

PWEBS

Presentazione

Matteo Merli

<merli@petaligrigi.net>

2 aprile 2003

Indice

1	Introduzione	1
2	Struttura generale	1
3	Gestione delle connessioni	2
4	I moduli	3
5	CGI	4
5.1	PHP	4
6	Cache	4
7	Log	4

1 Introduzione

Pwebs è un progetto amatoriale nato dal desiderio di imparare il linguaggio Python realizzando un'applicazione "reale".

Sostanzialmente si tratta di un server HTTP, pensato per essere il più semplice e funzionale possibile, mantenendo allo stesso tempo la possibilità di aggiungere nuove funzionalità.

Al momento, la maggior parte delle caratteristiche di base sono state implementate. Dove per caratteristiche di base si intendono la gestione delle connessioni e l'implementazione del protocollo HTTP (nella versione 1.1).

2 Struttura generale

Il programma utilizza massicciamente il paradigma della programmazione ad oggetti, sul quale è basato anche Python stesso, ed è suddiviso in moduli¹ e classi.

Un'altra tecnica di programmazione utilizzata, in questo caso per implementare l'esecuzione concorrente, è quella dei threads. Questi vengono impiegati per gestire le singole connessioni con i client, più altre attività del server.

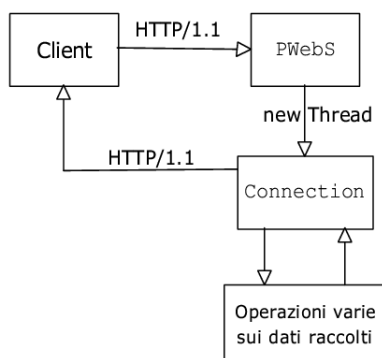
¹In questo caso si parla di moduli Python, che grosso modo possono essere visti come il codice contenuto nei vari file sorgenti separati. Questa definizione di moduli è diversa da quella che verrà poi utilizzata per indicare i moduli del server stesso. Le due cose potrebbero coincidere, ma il concetto di base è diverso.

Ho preferito utilizzare i threads al posto dei processi tradizionali (con la chiamata di sistema `fork()`) per minimizzare l'utilizzo di memoria e l'impatto che un gran numero di connessioni avrebbe sulle performance del sistema.

3 Gestione delle connessioni

Una volta che il server ha terminato la fase di inizializzazione ed è “*up and running*” si pone in ascolto alla porta indicata ed attende connessioni.

Per ogni connessione che viene accettata dal server, viene creato un nuovo thread ed una nuova istanza della classe `Connection`.

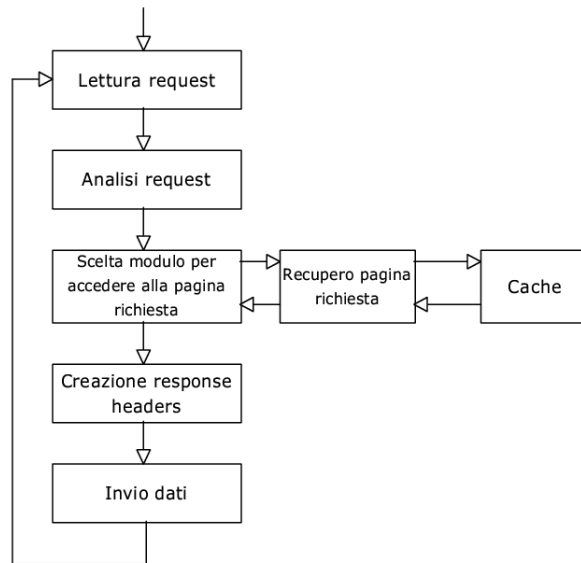


Nello schema precedente `PWebS` e `Connection` sono le due classi che si occupano rispettivamente di accettare le connessioni e gestirle. `Connection` segue tutto il “*ciclo di vita*” della connessione tra il client ed il server, dalla ricezione della richiesta HTTP, alla sua decodifica, la scelta del gestore appropriato per servire la richiesta ed infine l’invio dei dati al client.

Inoltre, se il client supporta le connessioni permanenti², l’istanza di `Connection` non viene distrutta, e si mette in attesa di un’altra richiesta HTTP. Altrimenti viene cancellata l’istanza di `Connection` e viene terminato il thread nella quale era in esecuzione.

In questo schema vengono illustrate le principali fasi della gestione di una connessione.

²Le connessioni `keep-alive`, secondo HTTP/1.1, dovrebbero essere attivate di default se il browser non le disabilita esplicitamente con la direttiva `Connection: close`.



4 I moduli

Uno degli obiettivi prefissi, durante la fase di sviluppo di questo progetto, è quello di dotarlo di una struttura il più flessibile ed ordinata possibile, separando tutte le diverse funzionalità in blocchi di codice diversi ed indipendenti l'uno dall'altro.

