Université de Montréal

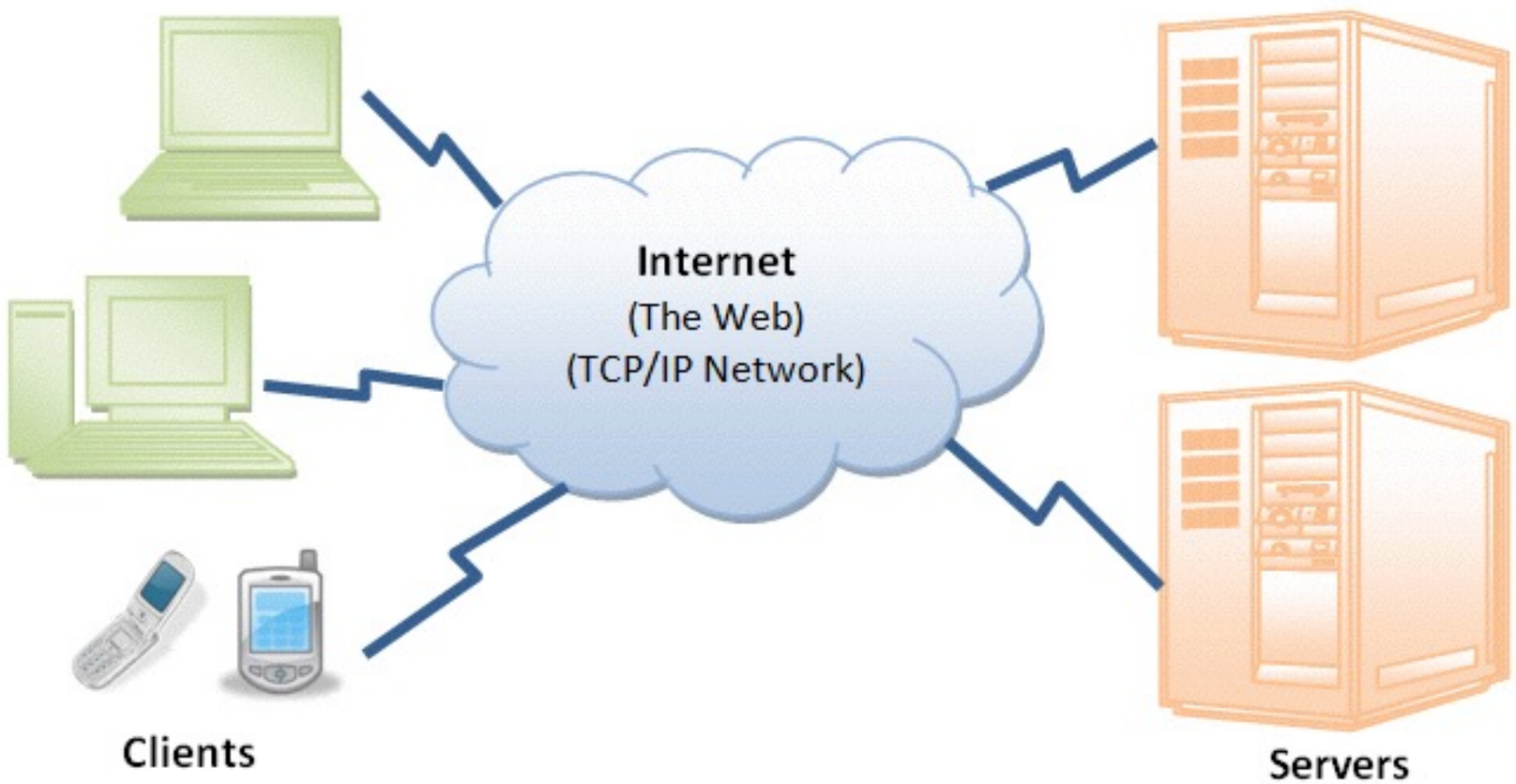
HTTP
Serveurs Web

Guy Lapalme

fortement inspiré de
http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html

