Basi di dati e WWW

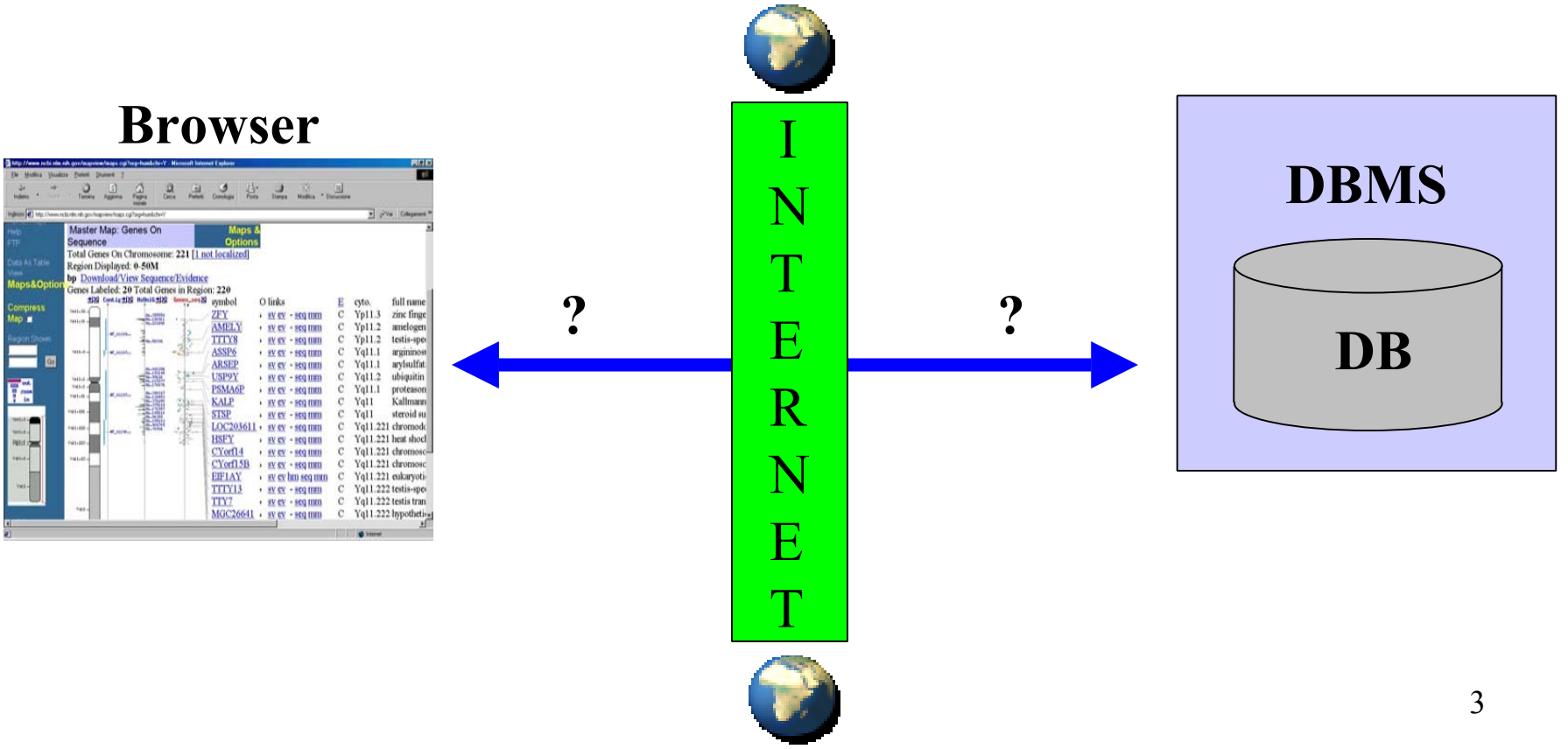
WIS
Web Information Systems

Obiettivi

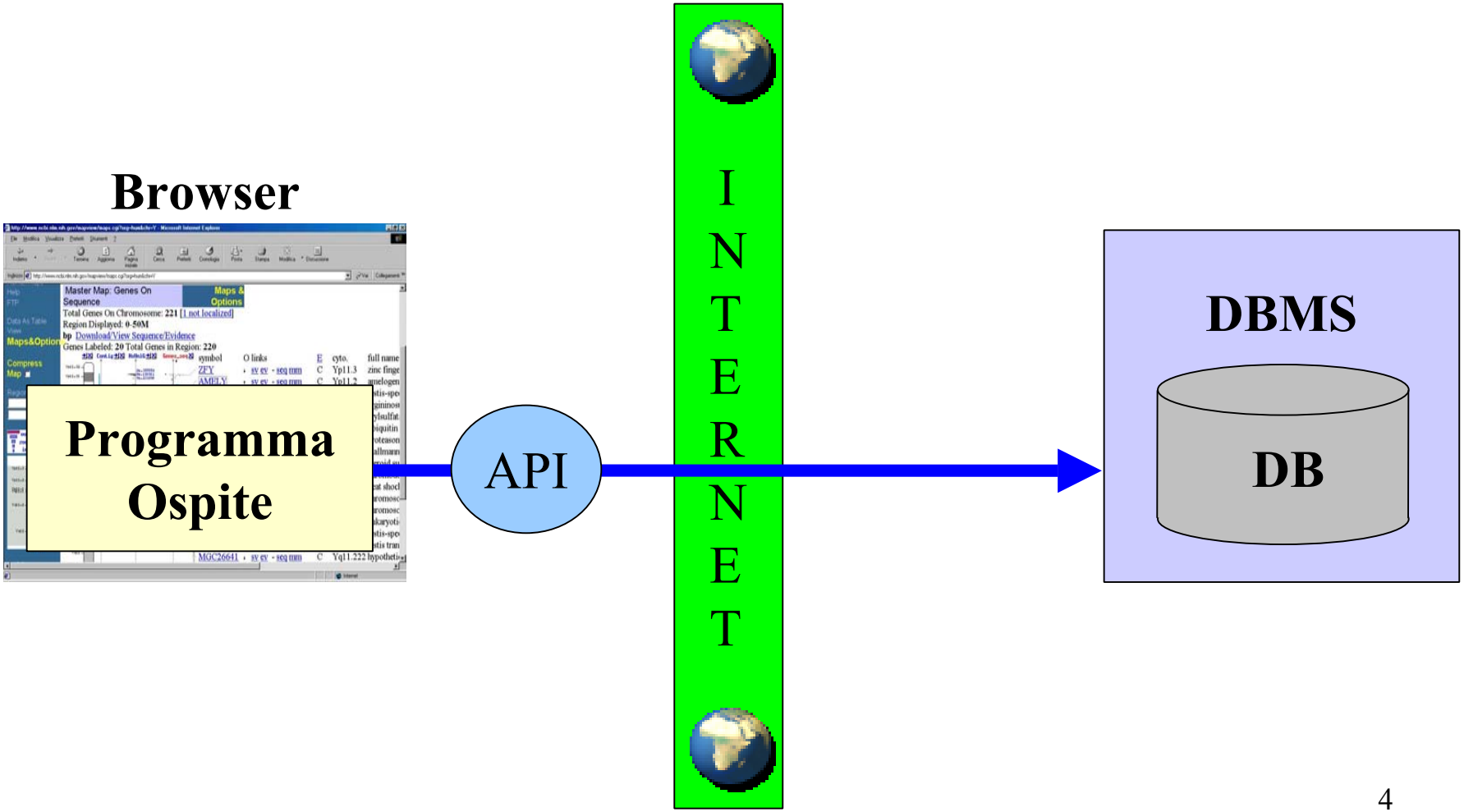
- Con l'avvento di Internet e' nata la necessita' di sviluppare **sistemi informatici accessibili via Web**.
- Gli utenti che utilizzano la rete hanno solitamente a disposizione un browser.
- Vedremo diversi modi per **estrarre informazioni** da una base di dati **attraverso un browser**.

Connessione Browser/DBMS via Web

- Come attuare la comunicazione?



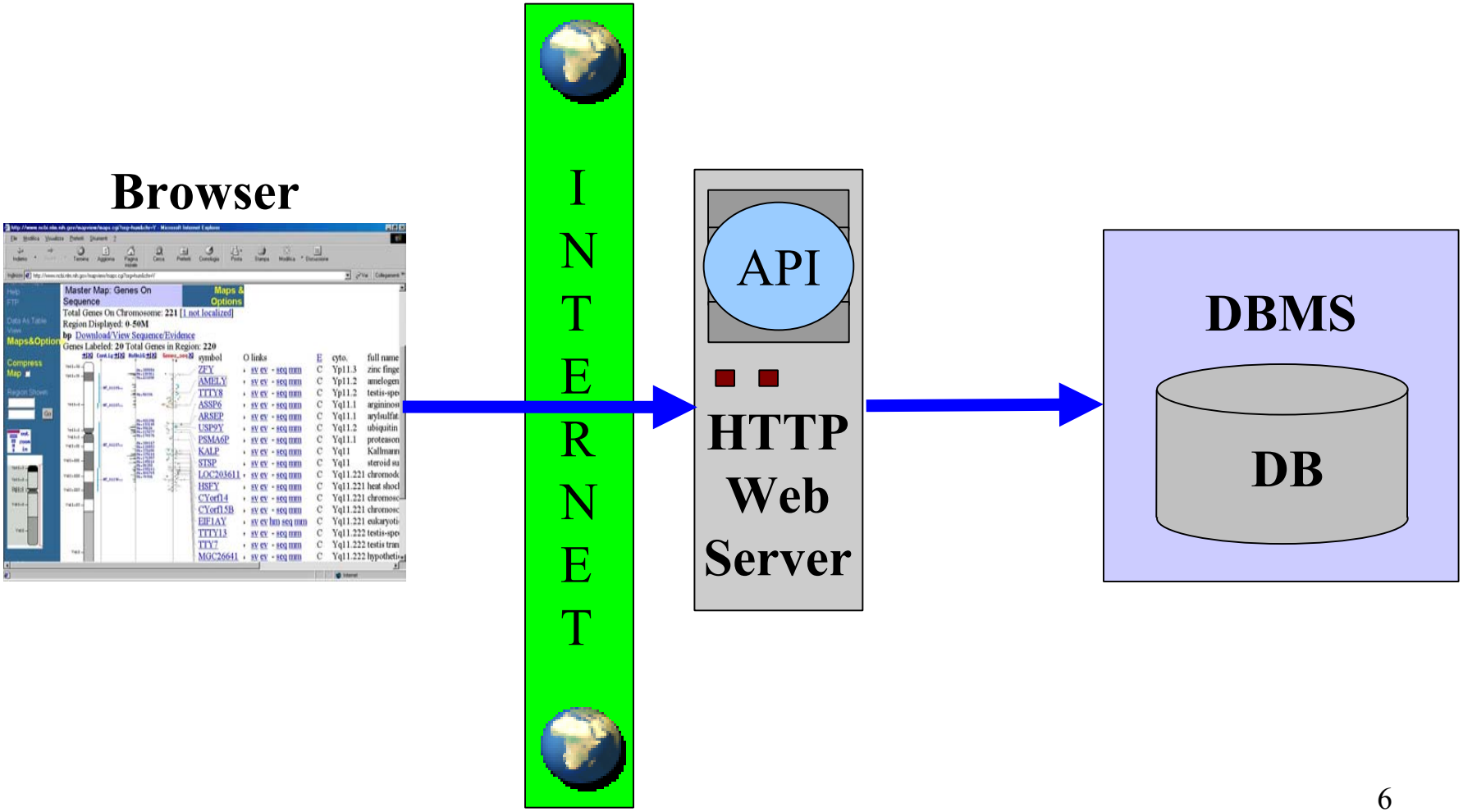
Scenario 1: Client Side



Scenario 1: Client Side

- Il browser utilizza un programma che permette la connessione alla base di dati.
- Questa soluzione e' consigliabile solo quando:
  - Esiste un controllo sui client (ad esempio la possibilita' di installare programmi adeguati su tutti i computer che accedono alla BD), oppure
  - il carico di lavoro dell'applicazione e' di piccola entita'.

