UNIVESITÁ DEGLI STUDI DI MILANO
LAUREA TRIENNALE IN COMUNICAZIONE DIGITALE
PROGETTO LABORATORIO DI RETI DI CALCOLATORI

Dopo Natale puoi...

Matteo Zignani

20 dicembre 2013

1 PRESENTAZIONE DEL PROBLEMA

Il lascito del Natale è tanta felicità e tanti regali indesiderati fatti dai nostri parenti o amici; regali di una bruttezza e inutilità incorporabile. Questo fenomeno è la fortuna di molti siti di aste online, che dopo le festività vedono incrementare il numero di oggetti messi all'asta in attesa che qualche cliente col gusto per l'orrido faccia un'offerta. Questo sistema delle aste ha un piccolo difetto, specialmente in queste occasioni: può capitare che un nostro amico tenti di acquistare un oggetto che lui stesso ci ha regalato. Il progetto che lo studente deve sviluppare tenta di risolvere questo inconveniente. In particolare il servizio da implementare prevede che un utente iscritto possa vendere un oggetto e garantisce che l'oggetto non possa essere visualizzato e acquistato da uno dei suoi amici. Il servizio, quindi, deve gestire gli utenti connessi e la compravendita degli oggetti.
Alla base dell'architettura client/server su cui si fonda il progetto, abbiamo una rete sociale rappresentante le relazioni di amicizia tra gli utenti iscritti al nostro servizio. In generale il servizio permette di vendere e acquistare oggetti, solitamente frutto del Natale, senza offendere i nostri amici. L'erogazione del servizio è garantita e gestita da una serie di API che il servizio stesso mette a disposizione. Le API e le modalità di richiesta delle stesse saranno descritte nella sezione successiva.
Lo studente deve sviluppare solamente la parte server dell'architettura descritta. Per quanto riguarda il client si possono utilizzare vari programmi per la fase di test: telnet (Linux e Mac), putty () o un qualsiasi browser facendo richieste a localhost:<portaServerInAscolto>. Il server DEVE essere MULTITHREAD.

2 API

Nelle seguenti sottosezioni vengono descritti il formato delle API e le risorse offerte dal server inerenti la gestione della social network e della compravendita di oggetti.

2.1 Formato della richiesta

Il server riconosce come valide le richieste che seguono il protocollo HTTP. In particolare il server riconosce solo richieste con metodo di accesso di tipo GET. Per questo motivo lo studente NON deve implementare un server HTTP completo ma un suo piccolo sottoinsieme. Un esempio di possibile richiesta valida può essere:

```
GET <URL Risorsa con eventuali parametri> HTTP/1.0
user-agent: Mozilla 5.0, Chrome, Safari
host: localhost:6000
content-type: text/html
<rigaVuota>
```

Il server deve verificare la validità della riga di intestazione e ignorare tutti i campi settati nella definizione dell'header. Una volta rilevata la stringa vuota di terminazione della richiesta, il server deve eseguire l'operazione richiesta dall'URL e rispondere secondo il formato HTTP. Se la richiesta è andata a buon fine, il formato della risposta è:

```
HTTP/1.0 200 OK

<messaggio>
```

dove < $messaggio$ > è un messaggio testuale che dipende dalla risorsa invocata. Nel caso di un qualsiasi errore il formato della risposta sarà:

```
HTTP/1.0 400 ERRORE

<messaggio>
```

Anche in questo caso < $messaggio$ > contiene un messaggio di errore dipendente dalla risorsa invocata.

2.2 Login

La risorsa login è stat eliminata dal progetto. Per indicare quale utente richiede la risorsa, ogni risorsa richiede un parametro aggiuntivo chiamato $nomeUtente$.

2.3 Vendita di un oggetto

Un utente $nomeUtente$ può vendere un oggetto utilizzando la risorsa $vendi$. La risorsa *vendi* accetta come nomi di parametri validi $nomeOggetto$ e $prezzo$. Il primo indica il nome (stringa) dell'oggetto da vendere mentre il secondo ne specifica il prezzo (numero intero). Un possibile esempio di richiesta di vendita valida è il seguente:

```
GET /vendi?nome=lampadaIkea&prezzo=20&nomeUtente=Aldo /HTTP/1.0
user-agent: Mozilla 5.0, Chrome, Safari
host: localhost:6000
content-type: text/html
<rigaVuota>
```

Una possibile risposta positiva è data da:

```
HTTP/1.0 200 OK

Oggetto lampadaIkea messa in vendita per 20 euro
```

2.4 ACQUISTI DISPONIBILI

Un utente *nomeUtente* può visualizzare gli oggetti messi in vendita utilizzando la risorsa *vediProdotti*. La risorsa non prevede nessun parametro oltre a *nomeUtente*. Si noti che la risorsa restituisce SOLO oggetti che sono stati messi in vendita dai NON-amici dell'utente che effettua la richiesta. Per esempio se A e B sono amici e A vende una lampada, quest'ultima non comparirà nella lista degli oggetti in vendita richiesta da B. Un possibile esempio di richiesta valida è il seguente:

```
GET /vediProdotti?nomeUtente=Aldo /HTTP/1.0
user-agent: Mozilla 5.0, Chrome, Safari
host: localhost:6000
content-type: text/html
<rigaVuota>
```

Una possibile risposta positiva è data da:

```
HTTP/1.0 200 OK

lampadaIkea 20
sediaInFaggio 50
sopramobileInCeramica
dentieraDellaNonna 1500
```

Il corpo della risposta contiene una serie di linee di testo e ogni linea segue il formato $< prodotto >< spazio >< prezzo >$.