rali

Internet

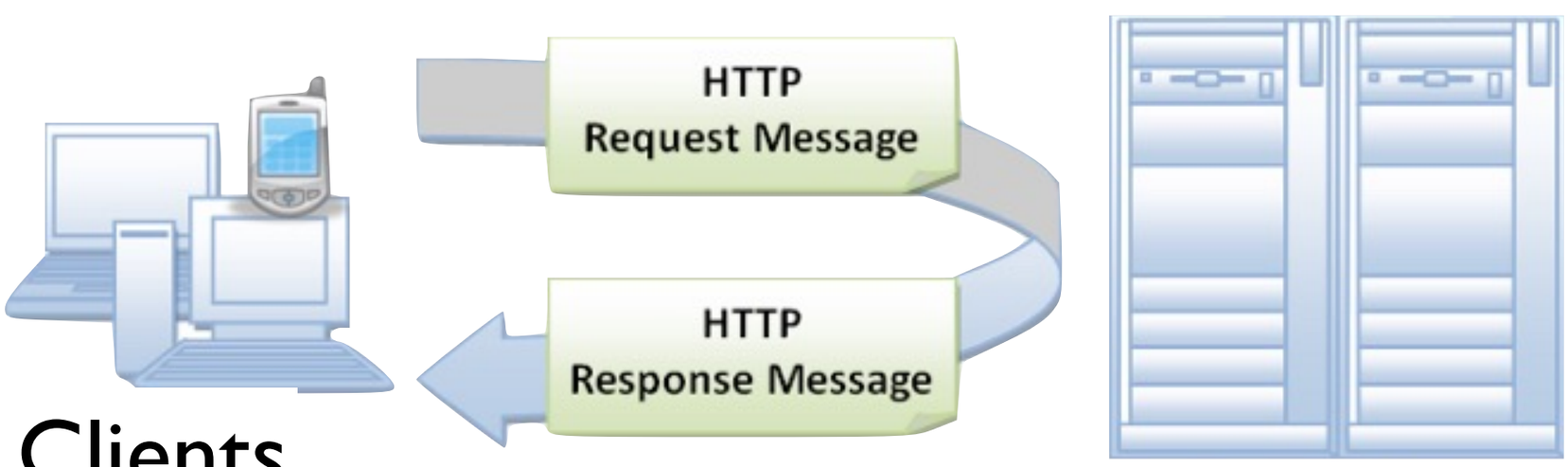


HyperText Transfer Protocol - HTTP

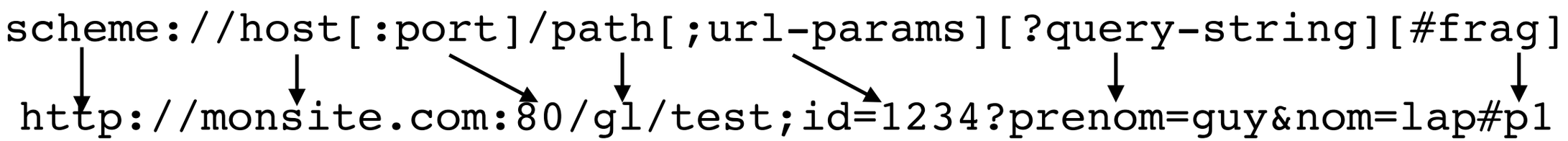
initialement RFC 2616

maintenant RFC 7230 à RFC 7235

- protocole de transfert asymétrique (pull) **requête**/réponse ou **client**/serveur
- sans état
- permet la négociation des types de données et de leur représentation



