Social Gossip

un servizio di chat multifunzionale

Laboratorio di Reti, Corsi A e B
Progetto di Fine Corso – A.A. 2017/18

Piano della presentazione

1. Scopo del progetto
2. Funzionalità
3. Implementazione
4. Modalità di svolgimento e consegna

Scopo del Progetto

progettazione ed implementazione di
Social Gossip,

un servizio sviluppato secondo il modello client-server che offra agli utenti registrati la possibilità di far parte di una rete sociale e di utilizzare diversi tipi di chat.

Servizi di Gestione di Rete Sociale

Il primo insieme di servizi contiene servizi relativi alla gestione della rete sociale, ad esempio:

- la registrazione ed il login sulla rete,
- la ricerca di amici,
- la possibilità di stabilire una nuova amicizia e di ottenere la lista dei propri amici,
- la possibilità di settare il proprio stato di presenza in rete.

Servizi di Chat

I servizi del secondo insieme consentono agli utenti di utilizzare diversi tipi di chat

chat amici

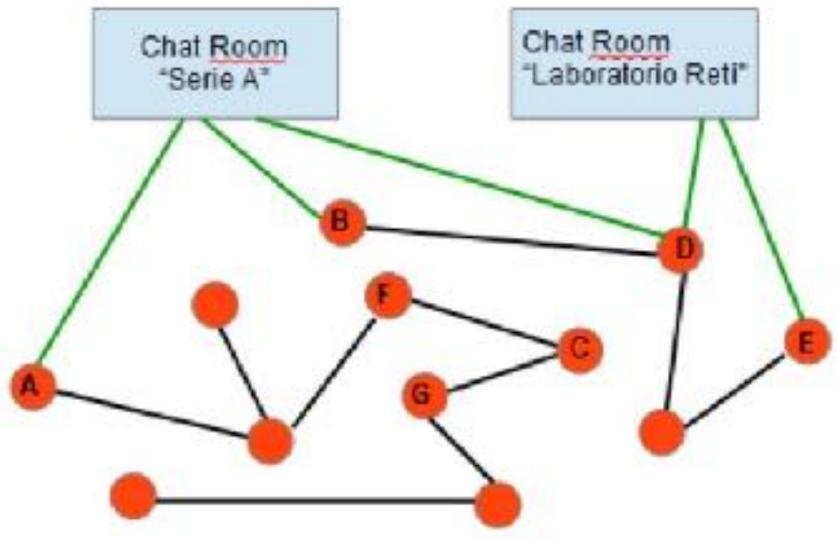
consente di selezionare e quindi chattare con un amico appartenente alla propria rete sociale.
L'utente seleziona l'amico con cui intende chattare e quindi viene creato un canale di comunicazione che collega i due elementi della social network.

Inviare un file mediante chat

servizio di traduzione

chat room

consente di chattare su un particolare tema, con utenti diversi, anche non appartenenti alla propria rete sociale, purché siano iscritti al servizio.



Piano della presentazione

1. Scopo del progetto
2. Funzionalità
3. Implementazione
4. Modalità di svolgimento e consegna

Operazioni sulla Social Network

`register (<nickname><lingua>)`

Il client richiede che l'utente con un certo nickname venga registrato nella social network. Se esiste già un utente con quel nickname, il server comunica che non è possibile inserire l'utente.
Il secondo parametro lingua può assumere i valori ISO 639

`login (<nickname>)`

Il client richiede per l'utente la login al servizio.
Se l'utente è registrato, viene loggato e il server comunica l'avvenuta identificazione, altrimenti comunica che l'utente non è registrato alla social network ed il login non viene effettuato. Questa operazione provoca anche un cambio di stato (da 'offline' a 'online') dell'utente. Ogni client viene notificato del cambio stato di un suo amico

`look-up (<nickname>)`

L'utente richiede di ricercare un altro utente.
Il server informa il client circa il risultato della ricerca

Operazioni sulla Social Network

`friendship (<nickname>)`

Mediante questa comando l'utente richiede che venga creata una relazione di amicizia con un utente specificato.
A differenza di altri social network, in Social Gossip, il server non attende che l'utente accetti l'amicizia, ma inserisce un link nel grafo tra l'utente che richiede l'amicizia e quello specificato come argomento.
Il server informa il client del risultato dell'operazione.
Ogni utente viene informato quando un altro utente crea una relazione di amicizia che lo coinvolge.

