Reti di Calcolatori – 10 gennaio 2017

Si progetti un'applicazione Client/Server che, utilizzando le socket, aiuti un programmatore a pianificare il proprio contributo a un progetto software open source fornendogli un elenco dei relativi "issue" (ovverosia di problemi da sistemare, come bug da risolvere e nuove funzioni da implementare). L'applicazione deve presentare la seguente interfaccia:

elenca_issue_oss server porta

dove ***server*** rappresenta il nome logico del Server e ***porta*** rappresenta il numero di porta del servizio.

Per prima cosa, il Client si deve interfacciare con l'utente, da cui riceve (via terminale) il *nome del progetto open source* (es., "NeoVim", "Ruby on Rails", "KDE", ecc.), la *tipologia dei problemi* (es. "interfaccia grafica", "refactoring", ecc.), e il *numero N di issue* di interesse. Il Client deve quindi trasmettere le informazioni al Server, che a sua volta dovrà reperire le informazioni sugli N issue più recenti relativi al progetto open source e alla tipologia di problemi di interesse ed elencarli in ordine di difficoltà crescente e restituirle al Client.

A questo proposito, si supponga che le informazioni sui problemi dei vari progetti open source siano salvate sul Server in una serie di file di testo nella directory */var/local/issues*, ciascuno dei quali conterrà le informazioni su uno specifico progetto software. Quindi, per esempio, le informazioni sul progetto "NeoVim" saranno salvate nel file */var/local/issues/NeoVim.txt*, ecc. Ciascuna riga di tali file conterrà tutte le informazioni relative a uno specifico issue, con (in quest'ordine) il livello di difficoltà (da 1 a 100, dove 100 rappresenta la difficoltà massima) del problema, la tipologia del problema, una breve descrizione del problema, ecc. In ciascun file, le righe sono rigorosamente disposte in ordine di data e ora di pubblicazione del relativo issue. Quindi, per esempio, una riga relativa a un issue pubblicato stamattina comparirà dopo la riga relativa a un issue pubblicato ieri.

Una volta ricevute le informazioni dal Server, il Client le stampa a video e si mette in attesa della richiesta successiva. Il Client deve terminare quando l'utente digita "fine".

ATTENZIONE: Si realizzino il Client e il Server in C, ma il Client deve essere realizzato anche in Java.