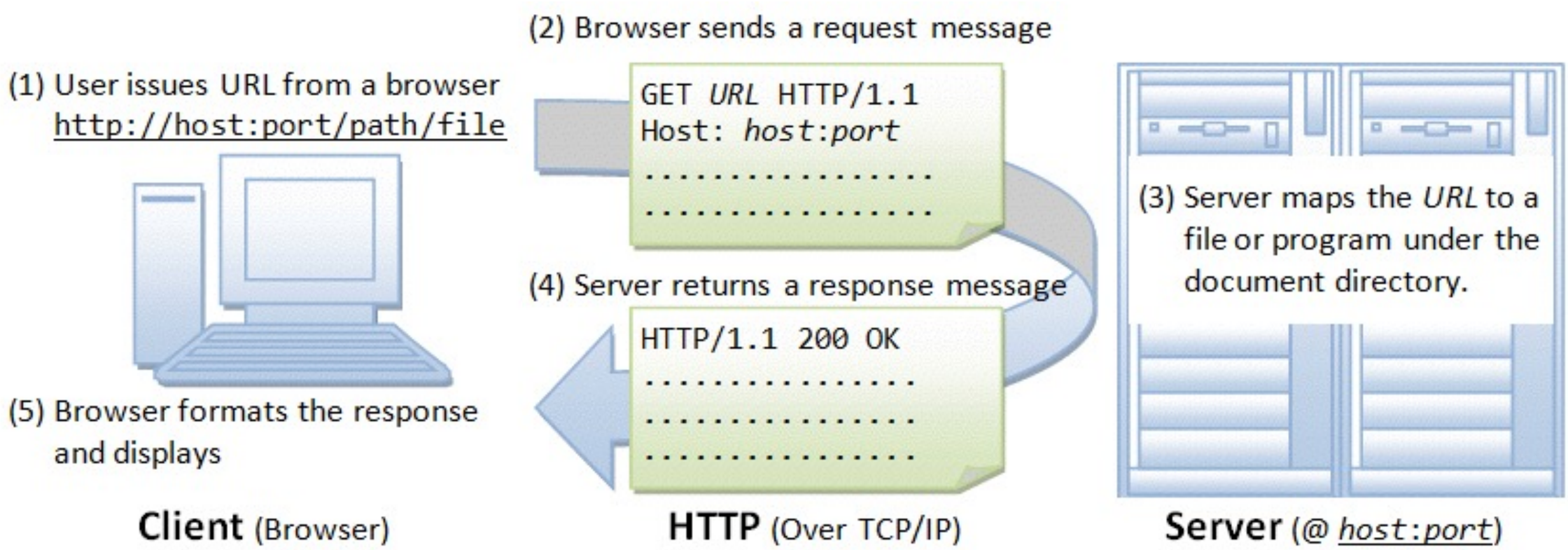
Uniform Resource Locator - URL
Uniform Resource Identifier - URI



- analysable avec une regex
- query-string est *encodé* pour les caractères spéciaux (~: %7e, +:%2b, espace: + ou %20)
- autres types d'url
  - ftp://ftp.is.ca.ca/rfc/rfc1808.txt
  - ldap://[2001.db8::7]/c=GB?objectClass?one
  - mailto:lapalme@iro.umontreal.ca
  - tel:+1-514-343-6111
  - telnet: 132:204.24.179
  - isbn: 978-2-1234-5680-3

