel file **./disponibilita_vini.txt** all'interno del filesystem del Server. In particolare, si supponga che ogni riga del file contenga i dati di una particolare tipologia di vino (**nome, numero di bottiglie disponibili, annata, costo…**).

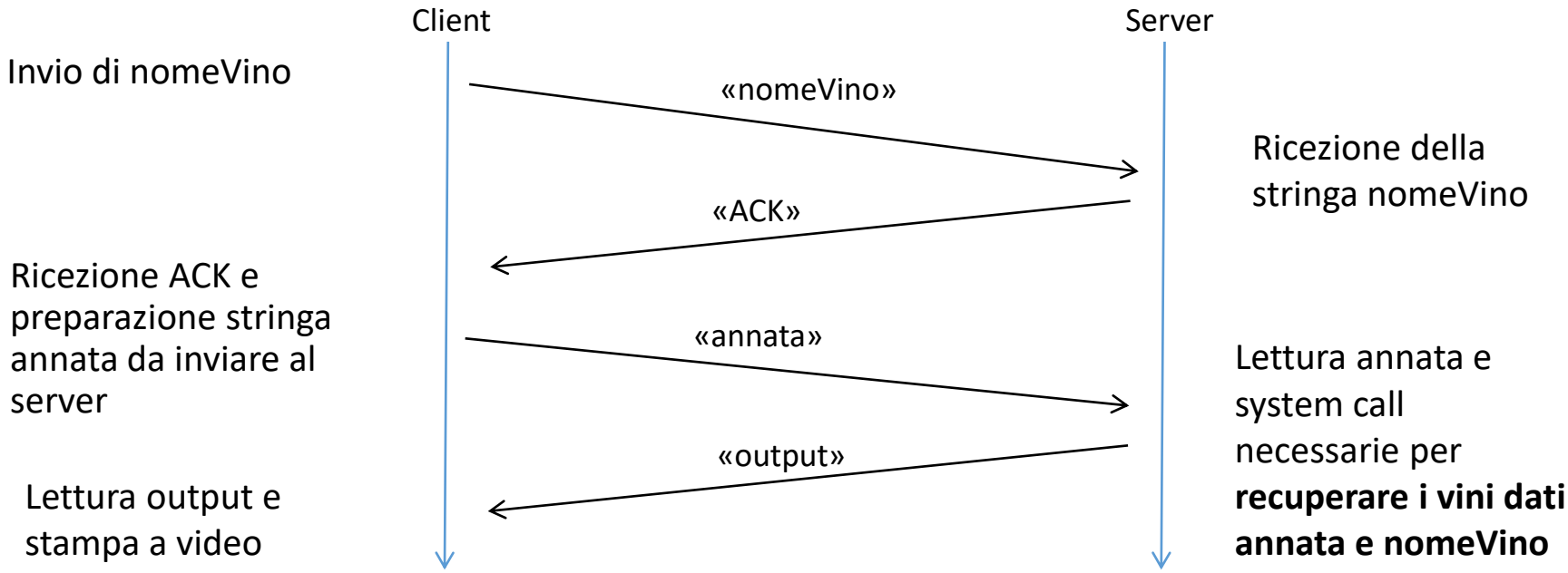
Al termine di ogni richiesta il Client deve mettersi in attesa della richiesta successiva e terminare qualora l'utente inserisca la stringa "fine".

Si realizzi l'applicazione utilizzando il linguaggio Unix/C sia per il Client che per il Server.

disponibilita_vini.txt

VinoRosso 5 2014
VinoNero 6 2013
VinoBlu 5 2014
VinoRosso 6 2015
VinoBlu 9 2014

Verifica Vini



Attenzione: fork, redirezione, exec (doppia grep), pipe