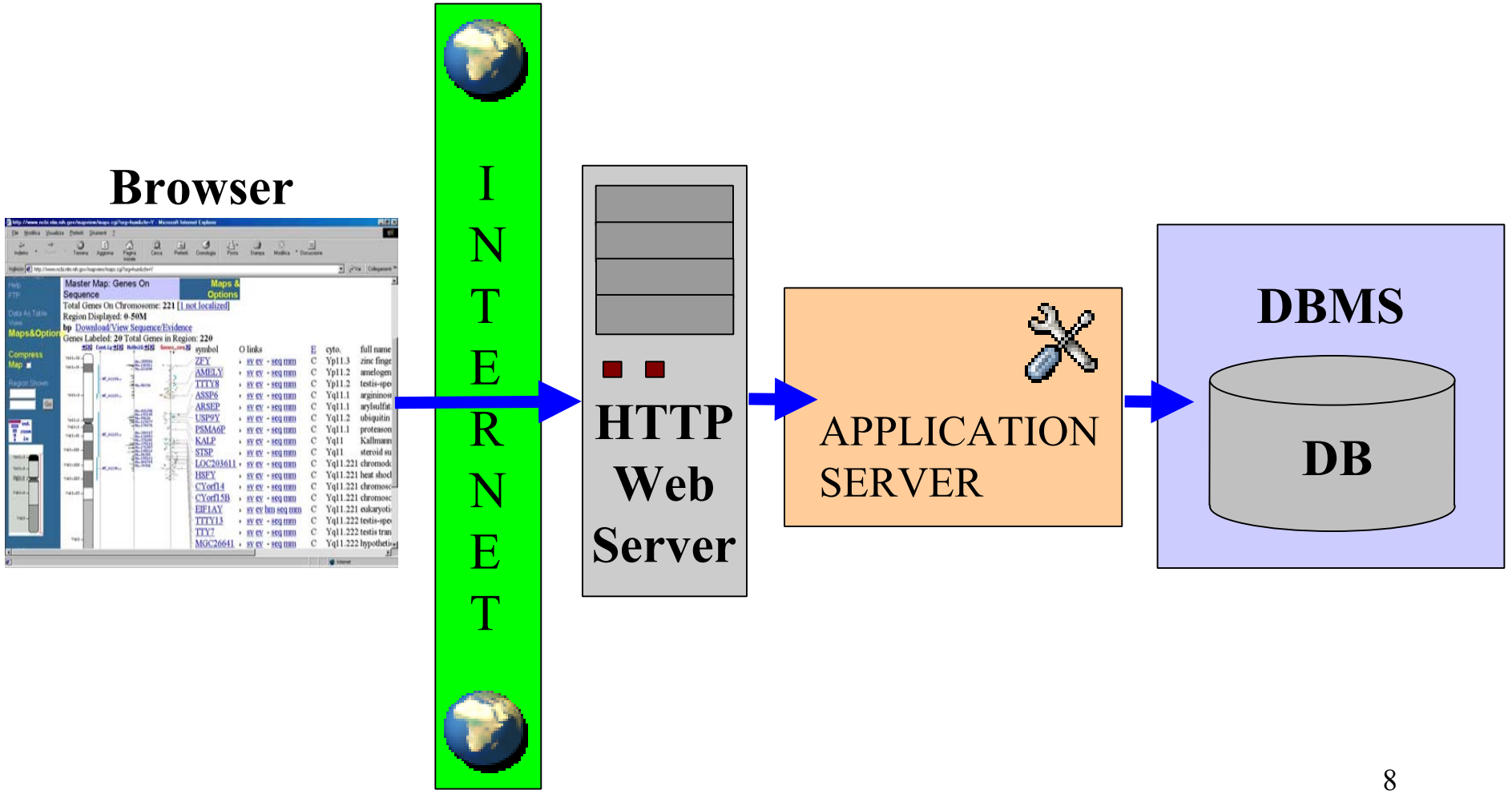
Scenario 2: Server Side (ISAPI/NSAPI)



Scenario 2: Server Side (ISAPI/NSAPI)

- E' possibile estendere il server Web, in modo che esso comunichi direttamente con il DBMS.
- Non esiste uno standard.
- Le estensioni sono difficili da programmare (gestione della concorrenza).
- I moduli "girano" insieme al Web server. Se i primi hanno dei problemi, il secondo ne puo' risentire (devono essere sicuri e stabili).

Scenario 2: Server Side (Application Server)



Scenario 2: Server Side (Application Server)

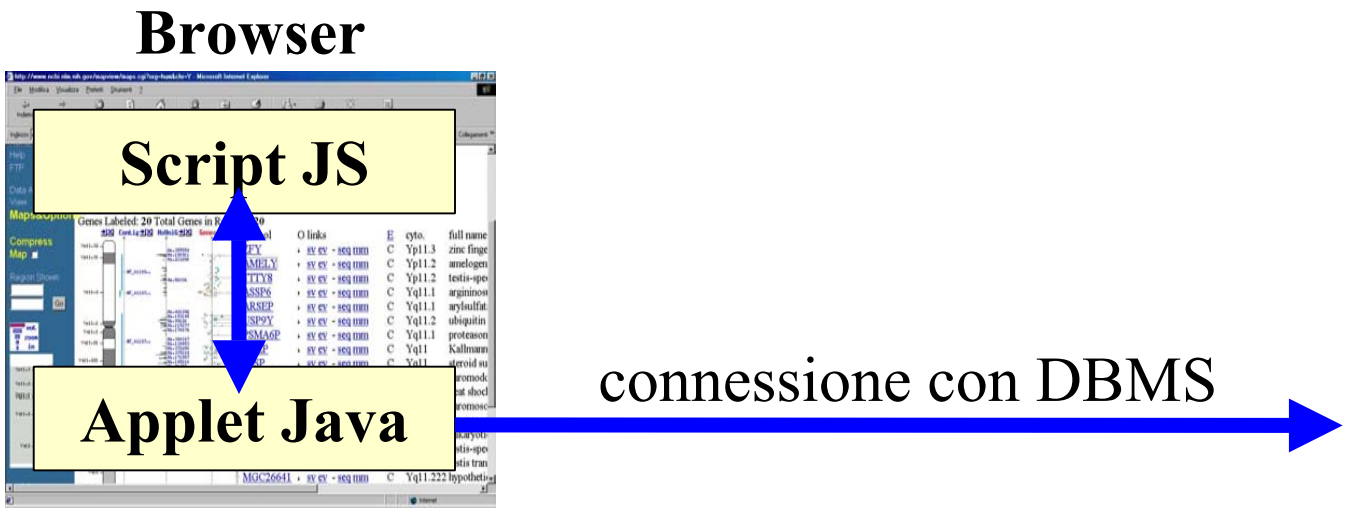
- La parte dell'applicazione che comunica con il database viene implementata al di fuori del server Web, con il quale comunica.
- Questo rende il sistema scalabile e sicuro.
- In teoria, questa soluzione puo' essere meno efficiente dell'utilizzo di ISAPI, in cui la DLL con l'estensione viene caricata una volta per tutte.
- Esistono comunque implementazioni molto efficienti di questa strategia, che vedremo.

Basi di dati e WWW

Scenario 1:
Client Side

JavaScript e Java

- Una Applet Java puo' comunicare con un DBMS, direttamente o tramite altri programmi (ad esempio una servlet).
- Uno script JavaScript puo' accedere ai metodi dell'applet tramite DOM.

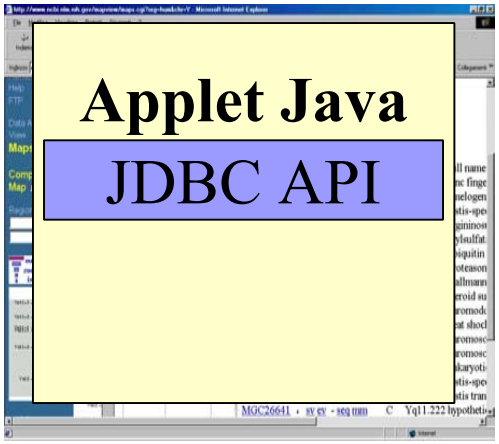


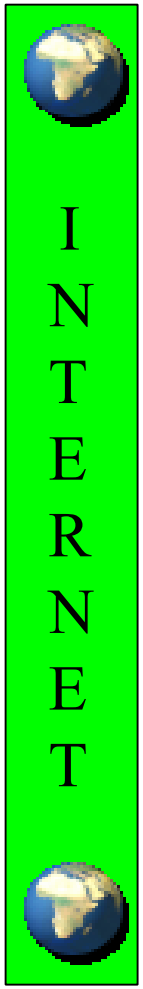
Applet Java

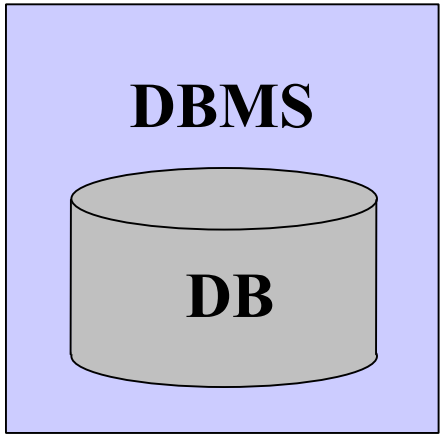
- E' possibile far comunicare Java e DBMS tramite un'API (Application Programming Interface) chiamata JDBC.
- JDBC e' attualmente alle versioni 3.0 (gia' disponibile) e 4.0 (in preparazione).
- JDBC serve per connettersi e interrogare basi di dati indipendentemente dal DBMS e dalla piattaforma utilizzata dal client.

