Tecnologie di Sviluppo per il Web

Programmazione Web: Problemi Tecnologici

versione 3.0

G. Mecca – mecca@unibas.it – Università della Basilicata

Sommario

- Introduzione
- Logica Applicativa
- Richieste e Risposte
- Stato delle Sessioni
  - ⇨Cookies
- Contesto dell'Applicazione
- Riassumiamo i Problemi

Introduzione

- Nel seguito
  - ⇨i principali problemi tecnologici alla base dello sviluppo di un'applicazione Web
- In particolare
  - ⇨sviluppo dell'interfaccia
  - ⇨esecuzione della logica applicativa sul server
  - ⇨gestione delle richieste dell'utente
  - ⇨gestione delle sessioni di lavoro
  - ⇨gestione dell'accesso distribuito

Interfaccia

- Funzioni dell'interfaccia utente
  - ⇨interazione con l'utente
  - ⇨visualizzazione dei messaggi di output e dei "controlli" finalizzati all'acquisizione di dati dall'utente
  - ⇨produrre "eventi" sulla base dei gesti dell'utente
- Nelle applicazioni Web
  - ⇨l'interfaccia utente è resa dal browser

Interfaccia

- Il livello del client
  - ⇨l'utente interagisce con il browser
  - ⇨il browser produce gli schermi
  - ⇨il browser cattura i gesti dell'utente
- La tecnologia del browser
  - ⇨basata sul linguaggio HTML
  - ⇨gli schemi devono essere costruiti utilizzando il linguaggio HTML

Interfaccia

- Controlli grafici forniti dall'HTML
  - ⇨testo, immagini, collegamenti ipertestuali
  - ⇨elementi delle maschere HTML ("form")
- Il modulo "Basic Forms" di XHTML
  - ⇨insieme di controlli relativamente povero
  - ⇨non esistono controlli di tipo "tabella", "albero", "file chooser", "color chooser" ecc.

Interfaccia

- Gestione degli eventi
  - ⇨gli eventi vengono generati dal browser, ma devono essere gestiti sul server
- Eventi significativi per il server
  - ⇨solo quelli che generano una richiesta HTTP
  - ⇨l'utente schiaccia il bottone di una form (richiesta get oppure post)
  - ⇨l'utente seleziona un collegamento (…, richiesta get)

Interfaccia

- Attenzione alla differenza
  - ⇨ci sono molti eventi scatenati dalla pagina HTML che sono gestibili sul client (es: utilizzando JavaScript – modifica di un campo di testo, sorvolo di un'immagine ecc.)
  - ⇨ma in questo caso l'applicazione gira sul server, e quindi sono significativi esclusivamente gli eventi percepiti dal server

Interfaccia

- Un'altra differenza importante
  - ciascuna risposta corrisponde ad uno schermo, ovvero ad un blocco di codice HTML inviato al browser
  - il browser a seguito della risposta non mantiene lo stato dello schermo precedente
  - viceversa, lo schermo viene ogni volta completamente ridisegnato dal browser

Interfaccia

- Prime lezioni importanti
  - la tecnologia della vista (interfaccia utente) è più povera rispetto a quella dei toolkit grafici tradizionali (es: Swing) – meno controlli
  - gli schermi vanno rigenerati sempre da zero
  - il sistema di gestione degli eventi è molto più "scarno"
  - gli unici eventi gestibili sul server sono quelli che corrispondono a richieste HTTP

Esecuzione

- Problema di comunicazione
  - ⇨le richieste HTTP vengono intercettate da un server HTTP
  - ⇨di per sè il server è fatto per fornire l'accesso a documenti sul file system
  - ⇨in questo caso, però, le richieste devono innescare l'esecuzione di applicazioni
  - ⇨quindi il server HTTP deve essere in grado di eseguire programmi

Esecuzione

- Il processo di comunicazione
  - ⇨richiesta del client – server – applicazione – server – risposta al client
  - ⇨ci vogliono delle regole per eseguire i vari passi e codice che le implementi
- In particolare
  - ⇨serve un modulo nel server HTTP che funga da "server applicativo" per la comunicazione con le applicazioni

Esecuzione

- Nel nostro esempio
  - un esempio molto semplice di comunicazione applicativa
  - basata su Common Gateway Interface (CGI)
  - protocollo di comunicazione tra server Web e applicazioni
  - soluzione originariamente introdotta da Netscape
  - sempre meno utilizzata al giorno d'oggi

