

UNIVERSITA' DEGLI STUDI DI TRENTO



SISTEMI OPERATIVI

AA 2014

CODEC

Autori:

Andrea GHIZZONI

Federica LAGO

1 Analisi del Progetto

Il progetto scelto deve poter funzionare solo per sistemi operativi Linux e consiste nello sviluppo di sistema client-server nel quale il client richiede al server di codificare o decodificare un testo scelto dall'utente attraverso una determinata chiave. La codifica e la decodifica avviene tramite traslitterazione sull'alfabeto inglese. La compilazione di tutto il progetto deve avvenire tramite makefile e produrrà due binari all'interno alla cartella bin rispettivamente per il client e per il server.

Il Server deve rispettare determinate caratteristiche:

- deve essere etichettato con un nome.
- deve rimanere in attesa della connessione di un client
- deve poter accettare la connessione attraverso un messaggio specifico
- deve poter ricevere i dati dal client per poter procedere all'elaborazione
- in caso di un errore inatteso (dati in ingresso malformati, errore nel canale di comunicazione, ecc..) deve produrre un errore opportuno e comunicarlo al client
- l'unico caso di terminazione del server e' attraverso i segnali di terminazione dei processi del sistema

Da testo il server deve poter rispondere alle richieste di piu' client, quindi sara' necessario che non sia lui direttamente a soddisfare tutte le richieste (intese come codifica/decodifica), ma che una terza entita' prenda in carico la richiesta del client e lo serva.

Il client, invece, deve poter accettare i parametri dell'utente, inviarli al server per poterli elaborare e predisporre per la ricezione dei risultati. Al client non e' richiesto di rimanere in esecuzione una volta che ha ricevuto i dati elaborati, ma solo di salvare i risultati su file o stamparli a console.

2 Analisi della Soluzione

Prima di presentare la soluzione riportiamo in seguito la gerarchia delle cartelle del progetto:

- /src contiene tutti i sorgenti necessari
 - /server contiene tutti i file per compilare il server
 - /client contiene tutti i file per compilare il client
 - /util contiene tutte le utility necessarie sia al client che al server.
 - /assets contiene il programma (con sorgenti) del programma generatore degli assets
- /assets conterra' degli esempi di input file generati automaticamente dal comando make assets
- /bin conterra' i binari del server e del client dopo il comando make bin
- /docs contiene i documenti del progetto

Di seguito verra' elencata la nostra soluzione distinguendo tra client, server e makefile.

Makefile

- sono presenti tutti i target richiesti nel testo del progetto (bin, assets, test ecc..)
- il compilatore gcc e' stato messo in condizioni di evidenziare piu' errori possibili e in modo da seguire lo standard ANSI C:
 - Werror: tutti i warning vengono segnati come errori
 - Wall: visualizza tutti i messaggi di warning
 - Wunreachable-code: visualizza un warning quando trova una variabile non usata
 - ansi: utilizza lo standard ANSI C
 - pedantic: il compilatore visualizza piu' warning del dovuto
 - O2: flag di ottimizzazione del compilatore

- il target `bin` non eseguirà direttamente la compilazione del client e del server, ma richiamerà i `makefile` dei rispettivi programmi; i quali compileranno tutte le dipendenze e poi il programma stesso. Ad esempio: nella cartella `src/server` è presente un `makefile` con tre target: `description`, `dep` e `server`: la prima semplicemente stampa una descrizione del `makefile`, la seconda compila tutte le dipendenze del server contenute nella cartella `include` e, infine, il target `server` compila appunto il server con tutte le dipendenze.