Caso di studio: "Alberi di decisione"
Metodi avanzati di programmazione (a.a. 2012/2013)

Introduzione

Il caso di studio ha come obiettivo la realizzazione di un sistema che permetta di costruire e consultare degli alberi di decisione[1] a partire da dataset[2] già posseduti.
Il sistema è stato suddiviso e sviluppato in due parti che comunicano tra loro: un *server* ed un *client*.
Il server è stato ideato per essere sempre in funzione su una macchina remota, raggiungibile dai vari client tramite il suo indirizzo IP o DNS[3]. Al suo avvio il server si connette ad un DBMS[4] dove si presuppongono esserci dataset memorizzati in apposite tabelle. Una volta effettuata la connessione esso resta sempre attivo ed eroga servizi ai client che si connettono.
Il client è la parte del sistema dedicata agli utenti. I servizi che offre vengono elaborati in tempo reale dal server. Tali funzionalità includono la possibilità di creare nuovi alberi di decisione a partire da dataset presenti sul DBMS del server o da file propri dell'utente[5], caricare alberi di decisione preesistenti, effettuare predizioni, mandare mail col report di una predizione a destinatari multipli, convertire dataset da un formato in un altro, ecc.

Contenuto della consegna

Nel CD sono presenti i sorgenti di due progetti (sistema sviluppato con l'ausilio di socket e sistema sviluppato con tecnologia RMI), i diagrammi delle classi e la javadoc di entrambi i progetti, i casi di test sulle funzionalità del sistema, i jar eseguibili del sistema, le guide per gli utilizzatori e per gli sviluppatori, i setup dei DBMS attualmente supportati[6] e di altri programmi necessari, il framework Apache Maven[7] e degli script[8] per inserire in un DBMS Postgre o MySQl due dataset di prova: iris e playtennis.

Funzionalità del server[9]

Tutte le funzionalità di base del server, sviluppate nel corso dell'anno, sono state preservate. Ne sono state aggiunte delle altre. Per maggiori informazioni si rimanda alla guida utente del server.
Il server può essere avviato tramite il corrispondente .jar eseguibile oppure da linea di comando per specificare dei personali parametri di connessione. È stato dotato di una semplice interfaccia grafica che permette all'utente di effettuare agevolmente il login ad uno dei DBMS supportati. Una volta effettuata la connessione il server si mostra in una interfaccia suddivisa in due pannelli: in quello a sinistra c'è una lista aggiornata in tempo reale di tutti i client connessi al server, in quello a destra è presente un log di tutte le azioni che il server compie. Le azioni che il server può effettuare sono:

- Costruire un albero di decisione da una tabella del database;
- Serializzare un albero di decisione su un file .dat;
- Caricare un albero di decisione da un file .dat;
- Avviare una predizione sull'ultimo albero memorizzato dal client;
- Chiudere la connessione con un client;
- Inviare una lista delle estensioni supportate per l'upload dei file[5];
- Ricevere un file dal client;
- Inviare una o più email con/senza allegato;
- Verificare la validità di un indirizzo mail;
- Convertire un dataset da un formato all'altro;

Da questo elenco già si evincono alcune delle funzionalità introdotte. La prima è la possibilità di mandare email: al termine di ogni predizione verrà richiesto all'utente se desidera inviare una mail; in caso affermativo, si troverà di fronte ad un pannello nel quale inserire tutte le informazioni per l'invio (indirizzo email del mittente[10] e dei destinatari, oggetto, corpo del messaggio) e in cui potrà specificare se allegare un report (in formato .pdf) della predizione appena conclusa. Tutte le informazioni saranno inviate al server, che dapprima verificherà la correttezza degli indirizzi inseriti ed in seguito elaborerà la richiesta. Se è stato deciso di allegare il report allora il server provvederà a crearne uno, altrimenti invierà direttamente le mail.

Una nuova funzionalità permette di scaricare un file che viene caricato da un client[5]. Il file sarà gestito da un apposito *reader* che lo trasformerà in un dataset e lo userà per costruire un albero di decisione. Infine, viene fornita all'utente la possibilità di convertire, attraverso un *writer*, un dataset in uno dei formati supportati per l'upload in un altro formato oppure in uno script sql.
Due altre funzionalità introdotte nel server consentono di "killare" un client attualmente connesso[14] e di pulire la schermata del log. Entrambe si attivano cliccando col pulsante destro del mouse rispettivamente sul client da "killare" (che deve essere selezionato nella lista) e sul pannello del log.
La veicolazione di informazioni tra client e server è gestita da apposite classi che client e server condividono e che permettono il trasferimento solo delle informazioni strettamente necessarie ai fini di una rapida comunicazione.
Per garantire longevità, manutenibilità ed evolvibilità al sistema sono stati applicati alcuni consolidati principi dell'ingegneria del software e pattern di progettazione. Sono stati di ispirazione i principi S.O.L.I.D. [11], con particolare attenzione per l'*Interface segregation principle* ed il *Dependency inversion principle* (mentre il *Single responsibility principle* ed il *Liskov substitution principle* sono indotti immediatamente dall'applicazione dei principi poc'anzi citati). Tra i pattern di progettazione utilizzati vi sono il *Command pattern*[12] ed il *Transfer object*[13]. Esempi di queste buone norme di programmazione sono riscontrabili nelle varie classi *factory* (adoperate, ad esempio, per le *properties* dei provider di posta elettronica, per il mapping dei tipi Java con i rispettivi tipi SQL, per l'instanziazione di specifiche classi di accesso ai DBMS supportati) e nella classe *executor*[14].

Funzionalità del client[9]

Come per il server, tutte le funzionalità di base del client, sviluppate nel corso dell'anno, sono state preservate. Ne sono state aggiunte delle altre. Per maggiori informazioni si rimanda alla guida utente del client.
Il client può essere avviato tramite il corrispondente .jar eseguibile oppure da linea di comando per specificare dei personali parametri di connessione. Si può inoltre utilizzare il client semplicemente collegandosi al seguente sito web: http://mapgroup.altervista.org/index.html (il server abbinato sarà attivo su una macchina raggiungibile da DNS[3]).
L'interfaccia grafica del client è stata arricchita per adattarsi alle funzionalità introdotte; un esempio è il box *TreeConstruction* che ora prevede la possibilità di caricare un file. Per permettere all'utente di selezionare il proprio dataset viene visualizzata una finestra in cui i file presenti nelle directory della macchina sono automaticamente filtrati secondo le estensioni supportate dal server; una barra di progresso notificherà all'utente lo stato del caricamento. In precedenza sono state già descritte le funzionalità di conversione dei file e quella di invio delle mail

Note

1. Alberi di decisione.
2. Dataset.
3. DNS.
4. DBMS.
5. Per maggiori dettagli vedere la guida utente "Informazioni sull'upload dei file".
6. MySQL e PostgreSQL per Windows (32-bit e 64-bit).
7. Per maggiori dettagli vedere le guide per sviluppatori "Configurazione Maven - Eclipse (ITA)" e "Configurazione Progetti in Eclipse".
8. Per Windows e Linux.
9. Le funzionalità sono sostanzialmente le medesime in entrambi i progetti (socket ed RMI), a meno di modifiche marginali per la gestione dei flussi di informazioni tra client e server (trasparente all'utente).
10. Attualmente è possibile inviare mail esclusivamente dai seguenti provider: gmail (si distingue in velocità dagli altri), yahoo, hotmail, alice, outlook e live.
11. S.O.L.I.D.
12. Command pattern.
13. Transfer object.
14. Solo per l'RMI.