`listfriend ()`

Restituisce una lista degli attuali amici dell'utente.

Operazioni di gestione di Chat Rooms

`create (<chatId>)`

Crea una chat-room con identificativo chatId.
chatId è una stringa che rappresenta il titolo della chat, cioè l'argomento di discussione (ad esempio "Serie A").
Nel caso una chat-room con chatId specificato esista già, il server comunica un errore

`addme(<chatId>)`

Aggiunge l'utente che invoca l'operazione alla chat-room specificata dal chatId

`chatlist()`

Restituisce una lista di tutte le chat-rooms, comprese quelle a cui l'utente è iscritto, con indicazione di quelle a cui l'utente è iscritto.

`closechat()`

Chiude una chat room.
Tutti gli utenti che ne facevano parte vengono informati dell'evento.

Operazioni di Chat (amici e room)

`file2friend(<nickname>)`

indica la volontà di un utente di inviare un file ad un amico.
Se l'utente non esiste oppure se l'utente non è al momento online, il server manda un messaggio di errore.

`msg2friend(<nickname><messaggio>)`

l'utente indica il nickname dell'amico con cui vuole chattare ed il messaggio da inviare.
Vengono segnalati eventuali errori se l'utente non è mio amico, se l'utente non esiste oppure se l'utente non è al momento online.
Il destinatario del messaggio riceverà dal server un messaggio in cui viene indicato il nickname dell'utente che ha inviato il messaggio, ed il messaggio tradotto nella lingua del destinatario.

`chatroom-message (<idChat><messaggio>)`

Questo comando consente all'utente che lo invoca di chiedere che il messaggio specificato venga inviato a tutti gli utenti (amici e non) iscritti alla chatroom con identificativo idChat.
Viene segnalato un errore se la chatroom non esiste o se nessun utente iscritto alla chat-room è online (ad esclusione dell'utente che invoca il comando). Gli iscritti alla chat-room riceveranno un messaggio con l'identificativo del mittente, l'identificativo della chat room ed il testo del messaggio.

Piano della presentazione

1. Scopo del progetto
2. Funzionalità
3. Implementazione
4. Modalità di svolgimento e consegna

Server

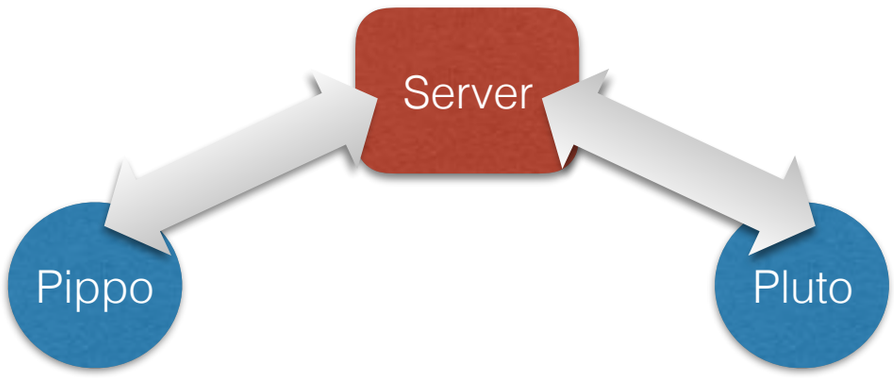
Il server è implementato seguendo l'architettura standard multi-threads
(l'utilizzo di NIO è permesso soltanto per il trasferimento dei file)

Il server memorizza

- il grafo che descrive le relazioni di amicizia tra gli utenti,
- lo stato degli utenti (online/offline).