2.5 COMPRARE UN OGGETTO

Un utente *nomeUtente* può comprare un oggetto utilizzando la risorsa *compra*. Questa risorsa necessita dell'argomento *nome* in cui si deve specificare il nome dell'oggetto da comprare. Una volta che l'oggetto è stato comprato non è più disponibile nella lista degli oggetti in vendita restituita dalla risorsa *vediProdotti*.Un possibile esempio di richiesta di acquisto valida è la seguente:

```
GET /compra?nome=lampadaIkea&nomeUtente=Aldo /HTTP/1.0
user-agent: Mozilla 5.0, Chrome, Safari
host: localhost:6000
content-type: text/html
<rigaVuota>
```

Una possibile risposta positiva è data da:

```
HTTP/1.0 200 OK

lampadaIkea è tuo!!!
```

2.6 VENDITE EFFETTUATE

Un utente *nomeUtente* può verificare quali oggetti messi in vendita dall'utente stesso sono stati venduti per mezzo della risorsa *vediVendite*. La risorsa non prevede alcun parametro oltre a *nomeUtente*. Si noti che la risorsa restituisce SOLO gli oggetti che sono stati messi in vendita dall'utente loggato e successivamente venduti. Un possibile esempio di richiesta valida è il seguente:

```
GET /vediVendite?nomeUtente=Aldo /HTTP/1.0
user-agent: Mozilla 5.0, Chrome, Safari
host: localhost:6000
content-type: text/html
<rigaVuota>
```

Una possibile risposta positiva è data da:

```
HTTP/1.0 200 OK

lampadaIkea
dentieraDellaNonna 1500
```

2.7 CREAZIONE DELLA SOCIAL NETWORK

Per quanto riguarda la rete sociale che impone i vincoli sulla vendita, essa viene caricata al momento della creazione del servizio e non può essere modificato durante l'esecuzione dello stesso. Il caricamento avviene per mezzo di un file chiamato *listaAmici.txt* e che contiene una lista degli archi non orientati presenti nella rete sociale. Ogni riga del file rappresenta un arco e segue il formato:

```
<nomeUtente>-<nomeUtente>
```

dove < *nomeUtente* > rappresenta un segnaposto (< e > non vanno inseriti). Lo studente deve prevedere una classe per il caricamento della social network da un file. Il file *listaAmici.txt* si trova sul sito del corso di Reti come materiale di supporto al progetto.

3 MODALITÀ DI CONSEGNA

Il progetto è individuale e verranno presi provvedimenti nel caso ci sia il sospetto di una eventuale copiatura. In particolare non verranno ammessi alla discussione del progetto e quindi saranno automaticamente esclusi dall'appello quei progetti con almeno un file con indice di similarità superiore al 55% (moss, tool di Stanford). Questo per non incentivare progetti scritti singolarmente ma frutto di brain-storming troppo di gruppo.
Il linguaggio da utilizzare per lo sviluppo è Java. Il codice del progetto e i relativi jar delle librerie utilizzate devono essere inviati al docente in una directory compressa (zip o tar.gz) denominata 'progettoGennaio2014'. La mail deve avere come oggetto 'ProgettoGennaio2014' e contenere nel corpo nome, cognome e numero di matricola dello studente. Il materiale deve essere consegnato entro le **23:59:59** del **09 gennaio 2014**. Valgono tassativamente le seguenti regole:

- Il codice che non compila non verrà considerato
- I progetti consegnati dopo la data di consegna non saranno ritenuti validi

Per essere considerato sufficiente e quindi ammissibile per la discussione, il progetto deve implementare correttamente tutte le funzioni precedentemente illustrate. Principali criteri di valutazione del codice:

- Gestione di possibili errori nell'input da parte dell'utente
- Modularità del codice e rispetto del paradigma ad oggetti
- Gestione oculata delle eccezioni
- Uso appropriato dei commenti
- Gestione e risoluzione di eventuali problematiche legate alla concorrenza sia lato client sia lato server

Per chiarimenti, dubbi potete contattare il docente all'indirizzo matteo.zignani@unimi.it, specificando come oggetto obbligatorio 'ProgettoReti'. Eventuali modifiche al testo del progetto saranno comunicate sulla pagina web del corso.