Esecuzione

- Idea
  - il server HTTP deve contenere un modulo per gestire applicazioni CGI
  - viene configurato specificando quali cartelle contengono applicazioni CGI
  - nel caso di richieste agli URI delle cartelle in questione, bisogna eseguire il programma e non restituire il codice sorgente

Esecuzione

- Se arriva una richiesta ad un URI CGI
  - ⇨il server Web esegue il programma
  - ⇨per la comunicazione utilizza i flussi standard del programma: standard input e standard output, che vengono opportunam. rediretti
  - ⇨il server invia al programma la richiesta HTTP scrivendola sullo standard input
  - ⇨il server riceve dal programma la risposta HTTP dallo standard output

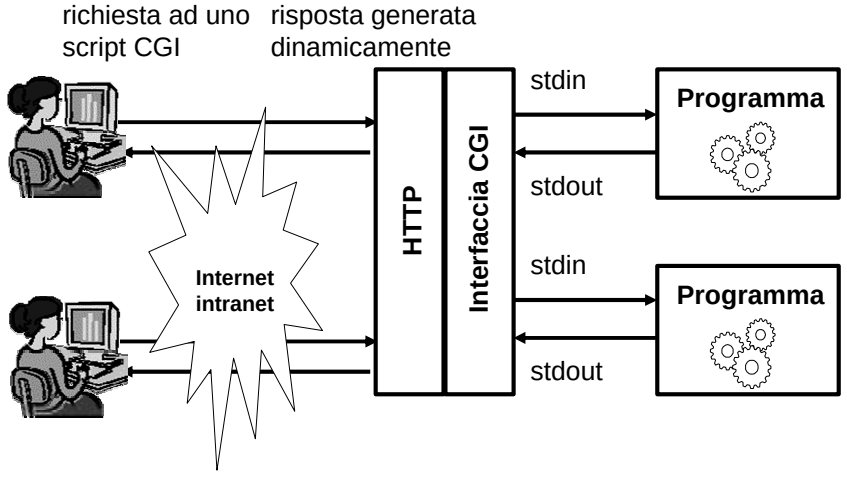
Esecuzione

- In aggiunta
  - ⇨all'interno del processo in cui viene eseguita l'applicazione, sono visibili delle variabili di ambiente inizializzate dal modulo CGI
  - ⇨attraverso le variabili vengono comunicati alcuni dati relativi alla richiesta
  - ⇨es: metodo (get o post), intestazioni (es: content length), eventuale query string nell'URI ecc.

Esecuzione

- Ciclo di vita del programma da eseguire
  - ⇨per ogni richiesta viene attivato un nuovo processo
  - ⇨il programma acquisisce i dati dallo std. input
  - ⇨il programma effettua le sue elaborazioni
  - ⇨il programma produce la risposta sullo std. output
  - ⇨infine termina e non lascia traccia
  - ⇨il tutto ricomincia alla richiesta successiva

Esecuzione



Esecuzione

- Problema n. 1: Esecuzione poco efficiente
  - ⇨ad ogni richiesta viene attivata una nuova istanza dell'applicazione
  - ⇨che viene distrutta al termine della transazione
  - ⇨nota: eseguire e distruggere processi è costoso per il sistema operativo

Esecuzione

- Problema n. 2: Esecuzione poco efficace
  - ⇨meccanismo molto limitato di comunicazione con il server HTTP
  - ⇨l'applicazione deve decodificare i dati forniti dall'utente sullo standard input
  - ⇨l'applicazione comunica con il server Web solo attraverso lo stdout
  - ⇨non è possibile accedere ad altri servizi; es: non può accedere al file di log del server

Esecuzione

- "Indovina il numero" su Web
  - ⇨la form iniziale è contenuta in una pagina HTML, indovina.html
  - ⇨scatena l'esecuzione di un assembly .NET attraverso il protocollo CGI
  - ⇨l'assembly è contenuto in una cartella configurata sotto Apache come cartella di applicazioni CGI

Esecuzione

>> indovina.html
>> indovinaINumeroCgi.exe

- L'esecuzione
  - ⇨ogni nuova richiesta esegue daccapo l'assembly (il metodo main di Principale)
  - ⇨l'oggetto principale crea una serie di oggetti secondari che servono a gestire adeguatamente la comunicazione
  - ⇨Richiesta, Risposta, Sessione, Applicazione

Richieste e Risposte

- Le richieste che arrivano al server
  - ⇨richieste HTTP
- Le risposte fornite dal server
  - ⇨risposte HTTP
- Quindi, il programma CGI deve
  - ⇨analizzare la richiesta HTTP
  - ⇨effettuare le elaborazioni
  - ⇨produrre una risposta HTTP