rali

Browsers

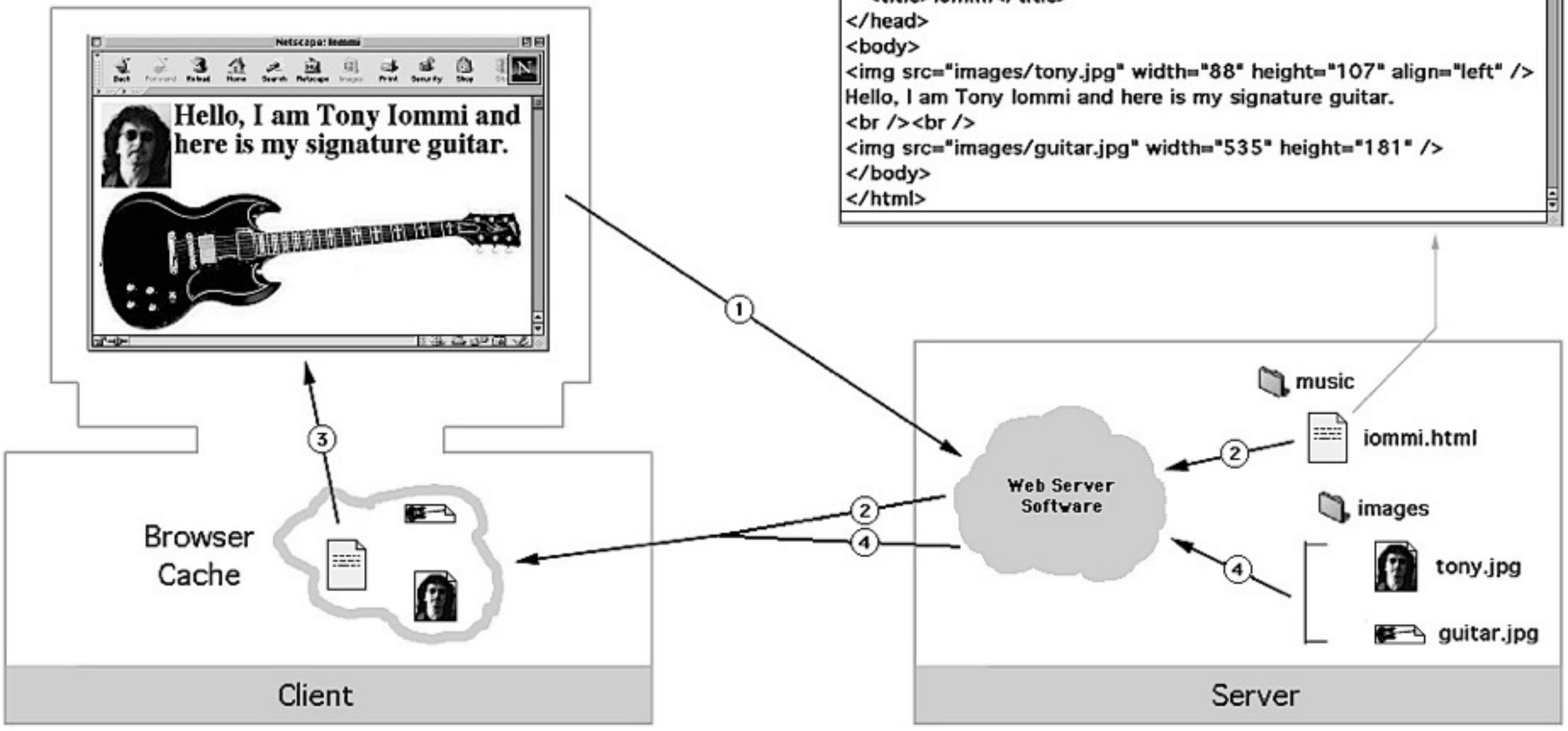


- Cet échange est effectué pour chaque élément différent (e.g. image, script, css) d'une page web
- Une même page peut combiner des informations provenant de plusieurs serveurs

Affichage simple HTML

1. URL request `http://music.uweb.edu/iommi.html`
2. A copy of the HTML document is transferred into the browser's cache.
3. The browser starts parsing the HTML code and asks the Web server software to send the two image files. (The "kept alive" socket is still being used.)
4. The image files are transferred into the cache, completing the Web page.

```
<html>
<head>
  <title>Iommi</title>
</head>
<body>
<img src="images/tony.jpg" width="88" height="107" align="left" />
Hello, I am Tony Iommi and here is my signature guitar.
<br /><br />
<img src="images/guitar.jpg" width="535" height="181" />
</body>
</html>
```