Esempio di utilizzo di JDBC

Browser

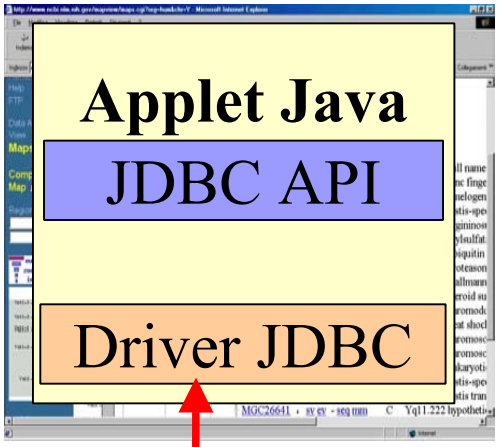




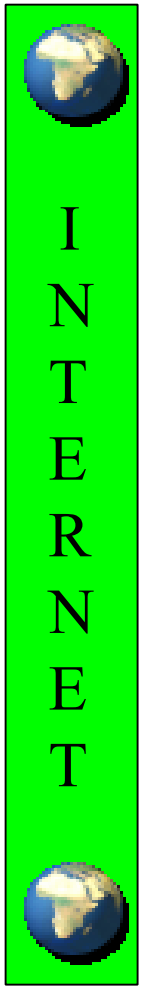


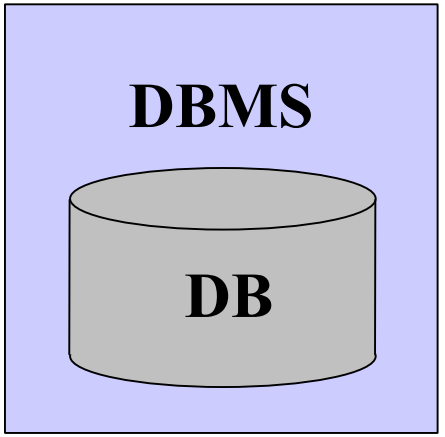
Class.forName("*COM.cloudscape.core.RmiJdbcDriver*");

Browser

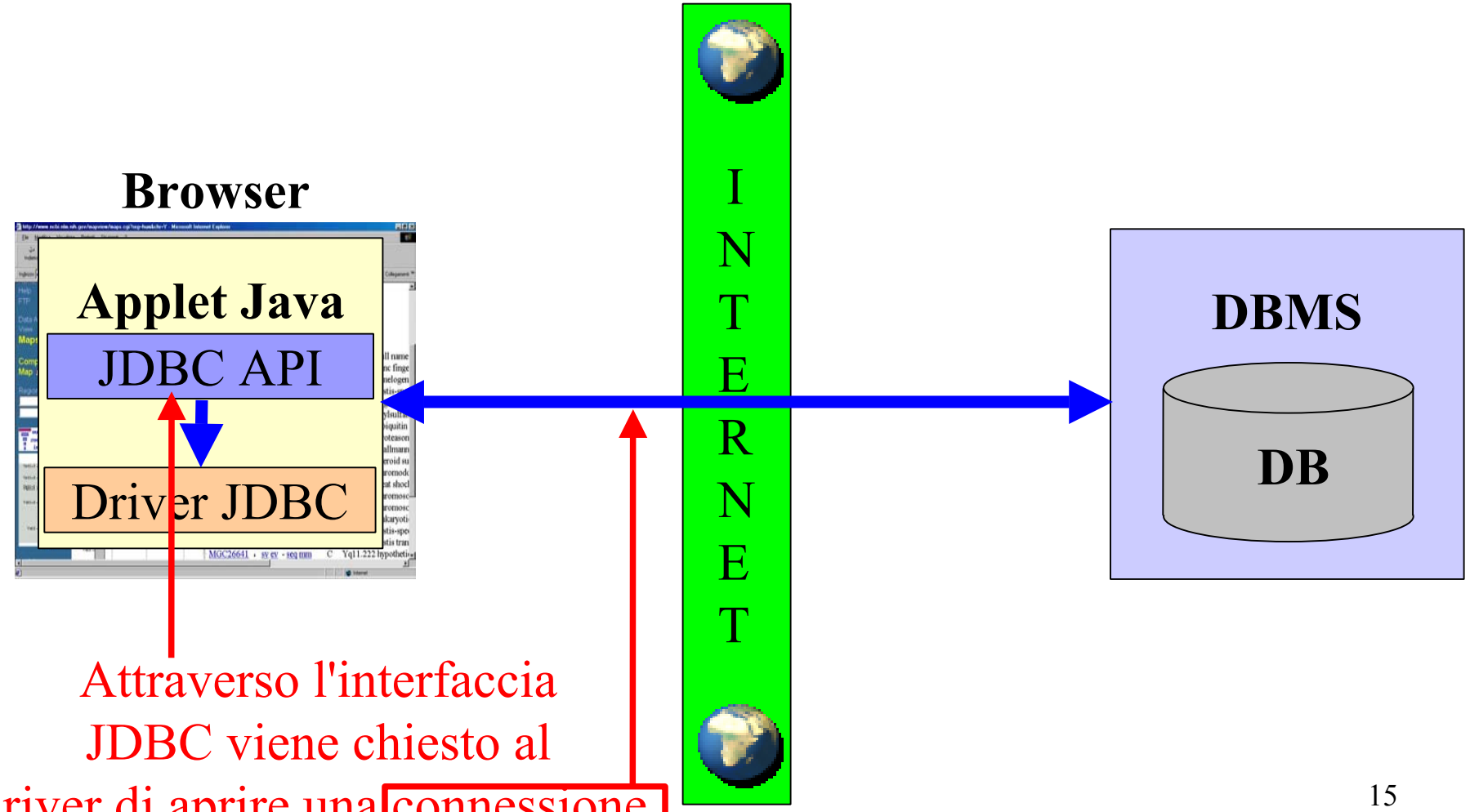


Il driver specifico per il DBMS utilizzato viene caricato.

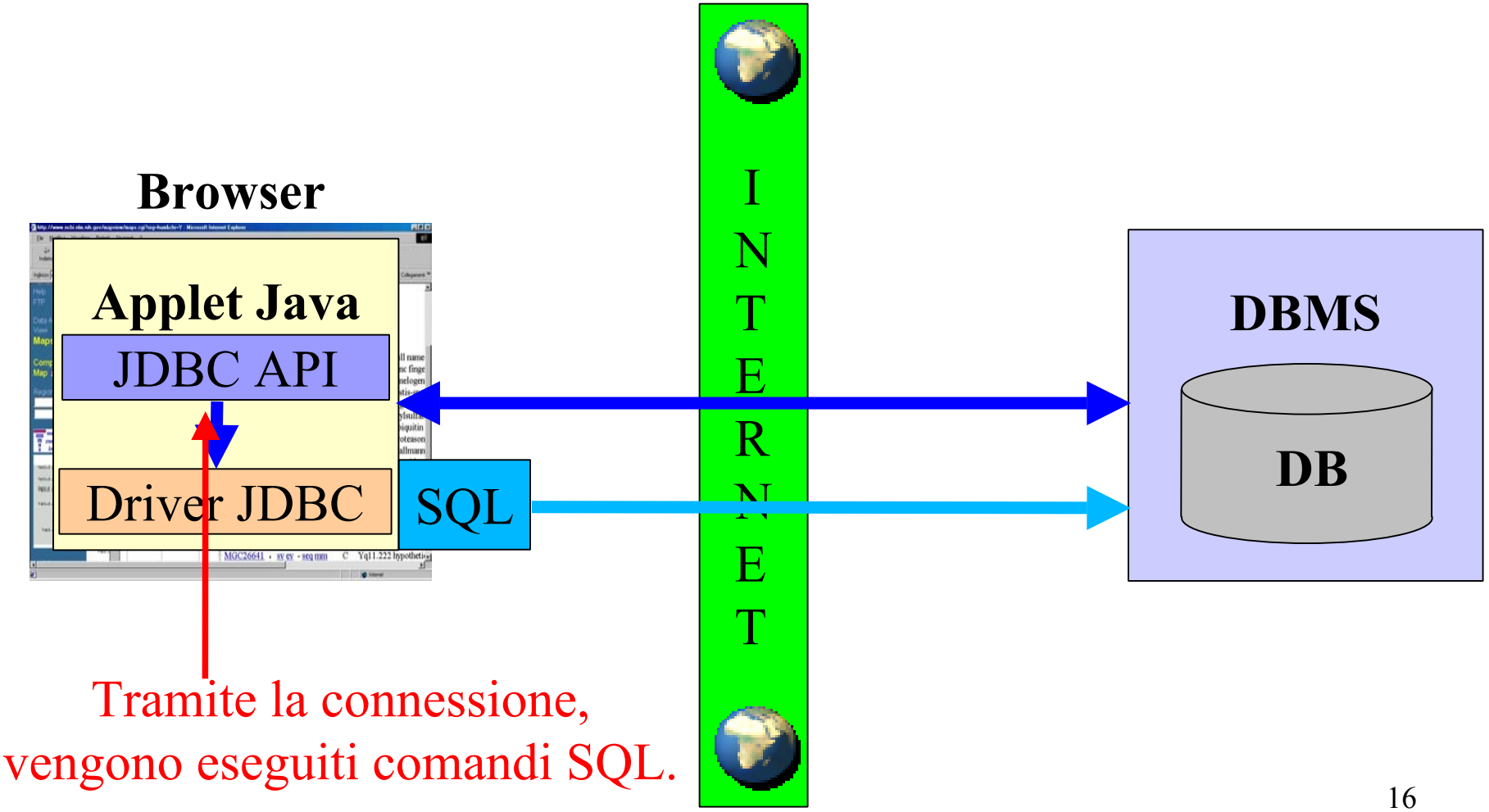




Connection con = DriverManager.getConnection(url,"","");