Richieste e Risposte

- Richiesta HTTP
  - ⇨stringa passata sullo standard input
  - ⇨metodo e URI, intestazioni, corpo
  - ⇨bisogna analizzarla i parametri forniti dall'utente attraverso la query string
  - ⇨es: nome, tentativo
  - ⇨ed eventuali altre intestazioni
  - ⇨è necessaria un'analisi sintattica; nell'esempio svolta dall'oggetto Richiesta

Richieste e Risposte

>> Richiesta.java

- Codice della query string
  - ⇨se la richiesta è di tipo get, è ottenibile consultando il valore della variabile di ambiente QUERY_STRING
  - ⇨altrimenti è contenuto nel corpo della richiesta HTTP e deve essere prelevato dallo standard input
  - ⇨è necessario effettuare la decodifica
  - ⇨e poi l'estrazione delle coppie nome, valore

Richieste e Risposte

- Una annotazione interessante
  - ⇨i componenti dell'applicazione rispondono a URI pubblici
  - ⇨possono ricevere richieste scorrette
  - ⇨è in generale necessario verificare che la richiesta sia corretta prima di elaborarla
  - ⇨es: richiesta in cui non c'è nè il parametro nome nè tentativo nella query string

Richieste e Risposte

>> Risposta.java

- Risposta HTTP
  - ⇨stringa prodotta sullo standard output
  - ⇨e restituita dal server HTTP al client
  - ⇨codice di risposta, intestazioni, corpo
  - ⇨il corpo contiene codice HTML costruito “al volo”, a seconda dello stato dell’applicazione
  - ⇨bisogna inoltre produrre correttamente le intestazioni (es: Content-Type)
  - ⇨compito svolto dall’oggetto Risposta

Richieste e Risposte

- Una caratteristica tipica delle appl. Web
  - ⇨i componenti di controllo (es: Principale) ricevono le richieste
  - ⇨eseguono la logica applicativa
  - ⇨producono lo schermo successivo (es: maschera per il tentativo)
  - ⇨in alcuni casi, sono costretti ad inoltrare le richieste a componenti diversi per produrre schermi diversi (es: SchermoFinale.cs)

Richieste e Risposte

- Problema n. 3: Gestione della Richiesta
  - ⇨ogni volta devo re-implementare l'analisi sintattica
  - ⇨facendo attenzione a tutti i dettagli (es: URI encoding, dettagli della sintassi ecc.)
- Problema n. 4: Produzione della Risposta
  - ⇨stampare stringhe HTML sullo standard input è lento e scomodo (es: virgolette)

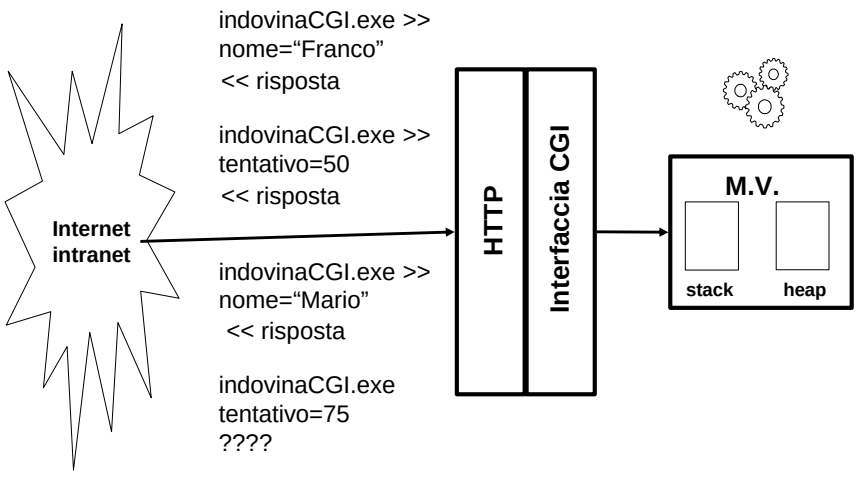
Stato delle Sessioni

- Il protocollo HTTP è privo di stato
  - ⇨ogni richiesta non ha memoria delle precedenti
  - ⇨il protocollo non prevede di associare due richieste successive dello stesso utente
- Problema n. 5: Gestire lo stato dell'applic.
  - ⇨problema molto serio
  - ⇨è difficile mantenere lo stato della sessione