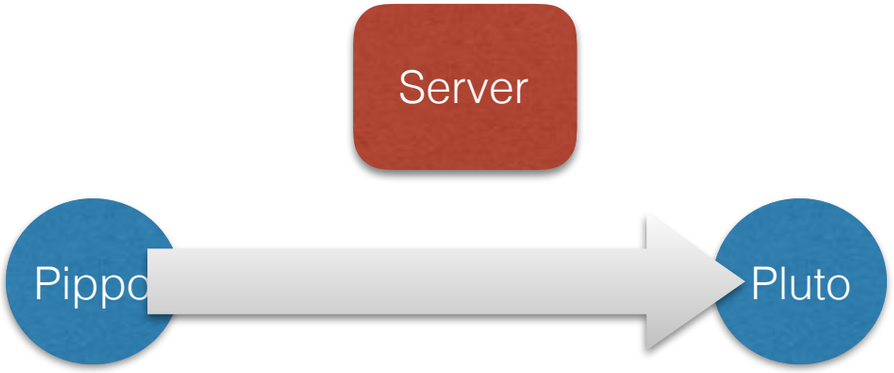
chat

il servizio di chat avviene sempre mediante il server, ovvero il server agisce sempre da intermediario tra i client interessati a chattare.



invio file

l'invio di un file (che è possibile esclusivamente tra amici) avviene in modo diretto tra i client.



Protocolli di comunicazione

Tutte le operazioni richieste dal client ed i rispettivi messaggi di risposta che riguardano

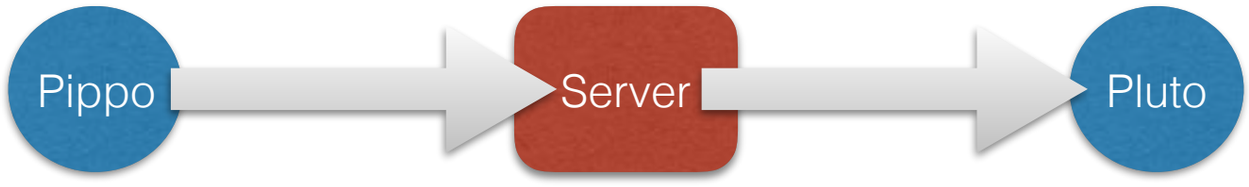
- la gestione della social network
- la gestione delle chat-room

vengono scambiati tra client e server mediante TCP.

Si richiede di definire opportuni formati JSON sia per i messaggi di richiesta che di risposta.

Protocolli di comunicazione

i messaggi della chat-amici,
vengono inviati al server tramite una diversa connessione TCP ed il server li invia all'amico selezionato, sempre mediante una connessione TCP