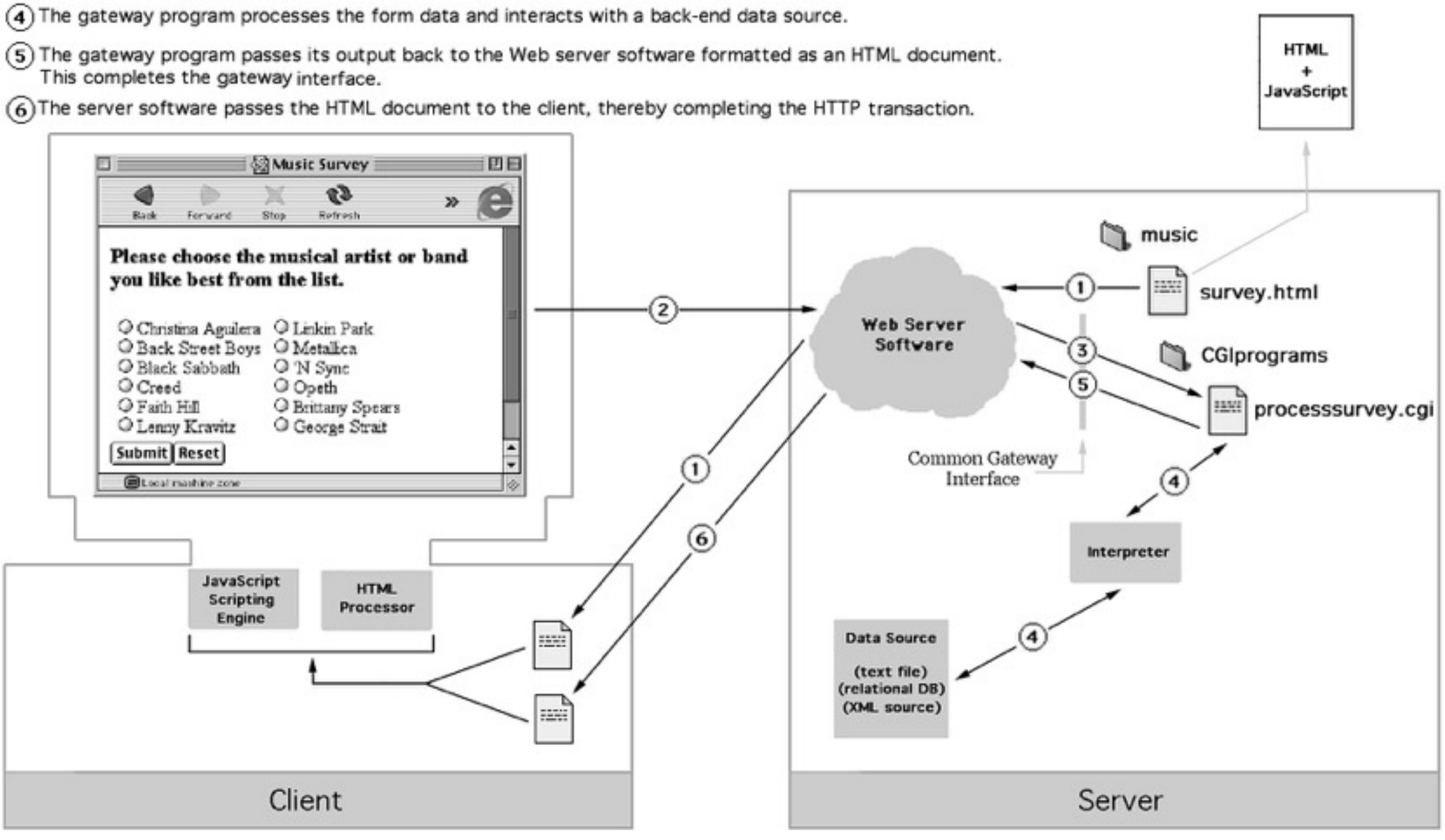


http://www.cknuckles.com/webapps/chap01/figures/fig1.08.gif

rali

Interaction plus complexe

1. The front end of the Web application is passed to the client as an HTML file that contains some JavaScript code. The JavaScript validates the form data on the client.
2. The user submits the form to a program named `processsurvey.cgi` on the Web server. Fundamentally, this is an HTTP transaction between the browser and the server software, but the URL points to an executable gateway program.
3. The server software executes the gateway program and passes it the form data. This is the first part of the gateway interface.
4. The gateway program processes the form data and interacts with a back-end data source.
5. The gateway program passes its output back to the Web server software formatted as an HTML document. This completes the gateway interface.
6. The server software passes the HTML document to the client, thereby completing the HTTP transaction.



http://www.cknuckles.com/webapps/chap01/figures/fig1.09.gif

rali

Traitement sur le serveur - I

- Requête arrive au serveur
- Résolution d'adresse
  - Virtual hosting
  - Address mapping (statique vs dynamique, alias)
  - Authentification