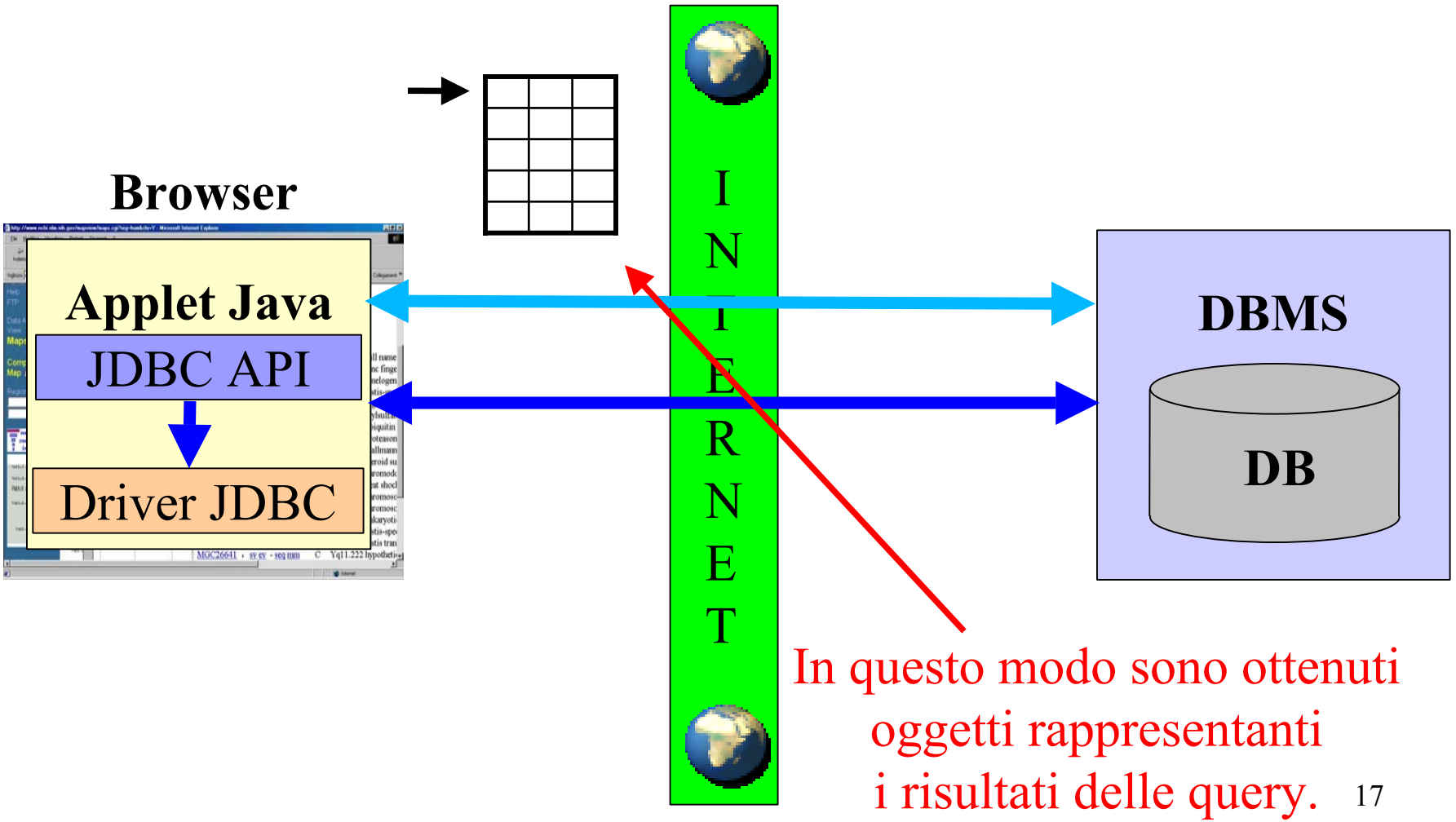


Statement stmt = con.createStatement();



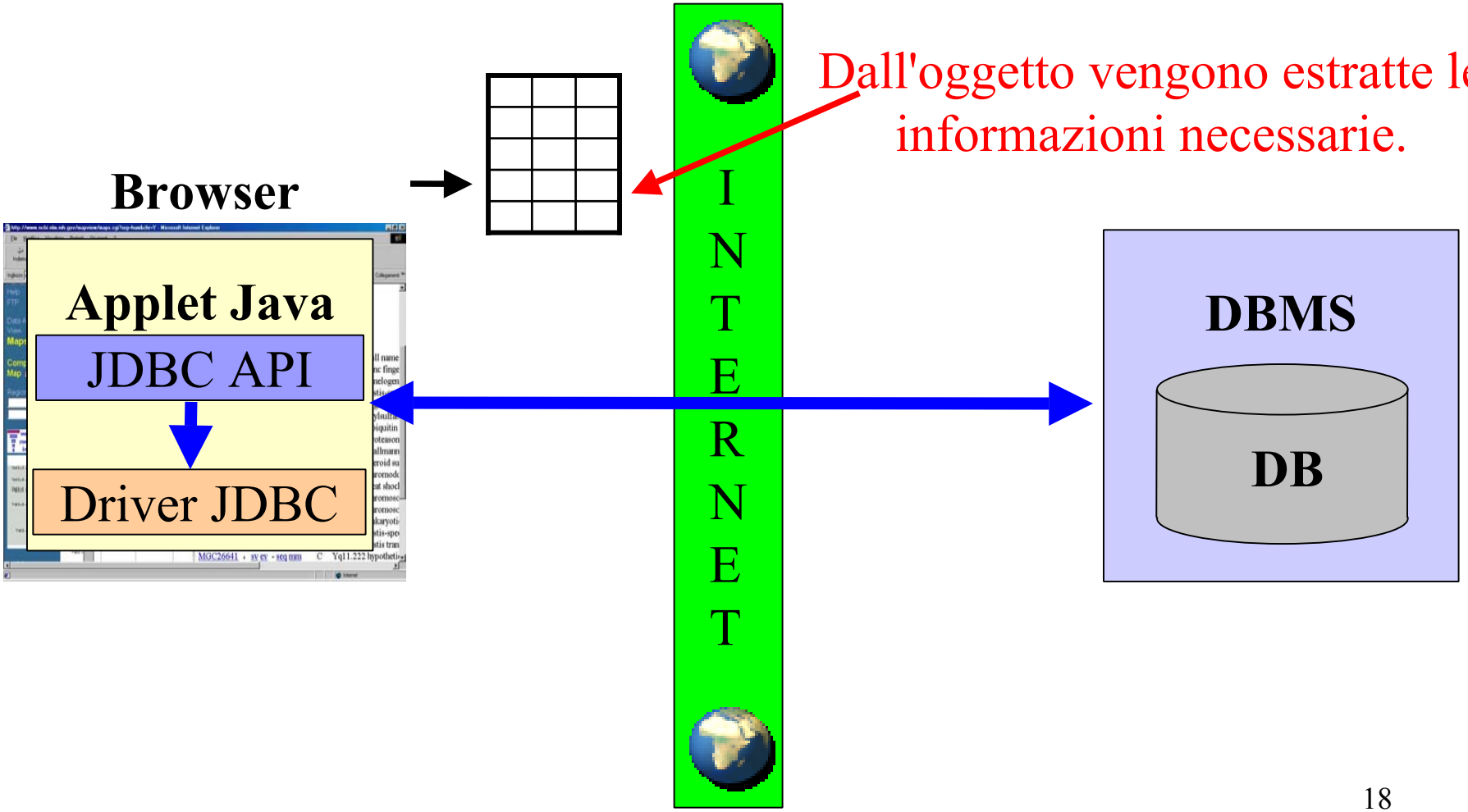
Tramite la connessione,
vengono eseguiti comandi SQL.

ResultSet rs = stmt.executeQuery("SELECT a FROM T");

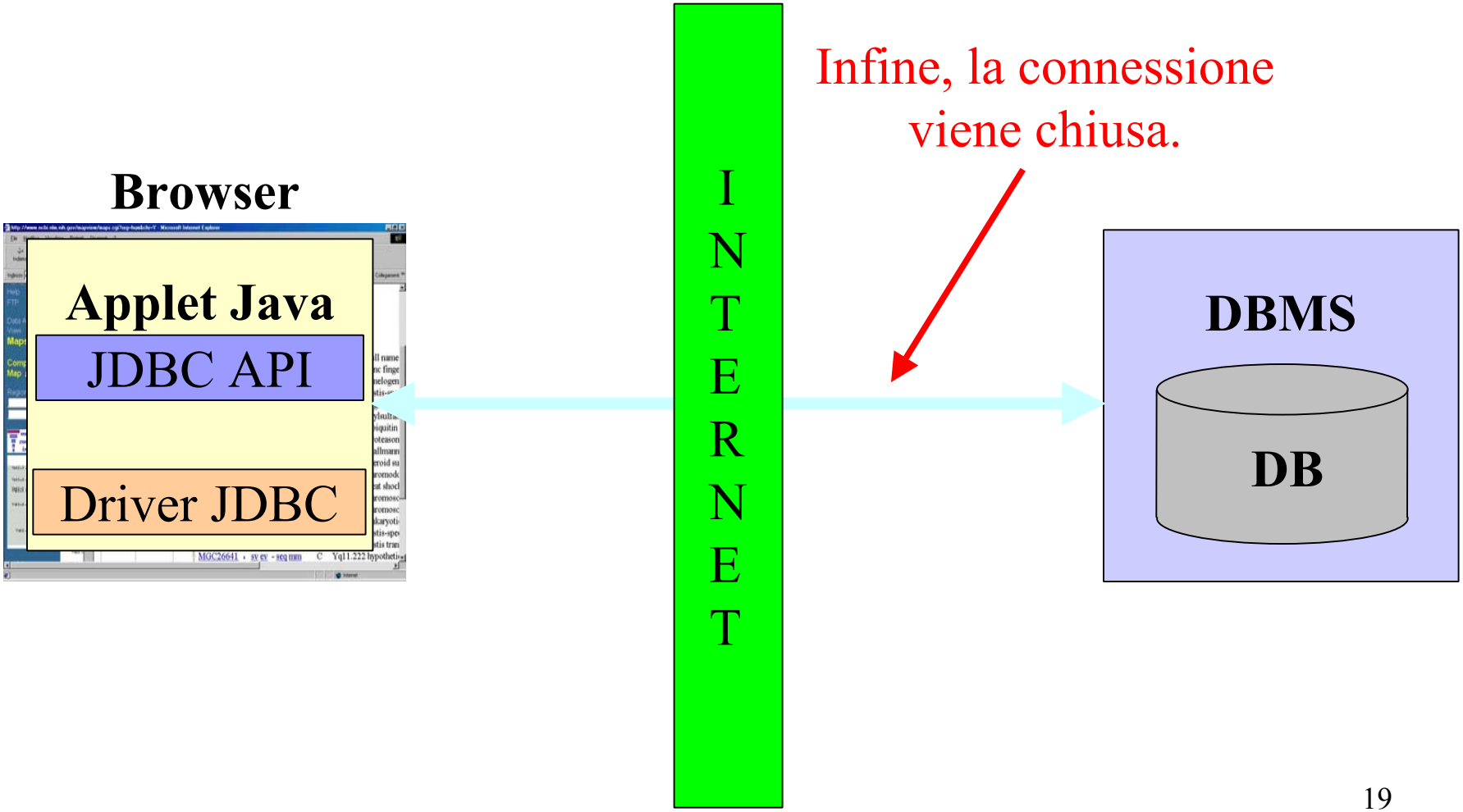


In questo modo sono ottenuti oggetti rappresentanti i risultati delle query.

```
while (rs.next()) { System.out.println(rs.getString(1)); }
```