Stato delle Sessioni

- Sessione di lavoro
  - ⇨sequenza di operazioni effettuate da un utente con l'applicazione fino alla chiusura
- In un'applicazione tradizionale
  - ⇨lo stato della sessione corrisponde allo stato della memoria (stack e heap)
  - ⇨ovvero allo stato del Modello
  - ⇨due click successivi dell'utente lavorano sugli stessi oggetti della memoria

Stato delle Sessioni



Stato delle Sessioni

- Problema delle richieste
  - ⇨il server non è in grado di sapere se richieste diverse appartengono alla stessa sessione (sono state generate dallo stesso utente)
  - ⇨oppure sono dovute ad accessi concorrenti da diversi utenti
- Problema del ciclo di vita
  - ⇨l'applicazione comincia e finisce ogni volta e non resta traccia della memoria

Stato delle Sessioni

- Di conseguenza
  - ⇨serve un meccanismo aggiuntivo rispetto a HTTP+CGI
  - ⇨un meccanismo che consenta di associare le richieste ad una sessione
  - ⇨e che consenta nelle richieste successive di ripristinare lo stato dell'applicazione (es: nome, numero da indov. e tentativi effettuati)
- Per esempio: i cookies

Cookies

- Cookies
  - ⇨originariamente introdotti da Netscape
  - ⇨meccanismo per tenere traccia dello stato della sessione in applicazioni Web
  - ⇨standardizzati successivamente
  - ⇨IETF RFC 2109 – HTTP State
- Attenzione
  - ⇨il loro uso è controverso (privatezza)

Cookies

- Idea fondamentale
  - ⇨utilizzare le intestazioni HTTP
  - ⇨due nuove intestazioni
- Set-Cookie
  - ⇨nelle risposte del server per inviare un cookie
- Cookie
  - ⇨nelle richieste del client per restituire i cookie

Cookies

- Ogni cookie è una stringa di caratteri
  - ⇨fatta di una o più coppie nome=valore associate ad una porzione di un sito
- Porzione di sito
  - ⇨il cookie è associato ad URI principale (es: http://www.unibas.it/didattica)
  - ⇨è valido per tutti gli URI che contengono l'URI principale come prefisso (es: http://www.unibas.it/didattica/vai/news.html)

Cookies

- Generazione del cookie
  - ⇨il cookie viene creato dal server
  - ⇨viene proposto al client attraverso l'intestazione "Set-Cookie"
- Accettazione del cookie
  - ⇨il client può accettare o rifiutare il cookie
  - ⇨a seconda delle impostazioni del browser (normalmente configurabili dall'utente)

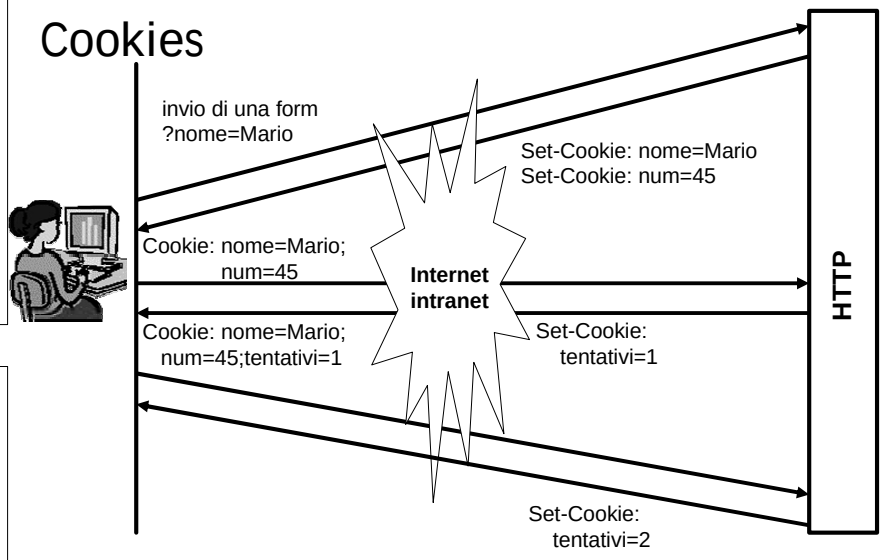
Cookies

- Se il browser accetta il cookie
  - ⇨lo salva sul disco locale, in un'area precisa
  - ⇨si impegna a re-inviarlo al server con l'intestazione "Cookie" per tutte le altre richieste che riguardano gli URI relativi
- In altri termini
  - ⇨il server può utilizzre i cookie per "salvare" lo stato della sessione sul disco del client
  - ⇨e ripristinarlo alla richiesta successiva