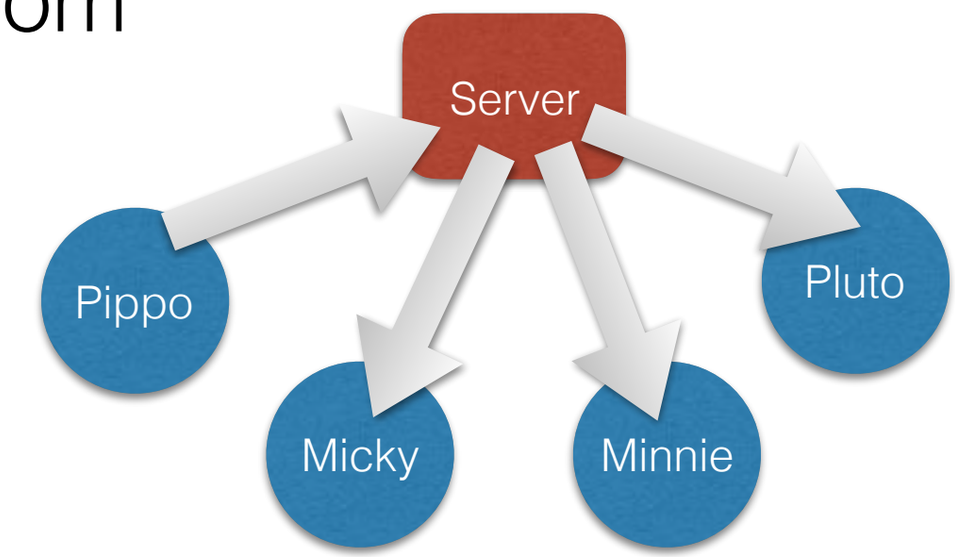


Protocolli di comunicazione

il server crea un gruppo di multicast per ogni chat-room.

Ogni messaggio spedito in una chat room, viene spedito al server mediante il protocollo UDP,

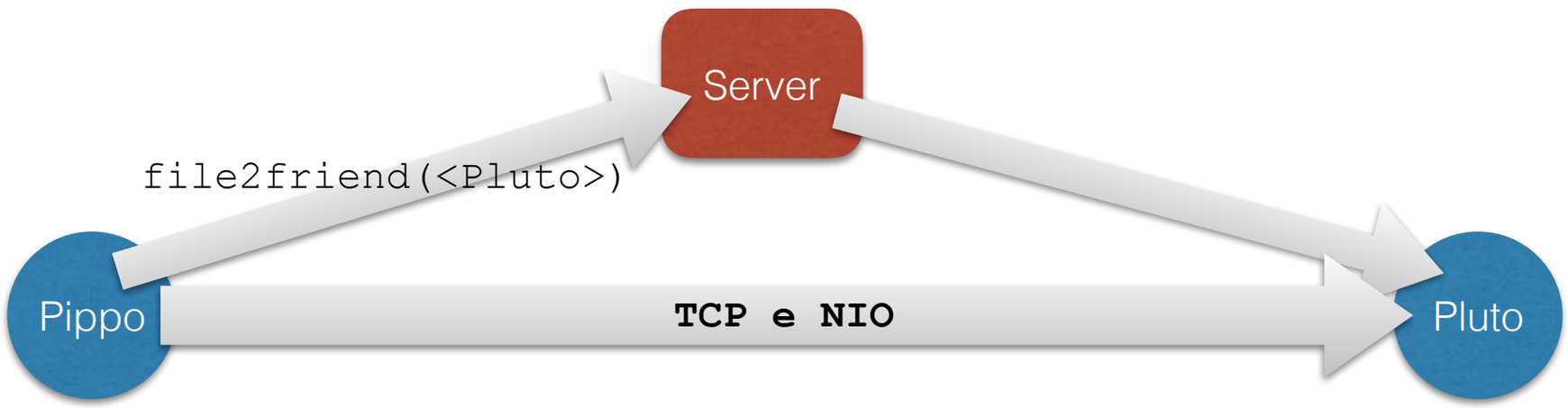
ed il server li spedisce a sua volta sul gruppo di multicast relativo a quella chat-room



Protocolli di comunicazione

per l'invio dei file, su richiesta dell'utente mittente il server contatta l'utente destinatario indicando di aprire un socket per la ricezione del file, e invia al mittente l'ip e la porta del destinatario.

Il trasferimento del file avviene in modo peer-to-peer tra i due client utilizzando TCP e NIO, senza ulteriori operazioni da parte del server.



Protocolli di comunicazione

al momento del login il client registra, tramite RMI, una callback.

Il server utilizza tale callback per informare il client quando uno dei suoi amici cambia stato (online/offline) oppure quando un altro utente ha creato una relazione di amicizia con il client.