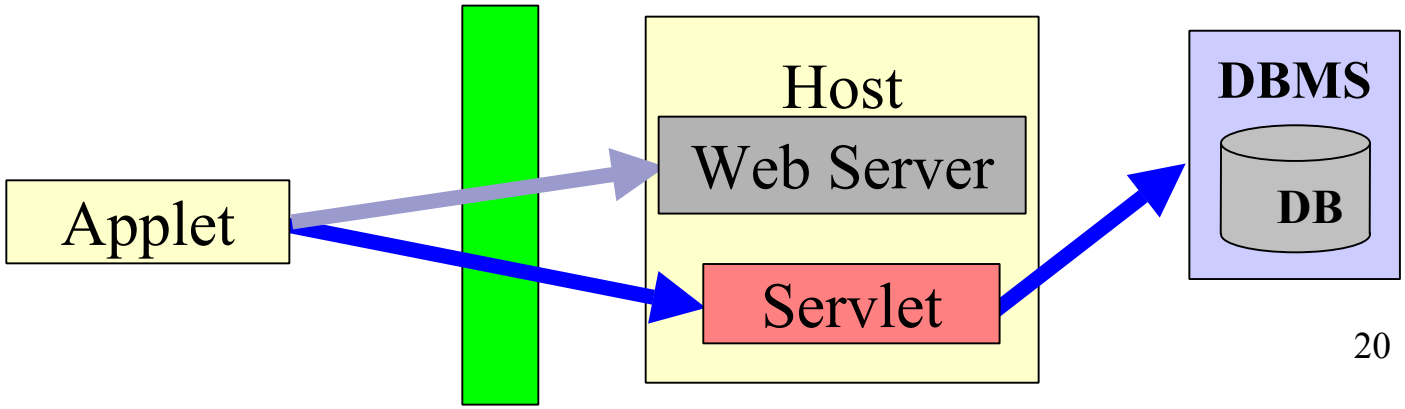


```
finally { if (con != null) con.close();}
```

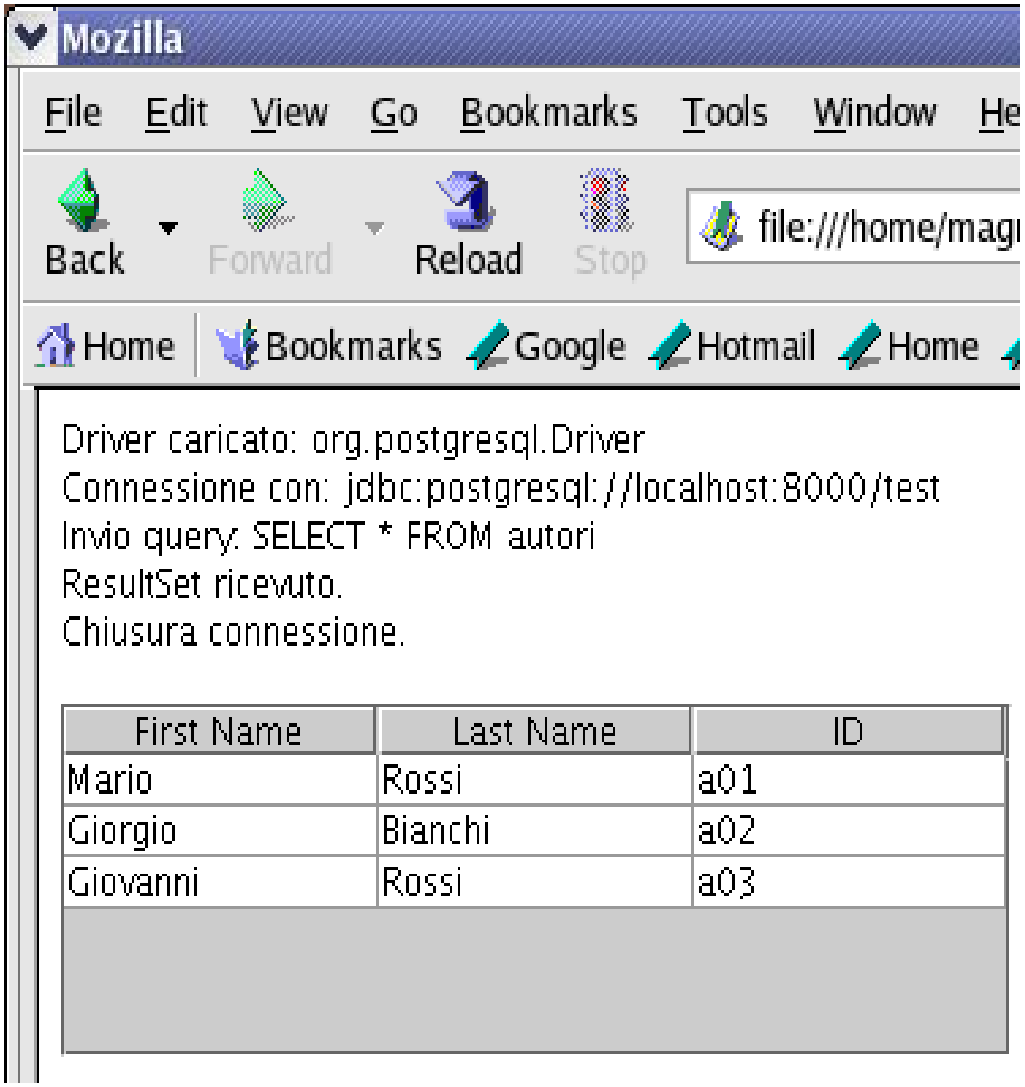


Connessione verso altri host

- ATTENZIONE. Un'applet (non autenticata e su internet) puo' connettersi solo con l'host da cui e' stata scaricata.
- Nel caso di un DBMS installato su una macchina diversa, e' necessario un middleware sull'host di origine.



Snapshot: Applet connessa a Postgresql



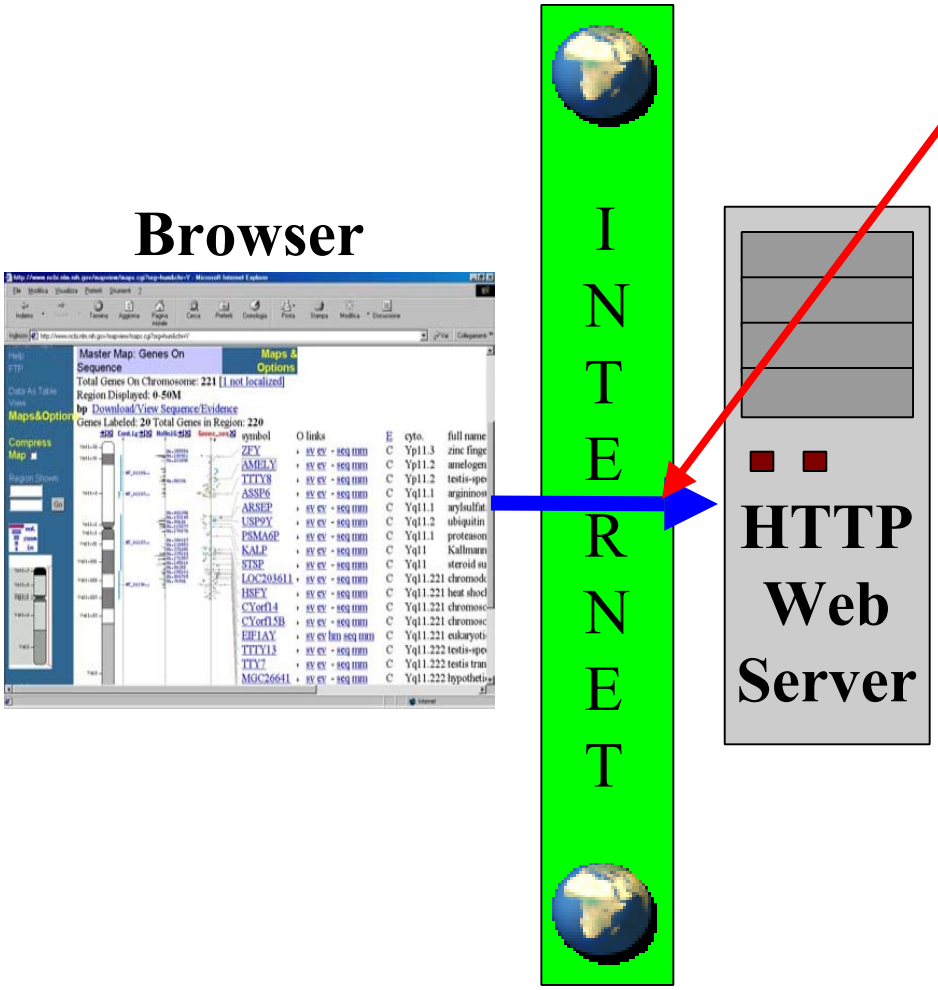
Basi di dati e WWW

Scenario 2:
Server Side (con Application Server)

Server Side in generale

- Un interfaccia browser/DB lato server puo' essere realizzata tramite metodi e linguaggi diversi (ricordiamo CGI, Servlet, pagine PHP, ASP, JSP).
- Tutti questi metodi agiscono in generale nello stesso modo.