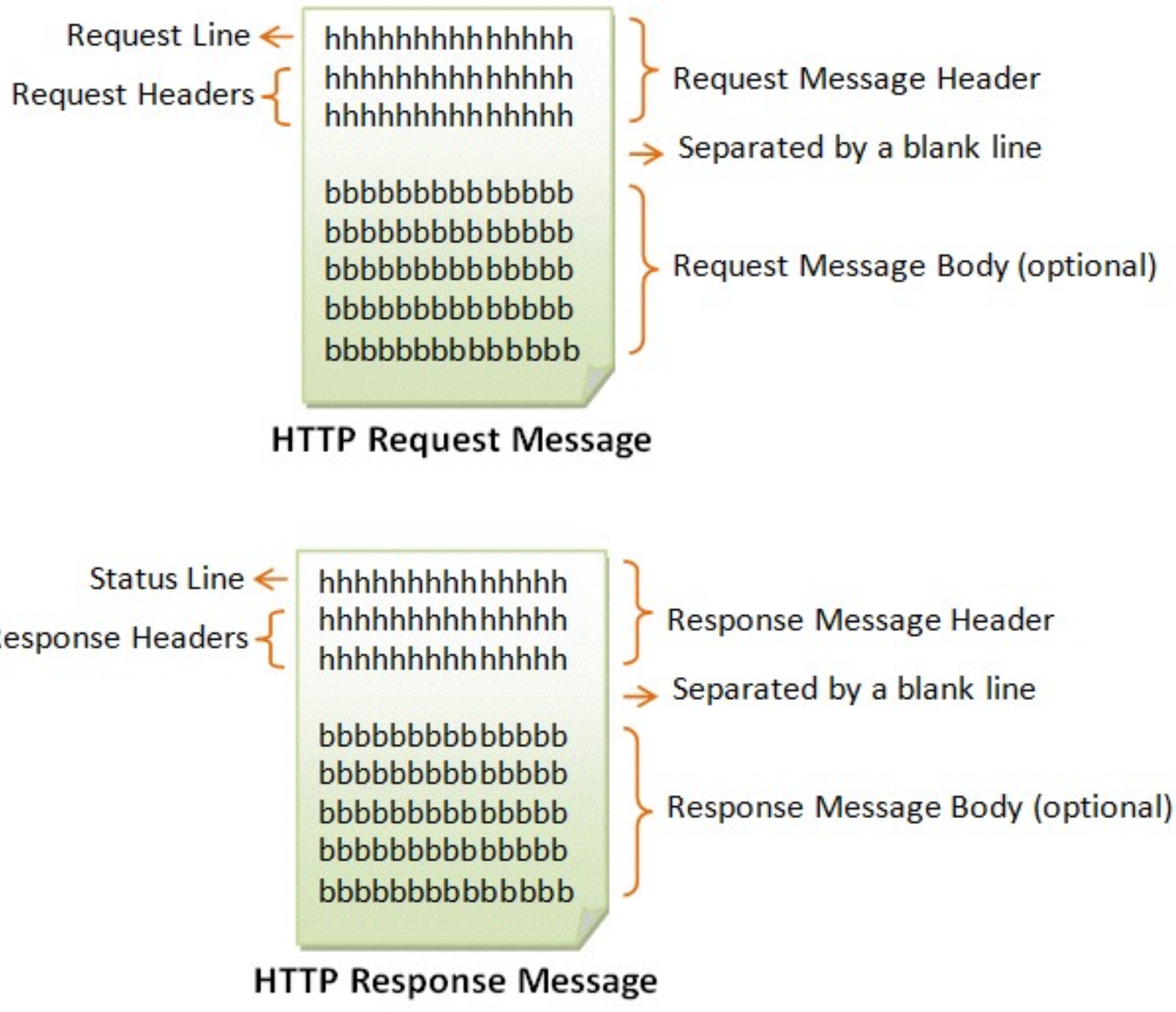
rali

Traitement sur le serveur - 2

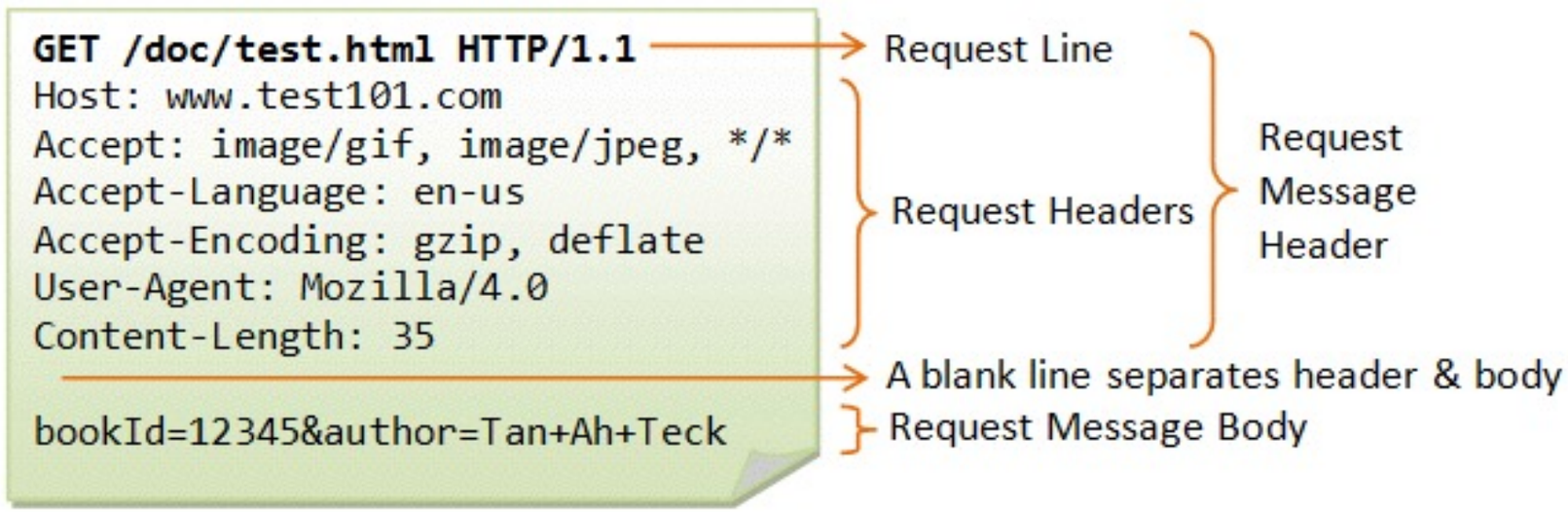
- Traitement de la requête
  - statique : retourne la page telle quelle
  - dynamique : lance un programme pour générer la page résultante
    - CGI
    - Servlet
    - Server Side Include
    - ASP / JSP
    - PHP
- Génération de la réponse (avec bonnes entêtes)