Struttura del Client

Lo sviluppo di un servizio di chat è naturalmente implementato definendo un’opportuna interfaccia grafica.

Tuttavia, è possibile anche utilizzare una interfaccia testuale: in questo caso, sono necessarie particolari interazioni dell’utente con l’applicazione per consentire la visualizzazione dei messaggi o di altri eventi in arrivo, come l’aver ricevuto una amicizia o il cambio di stato (online/offline) di un amico.

Altrimenti, l'utilizzo di un' unico terminale
causa problemi

Struttura del Client

l'utilizzo di un unico terminale
causa problemi

```
ClientSG:-msg2friend(Pluto, Ciao Pluto, come stai?)
ClientSG:-chatroom-message (Mickey, Serie A, Il Napoli quest'anno vince tutto)
ClientSG:-_
```

se un messaggio arriva quando
l' utente sta scrivendo

```
ClientSG:-msg2friend(Pluto, Ciao Pluto, come chatroom-message (Mickey, Serie A, Il Napoli
quest'anno vince tutto)_
```

Ad esempio, una possibile soluzione consiste nel far digitare all’utente un comando del tipo ’getIncomingNews’ per poter visualizzare i messaggi che gli sono stati spediti o gli altri eventi in entrata.

Traduzione

Per quanto riguarda la traduzione dei messaggi, il server si può avvalere di alcune REST API messe a disposizione da alcuni providers.

Si consiglia di utilizzare il servizio MyMemoryTranslated.net che offre un semplice servizio di traduzione free, La documentazione è reperibile all'indirizzo http://mymemory.translated.net/doc/spec.php.

L'interazione con il servizio REST di traduzione viene svolta dal server. Il server riceve la richiesta di traduzione, manda la richiesta REST al servizio di traduzione, e invia il messaggio tradotto al ricevente.