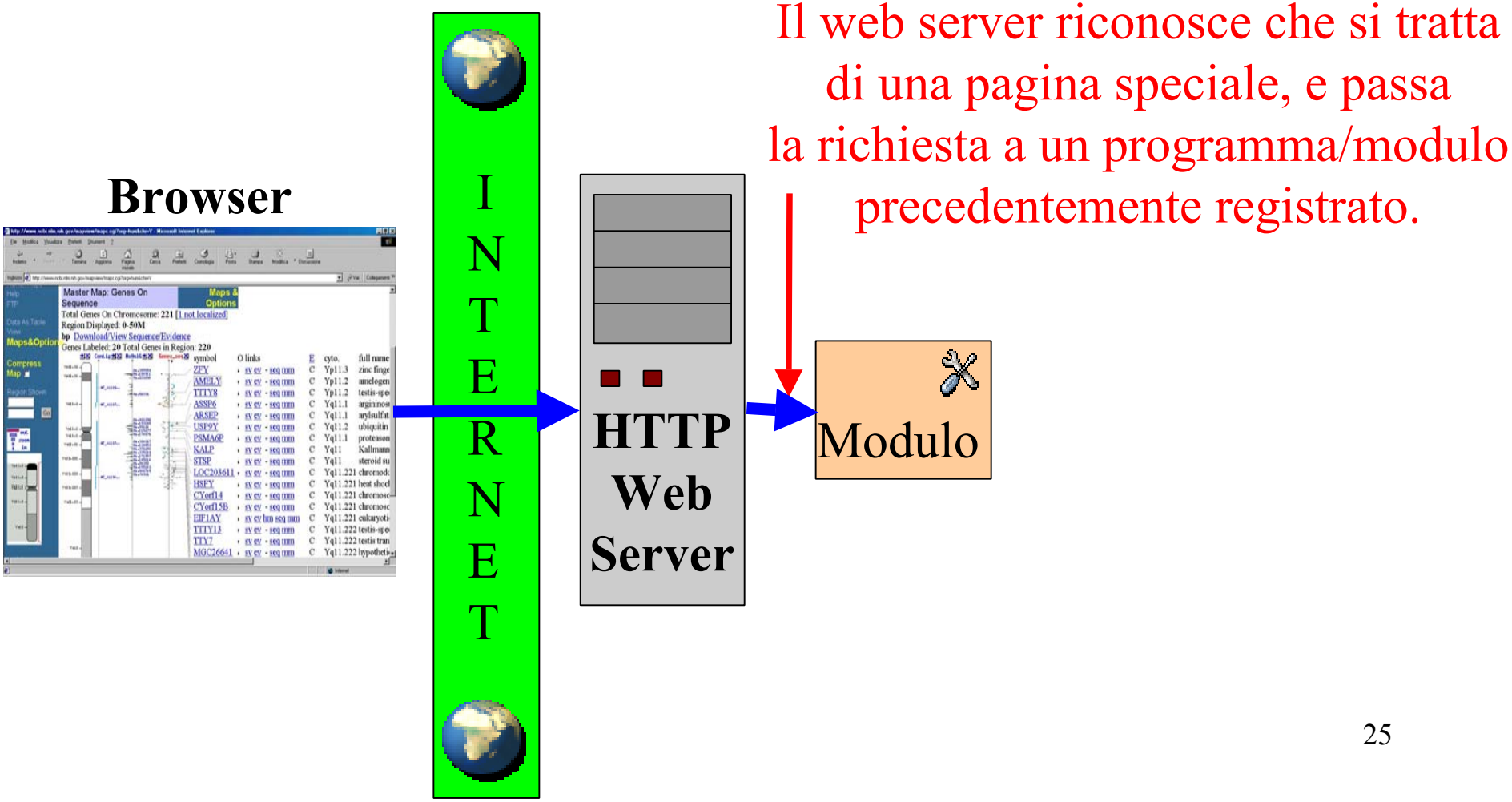
Richiesta del browser



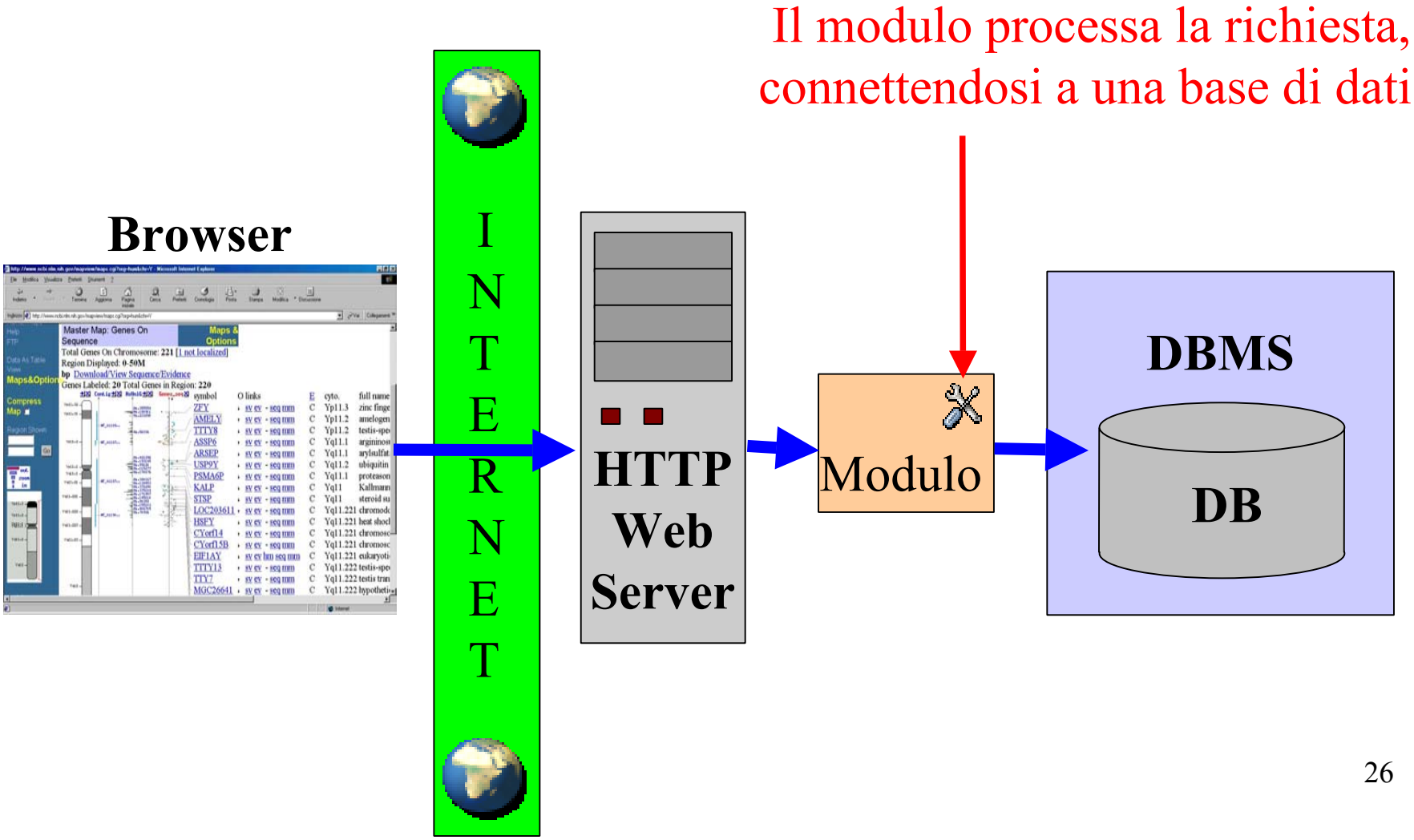
Viene richiesta una pagina che necessita di una connessione a una base di dati.

Queste pagine sono caratterizzate da un URL speciale, che puo' essere riconosciuto dal web server. Ad esempio, http://*host*/cgi-bin/

Forward a un modulo specifico

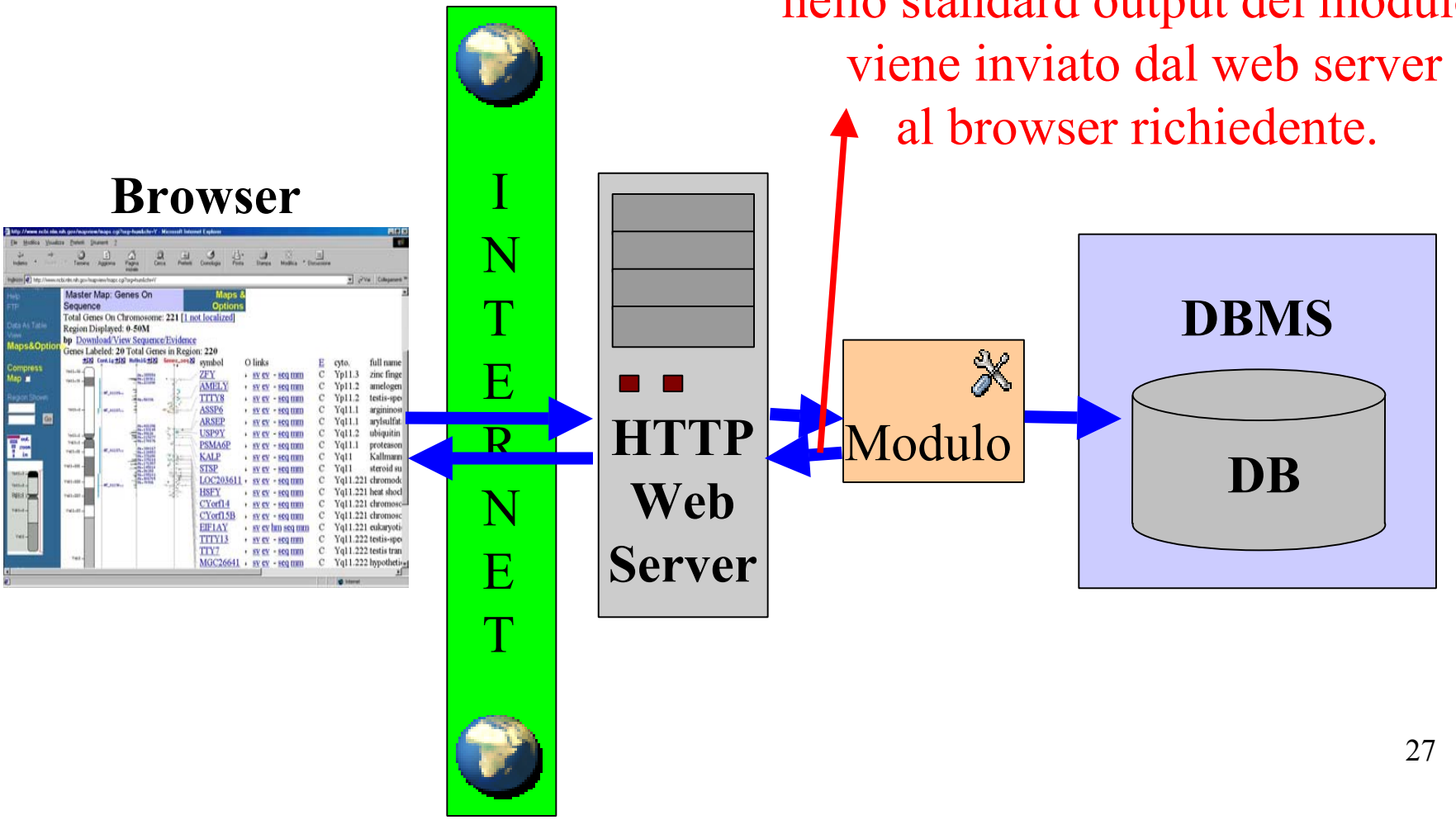


Elaborazione della richiesta



Restituzione del risultato

Il risultato, solitamente scritto nello standard output del modulo, viene inviato dal web server al browser richiedente.



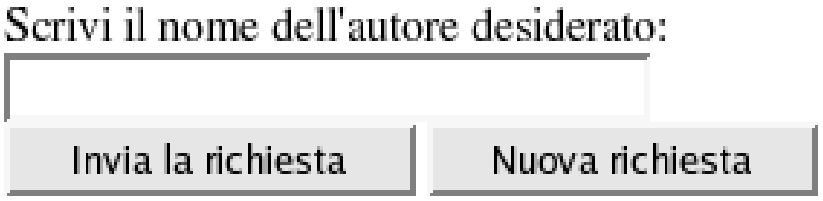
CGI

- Un esempio ben noto e' quello delle CGI (Common Gateway Interface).
- Una CGI viene implementata tramite un programma esterno al Web server, scritto tipicamente in Perl, residente solitamente in una directory virtuale */cgi-bin*.

