Schema Client Server

Web server
– Rende disponibili un set di pagine
– Usa porta 80

Web client
– Chiamato *browser*
– Crea connessioni TCP al server
– Manda le richieste per gli oggetti

Il protocollo è *HyperText Transfer Protocol HTTP)*

Il Browser

Controllore principale
– Riceve le richieste dall'utente
– Chiama il client e l'interprete
Client
– Possono essere uno o più all'interno del Browser
– Usa la rete per caricare gli oggetti
Interprete
– Possono essere uno o più
– Visualizzano gli oggetti

NB: Il Browser può avere molte componenti

Altri protocolli supportati

File transfer service: Protocollo FTP
Esempio
ftp://ftp.cs.purdue.edu/pub/comer/netbook/client.c

Il Caching nei Browsers

Cache per acceduti di recente
– Pagine HTML
– Immagini

Oggetti caricati normalmente in cache
Possono essere sovrascritti dall' utente
HTTP può verificare la data prima di caricarne una copia nuova

Tipi di pagine Web

Statiche
– Memorizzate in file
– Invariate

Dinamiche
– Create dal server
– Create su richiesta
– Output di un programma
– Usano la tecnologia *Common Gateway Interface (CGI)*

Attive
– Eseguite dal client
– Sono un programma
– Possono interagire con l' utente
– Usano Java

Tipi di documenti Web (riepilogo)

I documenti Web possono essere raggruppati in 3 categorie in base a quando ne cambiano i contenuti.
Le informazioni in un documento ***statico*** *rimangono invariate finchè non viene modificato dall'autore.*
In quelli ***dinamici*** *può cambiare ogni volta che viene richiesto al server.*
In quelli ***attivi*** *può cambiare quando il documento è stato caricato dal browser.*

CGI

URL indica
– Il Web server
– Il programma CGI su quel server
– Argomenti del programma

Web server
– Usa il TCP per comunicare
– Accetta richieste HTTP dal client
– Esegue la CGI richiesta
– Restituisce l'output al client

Programma CGI
– Viene eseguito
– Spesso è scritto in un linguaggio di scripting (Perl, PHP, ASP, etc..)
– Produce output
– L'Output inizia con gli Header

Gli Header CGI

– Finiscono con una linea vuota
– Identificano
  – Codifica usata
  – Tipo di documento
– Formato
  – *Keyword : informazione*

Esempi
HTML document header
Content Type: text/html

_
Text document header
Content Type: text/plain

_
Redirection header
Location: /over_here/item4

Esempio di script CGI

```
#!/bin/sh
#
# CGI script che stampa la data e l'  ora
#
# Output: document header seguito da una linea bianca
echo Content-type: text/plain
echo
# Output: la data
echo Questo documento è stato creato il ‘date‘
#
```

Questa CGI:
Genera un documento che contiene 3 linee di testo:
– Header
– Linea vuota
– La data di creazione del documento

Informazioni sullo stato

Ciclo di vita del programma CGI

– La CGI è richiamata dal server
– Il programma finisce dopo aver generato l' output

Per mantenere dati persistenti:

– Scrivere su file nel disco
– Leggere file da disco

Esempio di script CGI con informazioni sullo stato

```
#!/bin/sh
FILE=ipaddrs
echo Content-type: text/plain
echo
# controlla se l' indirizzo IP del tuo browser compare nel nostro file
if grep -s $REMOTE_ADDR $FILE >/dev/null 2>&1
then
echo Computer $REMOTE_ADDR ha visitato questa URL
else
# Aggiunge l' indirizzo IP del browser al file
echo $REMOTE_ADDR >> $FILE
echo Prima volta che si è contattati da questo computer $REMOTE_ADDR
fi
```

L' indirizzo IP del client è in una **variabile di ambiente**

Controlla se è presente nel file

Risponde al client

Gestione informazioni nelle URL

- URL può contenere argomenti
- Il punto interrogativo separa la CGI dagli argomenti
- Gli argomenti sono le informazioni

Esempio di codifica di parametri:

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
char *data;
long m,n;
printf("%s%c%c\n",
"Content-Type:text/html;charset=iso-
8859-1",13,10);
printf("<TITLE>Risultati del
prodotto</TITLE>\n");
printf("<H3>Risultati del prodotto
</H3>\n");
data = getenv("QUERY_STRING");
if(data == NULL)
  printf("<P>Errore i dati sono
sbagliati.");
else
if(sscanf(data,"m=%ld&n=%ld",&m,&n)!
=2)
  printf("<P>Errore! I dati devono
essere numerici.");
else
  printf("<P>Il prodotto di %ld e
%ld uguale %ld.",m,n,m*n);
return 0;
}
```

```
<FORM ACTION="http://localhost/cgi-tari/moltiplica">
<P>specifica gli operandi:
<INPUT NAME="m" SIZE="5">
<INPUT NAME="n" SIZE="5"><BR>
<INPUT TYPE="SUBMIT" VALUE="Moltiplica">
</FORM>
```


URL con valori generati

Quando si genera un documento dinamico il programma può conservare informazioni sullo stato utilizzandoli come parametri nelle URL.

Gli argomenti passati ad un programma attraverso la URL, possono quindi essere utilizzate per permettergli di conservare informazioni sullo stato da una chiamata all'altra.

Aggiornamento continuo delle informazioni

Necessario per:
- Animazioni
- Aggiornamenti rapidi (quotazioni di borsa)

Implementato con 2 meccanismi
- Server push
- Documenti attivi

Tecnologia Server Push (non più utilizzata)

Il Client si collega al Server
Il Server manda gli aggiornamenti

Scomoda

Tecnologia Pagine attive

Server
– Manda programmi al client

Client
– Esegue localmente i programmi

Programmi
– Gestiscono la rappresentazione
– Interagisce con l'utente

Visualizzazione delle pagine attive

Obbiettivi
– Multipiattaforma
– Esecuzione efficiente
– Trasmissione dei dati veloce

Conseguenze
– Rappresentazione compatta
– Esecuzione interpretata

Documento attivo in formato sorgente

Compilatore

Documento attivo in forma eseguibile

Server

Client

Esecuzione sul Client della forma binaria del documento attivo

Il compilatore produce un codice binario multipiattaforma
Il browser interpreta il codice binario

Java Technology

Sviluppata da Sun Microsystems

Usata per
- Applicazioni tradizionali
- Documenti attivi *(applet*s)

Comprende
- Linguaggio di programmazione
- Parte Run-Time
- Libreria delle classi

Caratteristiche del linguaggio Java

- Alto livello
- General Purpose
- Simile al C++
- Object Oriented
- Dinamico
- Strettamente tipizzato
- Type checking statico
- Concorrente

Java Run-Time Environment caratteristiche

- Esecuzione interpretata
- Garbage Collection automatica
- Esecuzione Multi-threaded
- Internet Access
- Supporto grafico

Java Library

Classi per

- Manipolazione grafica
- Low-Level Network I/O
- Interazione con Web Server
- Run-Time System Access
- File I/O
- Conventional Data Structures
- Event Capture
- Exception Handling

Scelte di interfaccie grafiche

Java include un graphics toolkit esteso che consiste di un supporto run-time per grafica così come per le interfaccie software.
Il toolkit permette al programmatore di scegliere un'interfaccia di alto livello che gestisce i dettagli oppure la possibilità di gestire i dettagli direttamente attraverso l'applet.

Esempio di Java Applet

Finestra con 2 oggetti

- area di testo
- Bottone

Cambia il testo quando viene schiacciato il bottone

Esempio di codice per Applet

```
import java.applet.*;
import java.awt.*;
public class clickcount extends Applet {
int count;
TextField f;
public void init() {
count = 0;
add(new Button("Click Here"));
f = new TextField("The button has not been
clicked at all.");
f.setEditable(false);
add(f);
}
public boolean action(Event e, Object arg) {
if (((Button) e.target).getLabel() == "Click Here") {
count += 1;
f.setText("The button has been clicked " + count
+ " times.");
}
return true;
}
}
```


Chiamata dell'Applet

Disponibile in HTML
Usa *applet* tag

Specifica
– *Codebase* (macchina e path)
– *Code* (specifica classe da usare)

Esempio

```
<applet codebase="www.nonexist.com/pth"
code="bbb.class">
```

Funzionalità di Java

Interfaccia HTML
– Controlla display
– Interagisce con l'utente

Interfaccia HTTP
– Accede pagine Web remote
– Chiama altre applets

Exceptions
– Indica circostanze impreviste
– Possono essere catturate e gestite