Piano della presentazione

1. Scopo del progetto
2. Funzionalità
3. Implementazione
4. Modalità di svolgimento e consegna

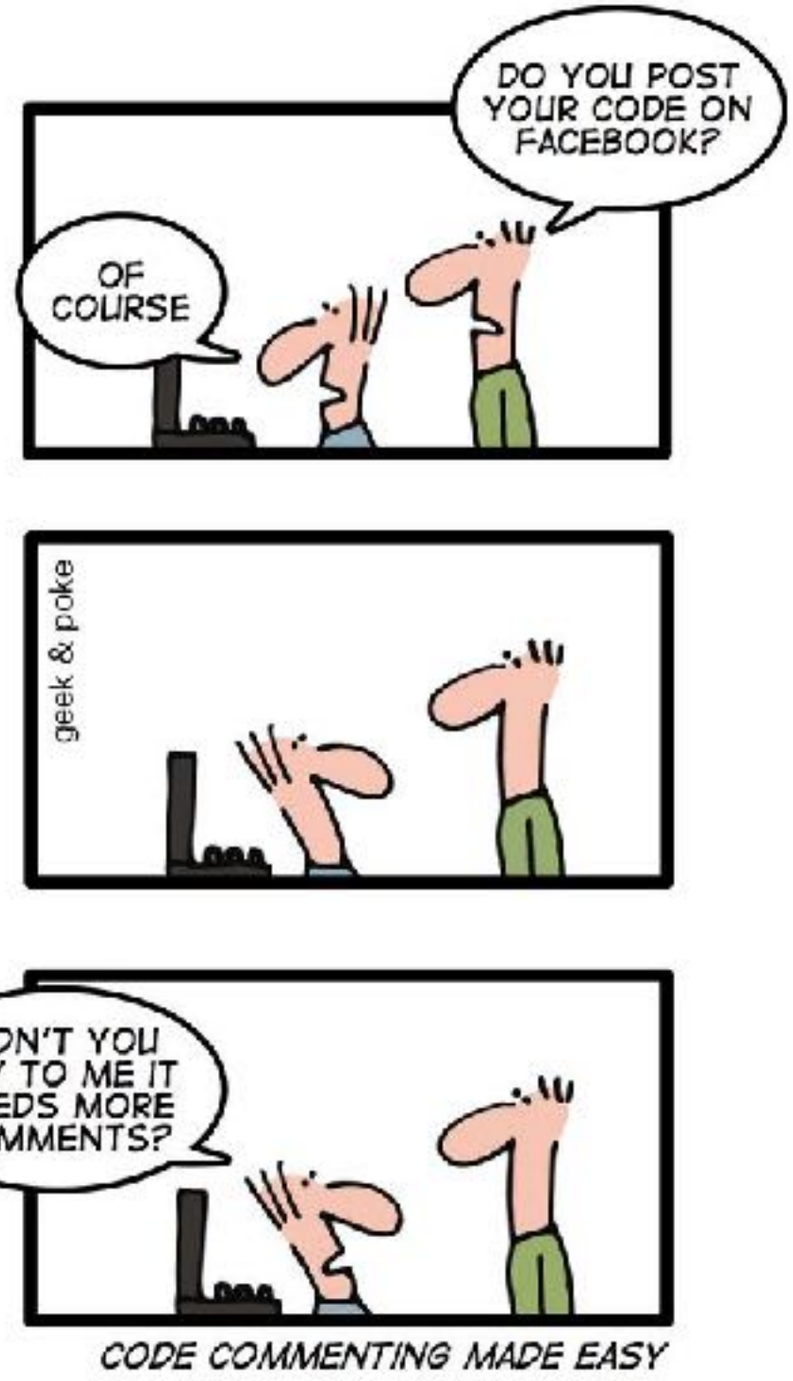
Modalità di svolgimento

Il progetto può essere svolto singolarmente o in coppia.

Se in coppia:

- entrambi gli studenti sono responsabili di tutto il materiale consegnato;
- tutto il codice deve essere capito e conosciuto a fondo da entrambi;
- una prova orale più approfondita (svolta singolarmente) per verificare la comprensione del codice;
- uno studente funge da **revisore del codice** dell'altro.

Code Review


DO YOU POST YOUR CODE ON FACEBOOK?
OF COURSE
geek & poke
DIDN'T YOU SAY TO ME IT NEEDS MORE COMMENTS?
CODE COMMENTING MADE EASY

Modalità di svolgimento

Il materiale consegnato deve comprendere:

- Il codice dell'applicazione e di eventuali programmi utilizzati per il test delle sue funzionalità.

- La relazione in formato pdf che deve contenere:
  - una descrizione generale dell'architettura del sistema. Motivare le scelte di progetto;
  - uno schema generale dei threads attivati, con particolare riferimento al controllo della concorrenza, e delle strutture dati utilizzate;
  - per ogni componente e struttura dati utilizzate, una descrizione delle classi definite (preferibilmente usando anche UML).

- Il codice eseguibile (2 archivi eseguibili .jar, uno per il client, l'altro per il server) con indicazioni precise sulle modalità di esecuzione.

Modalità di consegna

Relazione e codice sorgente devono essere consegnati

- sia in formato cartaceo, presso la portineria del Dipartimento,
- sia in formato elettronico, attraverso il Moodle del corso.

Gli eseguibili devono essere consegnati in formato elettronico.

Relativamente al primo appello, entrambe le consegne devono avvenire prima di
Venerdì 12 Gennaio alle ore 13:00.

Le prove orali si terranno tra Lunedì 15 e Giovedì 18 Gennaio, modulo eventuali eccezioni da concordare con i docenti

Modalità di valutazione

Alcuni fattori che influenzeranno il voto finale:

- gestione della concorrenza lato client e lato server
- modularità del codice e rispetto del paradigma ad oggetti
- gestione oculata delle eccezioni
- uso appropriato dei commenti
- pulizia e chiarezza del codice
- chiarezza della relazione