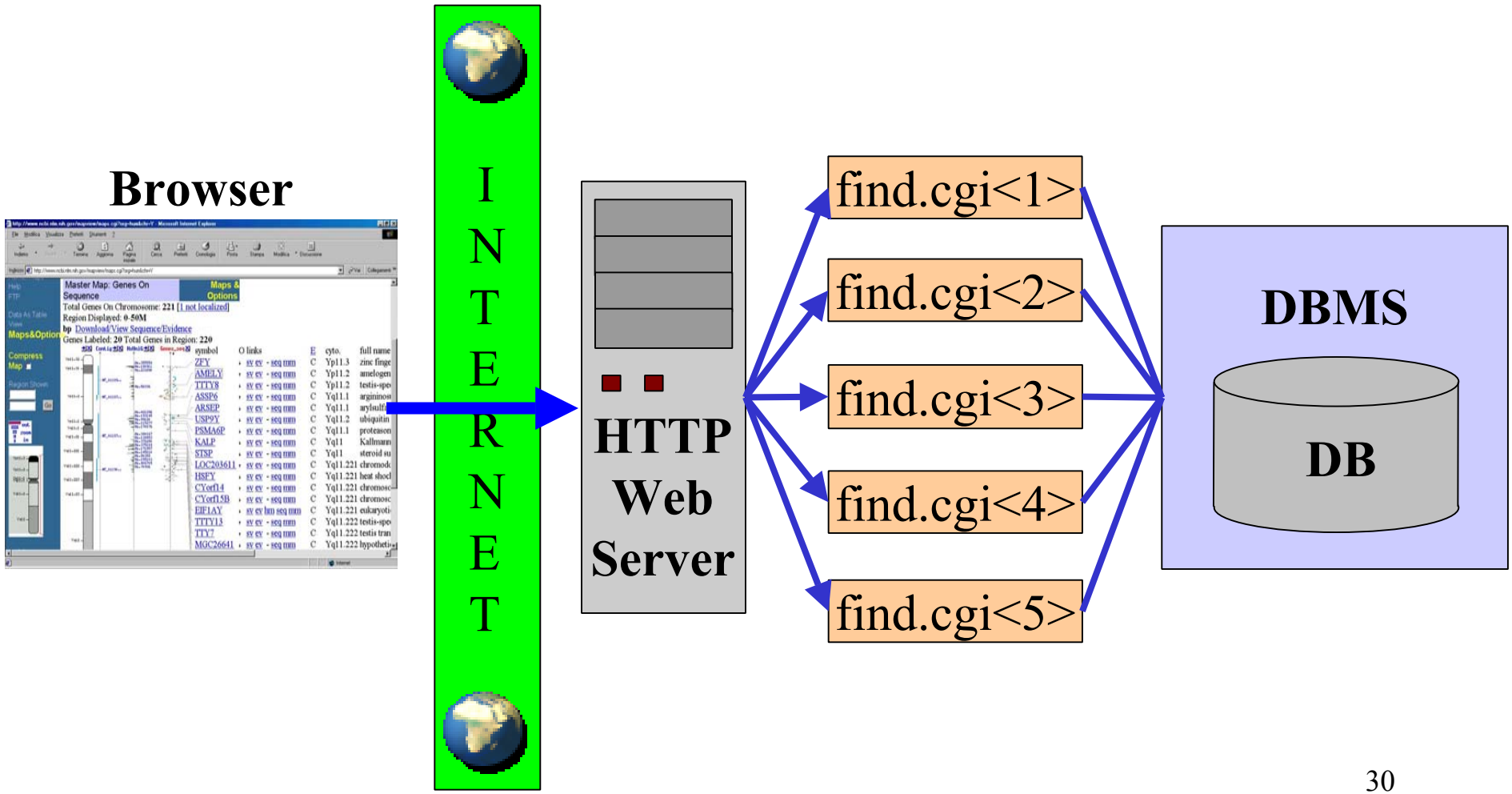
CGI

- Esempio di invocazione di una CGI.

```
<HTML><BODY>
<FORM action="cgi-bin/find.cgi" method="post">
    Scrivi il nome dell'autore desiderato:
    <INPUT type="text" name="authorName" size="30"
   maxlength="50">
    <INPUT type="submit" value="Invia la richiesta">
    <INPUT type="reset" value="Nuova richiesta">
</FORM>
</BODY></HTML>
```



CGI



Problemi delle CGI

- Per ogni richiesta viene inizializzato un nuovo processo. Le performance di una CGI sono il vero punto debole di questa soluzione.
- Una CGI scritta male presenta rischi di sicurezza.
- E' necessario un interprete esterno (nel caso di CGI scritte in perl).
- Alcune CGI non sono portabili (ad esempio se scritte in C).

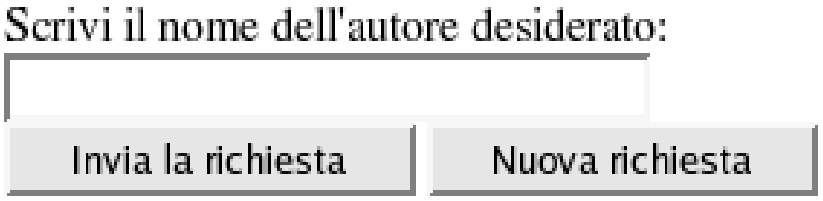
Java Servlet

- Una soluzione alle basse performance delle CGI sono le Servlet, applicazioni Java lato-server.
- Una servlet viene inizializzata solo una volta, poi resta disponibile (al prezzo di un utilizzo costante di risorse del sistema).
- Nel caso di piu' richieste contemporanee, non vengono creati nuovi processi, ma esse vengono gestite tramite *thread*.

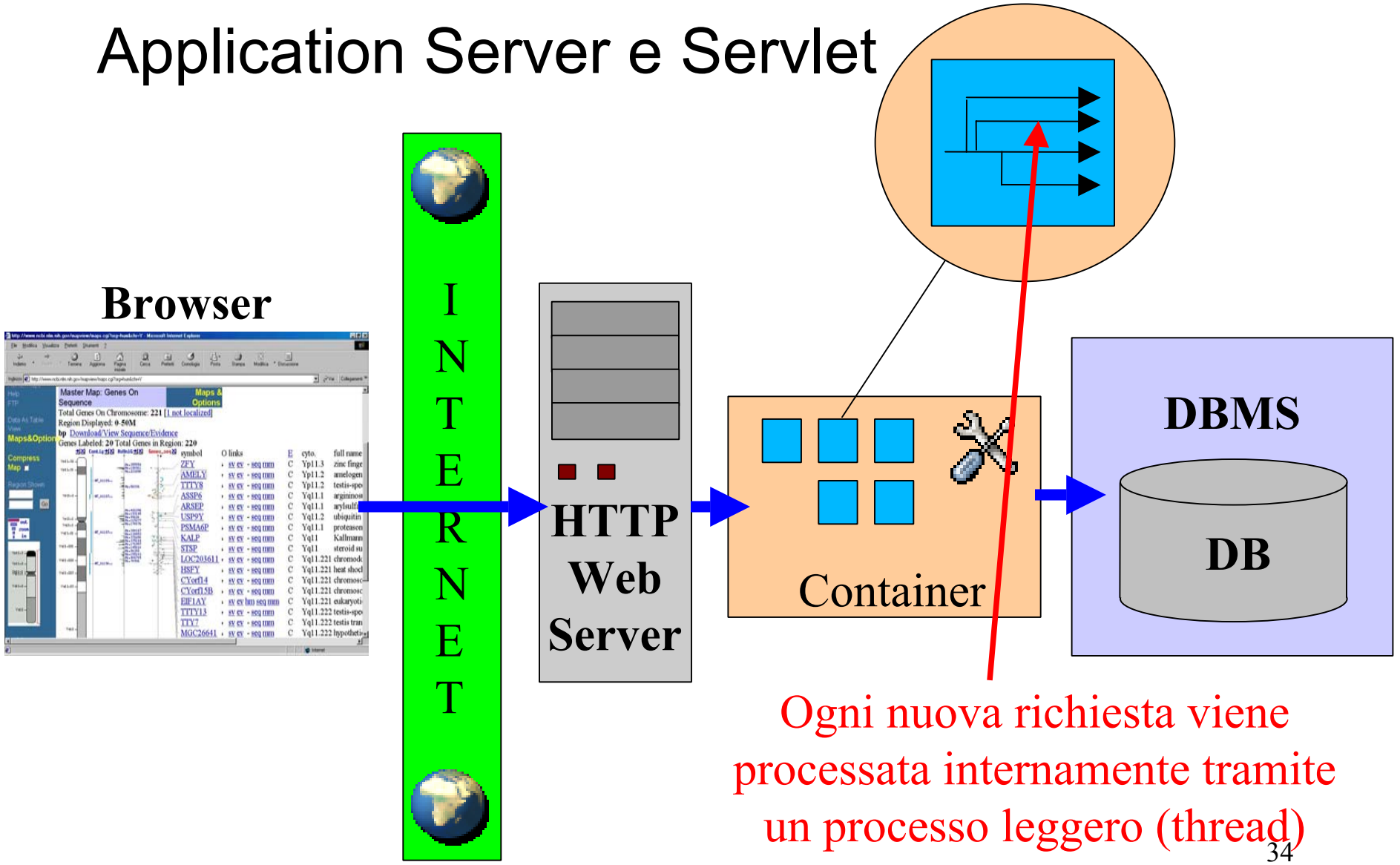
Invocazione di Servlet

- L'invocazione, come detto, e' sempre uguale.

```
<HTML><BODY>
<FORM action="servlet/find" method="post">
    Scrivi il nome dell'autore desiderato:
    <INPUT  type="text"   name="authorName"   size="30"
            maxlength="50">
    <INPUT type="submit"  value="Invia la richiesta">
    <INPUT type="reset"  value="Nuova richiesta">
</FORM>
</BODY></HTML>
```



Application Server e Servlet



Servlet

- Una servlet viene gestita da un applicazione, detta “contenitore”, che ne controlla il ciclo di vita e la interfaccia con l'esterno, utilizzando tutte le potenzialita' di java.
- Il metodo principale di una servlet e' service(), che mette a disposizione i parametri di ingresso e lo stream di output.