Cookies

- Validità di un cookie
  - ⇨tipicamente: finchè non viene chiusa la finestra del browser
  - ⇨in questo caso si dice che il cookie ha "validità di sessione" (dura finchè dura la sessione di lavoro del client)
  - ⇨ma è configurabile (esistono cookie eterni)

Cookies



Cookies

- "Indovina il numero" su Web
  - ⇨utilizzando i cookies posso "salvare" sul browser nome dell'utente, numero da indovinare, tentativi effettuati
  - ⇨ad ogni nuova richiesta, ripristino lo stato a partire dai cookie, e decido come proseguire
  - ⇨alla fine invio la risposta e aggiorno il valore dei cookie sul client (numero di tentativi)

Cookies

>> Sessione.java
>> Risposta.java

- Questi compiti sono svolti da
  - ⇨l'oggetto Sessione, che a partire dai cookie associati alla richiesta è in grado di ripristinare lo stato della partita
  - ⇨l'oggetto risposta, che data una Sessione produce i cookie aggiornati per il client in modo che la Sessione li possa ripristinare alla richiesta successiva

Contesto dell'Applicazione

- Nelle applicazioni multiutente
  - ⇨oltre ai dati delle sessioni (relative al singolo utente) è utile mantenere dati relativi all'applicazione nel suo complesso
  - ⇨esempio: numero di giocatori
  - ⇨esempio: statistiche sugli utenti di un forum di discussione
  - ⇨serve a dare all'utente la percezione dell'esistenza degli altri utenti

Contesto dell'Applicazione

- Si tratta di un problema nuovo
  - ⇨rispetto alle applicazioni monoutente
- In particolare
  - ⇨il funzionamento dell'applicazione consiste di varie sessioni
  - ⇨tutte le sessioni sono riconducibili allo stesso "contesto dell'applicazione"

Contesto dell'Applicazione

- Attenzione
  - ⇨in questo caso i cookies non aiutano
  - ⇨sono collegati all'interazione client/browser e non sono condivisi (pb. di aggiornamento)
- Soluzione tradizionale con CGI
  - ⇨i dati vengono salvati persistentemente
  - ⇨le applicazioni salvano e recuperano i dati da un file sul disco oppure da una base di dati

Contesto dell'Applicazione

>> Applicazione.java

- "Indovina il numero" su Web
  - ⇨il numero degli utenti che stanno in quel momento giocando viene salvato in un file
  - ⇨ogni volta che si aggiunge un utente il numero viene incrementato
  - ⇨ogni volta che un utente termina il gioco il numero viene decrementato

Contesto dell'Applicazione

- Problema n. 6: Contesto dell'applicazione
  - ⇨programmare sui file è scomodo e soggetto ad errori
  - ⇨inoltre c'è un problema di sincronizzazione (richieste diverse che lavorano sul contesto)
  - ⇨la gestione della sincronizzazione è affidata al file system, e non sempre è completamente adeguata alle necessità

Riassumiamo i Problemi

- In sintesi
  - ⇨lo sviluppo di applicazioni Web è decisamente diverso dallo sviluppo di applicazioni tradizionali
  - ⇨pone una serie di problemi tecnologici notevoli e di difficile soluzione
  - ⇨richiedono un'interazione a basso livello con il server HTTP

Riassumiamo i Problemi

- Riassunto dei problemi
  - ⇨Problema n. 1: esecuzione poco efficiente
  - ⇨Problema n. 2: esecuzione poco efficace
  - ⇨Problema n. 3: gestione della richiesta
  - ⇨Problema n. 4: produzione della risposta
  - ⇨Problema n. 5: stato della sessione
  - ⇨Problema n. 6: contesto dell'applicazione

Riassumiamo i Problemi

- Idea
  - ⇨piuttosto che lavorare a basso livello, è opportuno sfruttare piattaforme applicative che semplifichino la vita del programmatore
  - ⇨queste piattaforme forniscono vari servizi (es: decodifica della richiesta, produzione della risposta, gestione della sessione ecc.)
  - ⇨un approccio già visto: i DBMS

Riassumendo

- Introduzione
- Logica Applicativa
- Richieste e Risposte
- Stato delle Sessioni
  - ⇨Cookies
- Contesto dell'Applicazione
- Riassumiamo i Problemi

Termini della Licenza