Questo concetto può essere interpretato anche in maniera più spinta, cioè si può immaginare che il sistema supporti moduli aggiuntivi che implementano funzionalità in modo completamente autonomo dal *core* del sistema.

Il sistema può essere realizzato utilizzando una API del server che serva come interfaccia di riferimento per realizzare i moduli.

Sono state previste 4 classi di moduli, che mantengono la stessa API, ma si differenziano nel tipo di operazioni che andranno a compiere.

url Moduli che effettuano trasformazioni sull'url fornito dal browser, per esempio per implementare un aliasing sugli url.

mime Moduli che servono per trovare un mime type adeguato alla risorsa che si sta servendo.

response Moduli che gestiscono la parte principale di “*produzione*” della response del server. Un esempio di modulo che ricade in questa categoria è quello di `mod_ssi` per le SSI³. Questo modulo effettua il parsing dei documenti e genera sul momento i documenti HTML definitivi che verranno inviati al client.

encoding Moduli che vanno a modificare la codifica dei dati inviati al client.

I moduli devono essere indicati nel file di configurazione e vengono caricati all'avvio del server. Inoltre vengono anche create quattro liste (una per ogni categoria) che conterranno i riferimenti a tali moduli.

Ogni classe di moduli è pensata per intervenire in un momento ben preciso della gestione della richiesta. Qui il server controlla se ci sono moduli registrati per compiere tale operazione e, in seguito, controlla se tali moduli sono “*interessati*” a compiere qualche operazione. Il compito di valutare l'opportunità o meno di intervenire nel processo viene lasciato al modulo stesso.

³Server Side Includes

5 CGI

Il supporto per la *Common Gateway Interface* consente di realizzare pagine web dinamiche, attraverso programmi scritti in un qualsiasi linguaggio di programmazione che si appoggeranno alla suddetta interfaccia.

L'esempio più classico di linguaggio per realizzare applicazioni CGI è il Perl, ma possono essere utilizzati anche altri linguaggi interpretati o eseguibili binari.

Il concetto dei CGI è infatti quello di utilizzare un programma al posto di una pagina statica; quando il server riconosce che la richiesta effettuata si riferisce ad un CGI, manda in esecuzione l'applicazione, fornendogli eventualmente dei parametri, e infine utilizza l'output come documento effettivo da inviare al client.

L'implementazione dell'interfaccia CGI in Pwebs ricalca la specificazione ufficiale CGI 1.1.

5.1 PHP

Il PHP⁴, è un noto linguaggio "*server side*" che consente di realizzare pagine web dinamiche.

Normalmente, soprattutto per ragioni di performance, l'interprete PHP viene utilizzato come modulo del server web che si sta utilizzando. Esiste però anche la versione indipendente dell'interprete, che può essere lanciata da linea di comando.

E' quindi possibile eseguire applicazioni PHP "quasi" come se fossero dei CGI; le differenze sono infatti minime: la classe PHP è quindi semplicemente un wrapper che si appoggia alle funzionalità già implementate per i CGI.

6 Cache

PWebs utilizza un sistema di cache per aumentare le prestazioni nell'invio di pagine generate dinamicamente.

Questo sistema sostanzialmente tiene in memoria tutte le pagine inviate ai client e, per ogni nuova richiesta, controlla se la pagina è già presente in memoria. Questa tecnica è molto utile anche nel servire pagine statiche, in quanto evita aprire il file e leggere da disco tutte le volte.

In pratica questa cache è realizzata con un dizionario che utilizza come chiave il CRC calcolato sull'URL della risorsa e come dato la pagina (generata oppure statica) che era stata inviata in precedenza.

Ovviamente la cache viene tenuta aggiornata confrontando la data di modifica della risorsa con la data di inserimento nella cache. Se la copia in memoria è vecchia, verrà sostituita da quella aggiornata.

E' possibile impostare, nel file di configurazione, la dimensione massima di questa memoria. Quando viene raggiunto questo valore, vengono tolti elementi dalla cache fino a rientrare nei limiti.

La scelta delle pagine da togliere dalla cache viene effettuata utilizzando l'algoritmo LRU (Last Recently Used).

7 Log

Il sistema di log si occupa di registrare tutte le informazioni che possono essere utili a verificare a posteriori il funzionamento del server.

⁴PHP: Hypertext Preprocessor

Sono stati previsti due file di log con due differenti scopi, uno (`access_log`) per registrare tutte le richieste dei client e l'altro per i messaggi di errore o warning del server per malfunzionamenti non critici.

E' stato scelto di utilizzare per questi log il formato del web server Apache perché in tale modo possono essere analizzati da strumenti automatizzati già esistenti.

E' inoltre presente un sistema di rotating, che archivia i log, comprimendoli e numerandoli in successione quando superano la dimensione di 1Mb.