Metodo principale

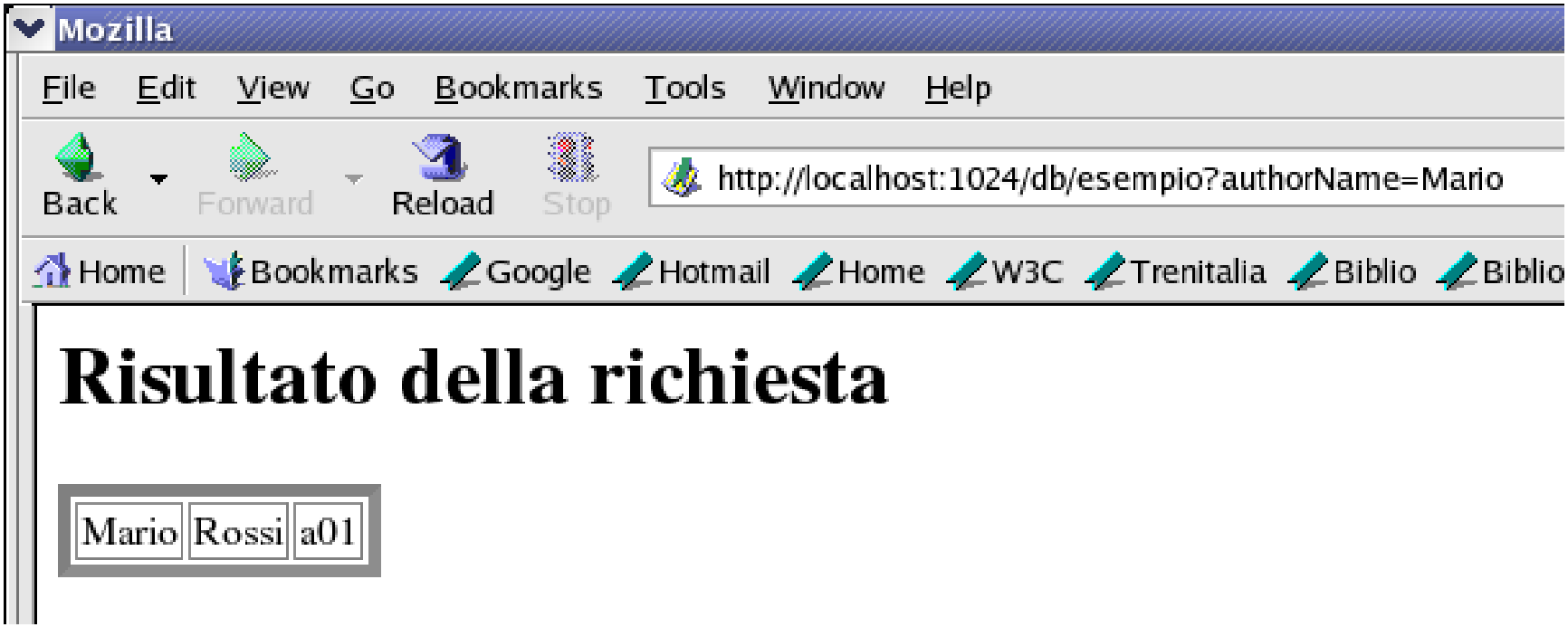
```
public void service(HttpServletRequest req,
                    HttpServletResponse res)
                    throws ServletException, IOException {
  res.setContentType("text/html");
  PrintWriter out = res.getWriter();
  String name = req.getParam("authorName");
  out.println("<h1>Risultato della richiesta</h1>");
  out.println("<table border='5'>");
  try { Statement stmt = con.createStatement();
 ResultSet rs = stmt.executeQuery("SELECT * FROM
  autori WHERE nome = " + name);
```

Metodo principale

```
        while (rs.next()) {
        out.print("<tr>");
        out.println("<td>" + rs.getString(1) + "</td>");
        out.println("<td>" + rs.getString(2) + "</td>");
        out.println("<td>" + rs.getString(3) + "</td>");
        out.print("</tr>");
    }
    stmt.close();
    out.println("</table>");
}
catch (SQLException e) {System.out.println(e);}
```

ECCETERA

Snapshot: Servlet connessa a Postgresql



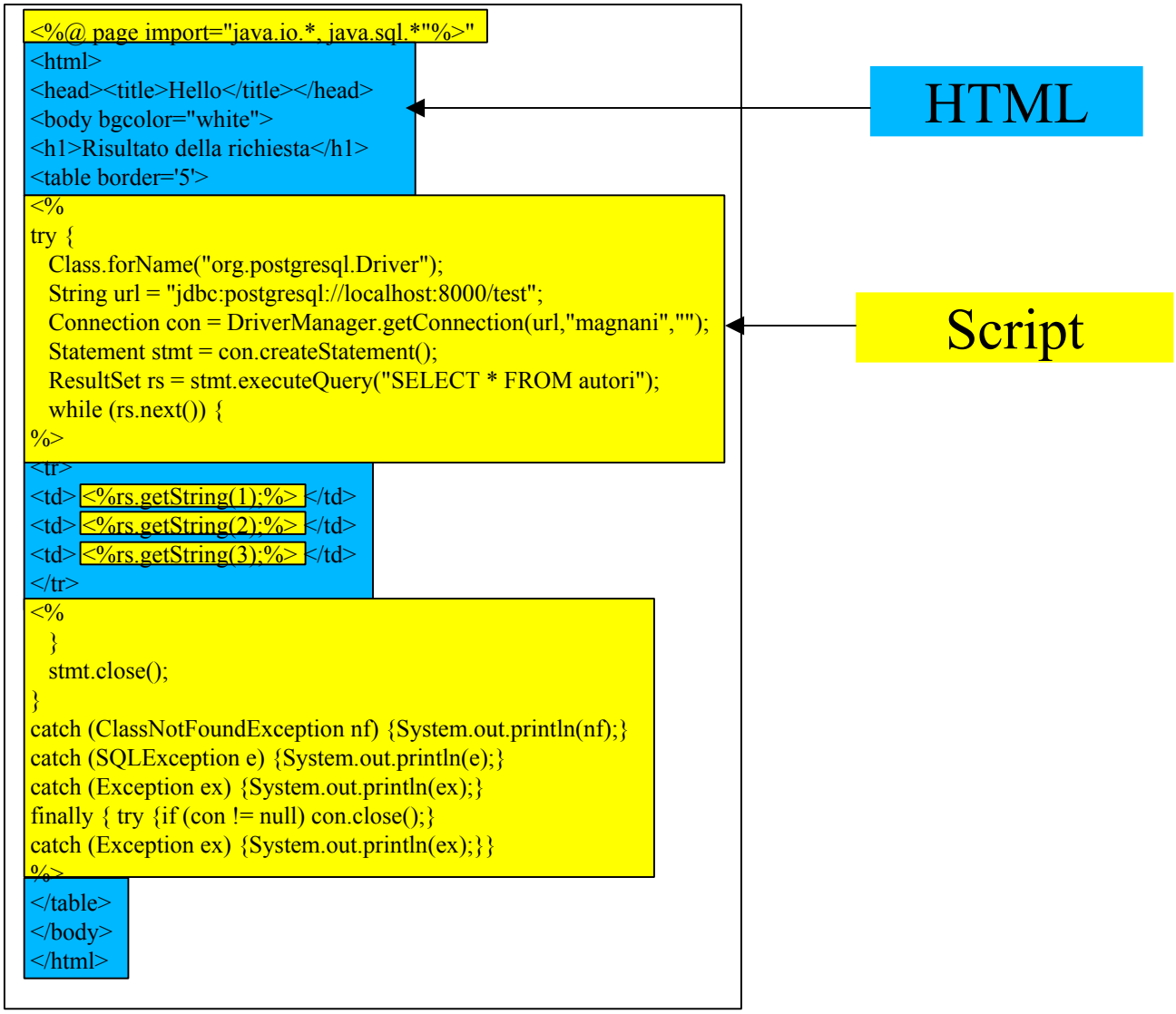
Separazione tra presentazione e contenuto

- Una servlet e' una soluzione abbastanza soddisfacente, ma non del tutto.
- Istruzioni per produrre HTML sono immerse nel codice java, rendendolo poco leggibile.
- Se si cambia l'aspetto della pagina (cosa che a volte cambia molto piu' velocemente del codice) bisogna riscrivere la servlet.
- La soluzione consiste nel separare la parte HTML dal codice java.

Separazione tra presentazione e contenuto

- Esistono diversi modi per realizzare la separazione tra presentazione e contenuto.
- ASP, ASP.NET, PHP, JSP
- Le pagine JSP vengono compilate in servlet, per cui ne mantengono i vantaggi, guadagnando in leggibilita' e riusabilita' del codice.

JSP


<%@ page import="java.io.*, java.sql.*"%>"
<html>
<head><title>Hello</title></head>
<body bgcolor="white">
<h1>Risultato della richiesta</h1>
<table border='5'>
<%
try {
Class.forName("org.postgresql.Driver");
String url = "jdbc:postgresql://localhost:8000/test";
Connection con = DriverManager.getConnection(url,"magnani","");
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM autori");
while (rs.next()) {
%>
<tr>
<td> <%rs.getString(1);%> </td>
<td> <%rs.getString(2);%> </td>
<td> <%rs.getString(3);%> </td>
</tr>
<%
}
stmt.close();
}
catch (ClassNotFoundException nf) {System.out.println(nf);}
catch (SQLException e) {System.out.println(e);}
catch (Exception ex) {System.out.println(ex);}
finally { try {if (con != null) con.close();}
catch (Exception ex) {System.out.println(ex);}}
%>
</table>
</body>
</html>
HTML
Script

Snapshot: JSP con postgresql

Considerazioni finali

- Soluzioni lato-client sono opportune per scenari in cui:
  - Esiste un controllo sui client.
  - L'applicazione Web non e' molto “pesante”.
- Attenzione: anche Java non e' portabile! Opera con le vecchie plug-in java si interrompe. Con la piu' recente l'applet dell'esempio va in crash. Con Mozilla/Netscape non si riescono a chiamare i metodi dell'applet tramite javascript. Oltre ad altri problemi che si scoprono solo dopo molte linee di codice.

Considerazioni finali

- La soluzione lato-server deve essere scelta sulla base della **complessita'** dell'applicazione (per semplici task le CGI in perl vanno benissimo), il **numero di utenti** (idem), la **riusabilita'** e la **manutenzione** del codice, la **piattaforma** di sviluppo e di utilizzo dell'applicazione.

Considerazioni finali

- In casi reali, le tecnologie presentate si utilizzano in modo piu' sofisticato.
- Ad esempio, le connessioni in Java non si aprono tramite il DriverManager, le eccezioni e altri parametri sono gestiti in modo piu' puntuale, si possono usare Enterprise JavaBeans, risorse esterne di vario tipo, filtri.

Bibliografia

- Atzeni, Ceri, Paraboschi, Torlone
  Basi di dati
  McGraw-Hill
  capitolo "Basi di dati e World Wide Web"

Webliografia

- Tutorial su vari argomenti trattati riguardanti java:

  http://developer.java.sun.com/developer/onlineTraining/