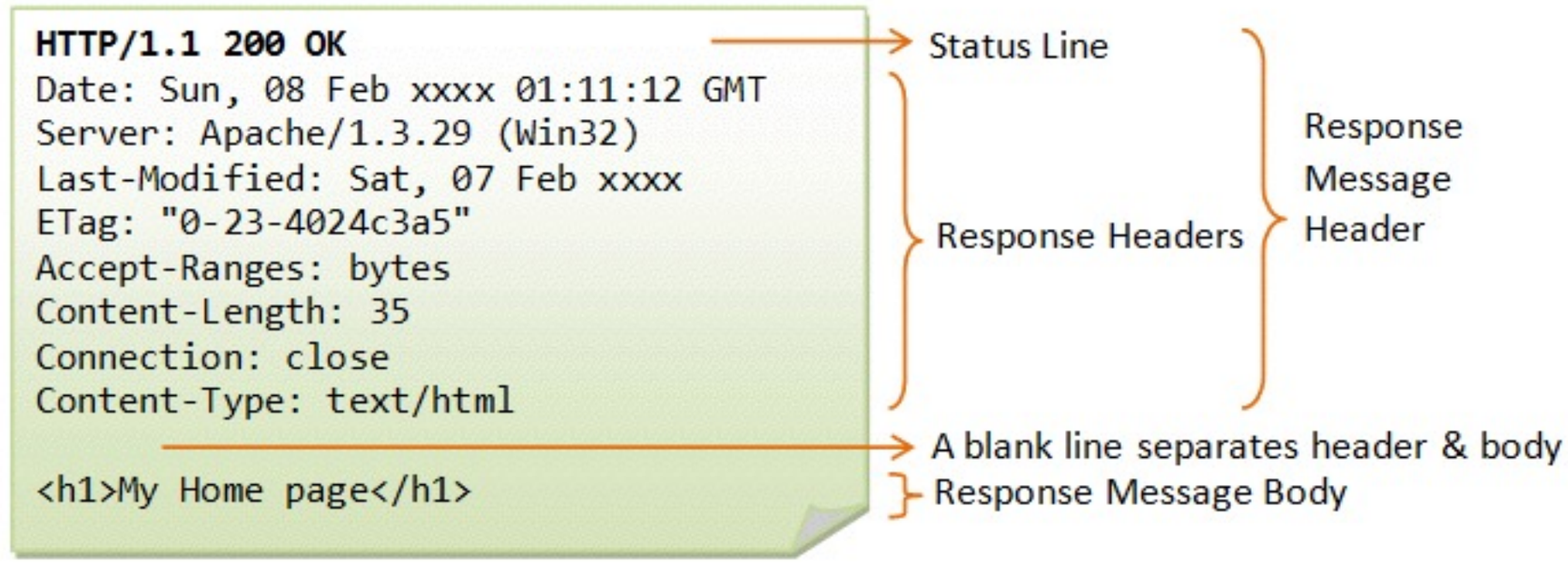
rali

Structure d'un message HTTP



Exemples de messages





Principales requêtes HTTP

- GET : recherche une ressource sur le serveur (*sans effet de bord*)
- POST : exécute le contenu de la resource avec le contenu du corps (*effet de bord possible*)
- HEAD : ne récupère que l'entête d'un GET
- PUT : remplace la ressource avec le contenu du message
- DELETE : détruit une ressource

rali

Principales réponses HTTP

- 1XX : Information (p.ex. requête reçue, le traitement continue)
- 2XX : Succès
- 3XX : Redirection
- 4XX : Erreur du client
  - 404 : non disponible
  - 402 : demande authentification
  - 403 : refusé
- 5XX : Erreur du serveur
  - 500 : erreur interne au serveur
  - 503 : serveur surchargé

rali

Tester HTTP avec serveur local

`python -m SimpleHTTPServer`

- sert les fichiers du répertoire courant
- sur le port 8000
- tester
  - dans un browser avec
    `http://localhost:8000`
  - `telnet 127.0.0.1 8000`
    `GET / HTTP/1.0`
  - programme Java

rali

Java

```
import java.net.*;
import java.io.*;

public class HttpClient {
   public static void main(String[] args) throws IOException {
      // The host and port to be connected.
      String host = "127.0.0.1";
      int port = 8000;
      // Create a TCP socket and connect to the host:port.
      Socket socket = new Socket(host, port);
      // Create the input and output streams for the network socket.
      BufferedReader in
         = new BufferedReader(
               new InputStreamReader(socket.getInputStream()));
      PrintWriter out
         = new PrintWriter(socket.getOutputStream(), true);
      // Send request to the HTTP server.
      out.println("GET /xhtml/ HTTP/1.0");
      out.println();   // blank line separating header & body
      out.flush();
      // Read the response and display on console.
      String line;
      // readLine() returns null if server close the network socket.
      while((line = in.readLine()) != null) {
         System.out.println(line);
      }
      // Close the I/O streams.
      in.close();
      out.close();
   }
}
```

rali

Quelques GET *intéressants*

- `GET /fichier`
  - 404 si fichier n'existe pas
  - 200 + contenu du fichier s'il existe
- `GET /dir`
  - 301 Moved permanently location: /dir/
- `GET /dir/`
  - 200 Liste du répertoire en HTML si configuré
  - 200 index.html si configuré

rali

Quelques headers

- `Host:` *domain-name*
- `Accept:` *mime-type-1, mime-type-2,...*
- `Accept-Charset:` *Charset-1, Charset-2,...*
- `Connection:` *Close|Keep-Alive*
- `Referer:` *referer-Url*
- `User-Agent:` *browser-type*
- `Content-Length:` *number-of-bytes*
- `Cookie:` *name1=value1, name2=value2*

rali

Formulaire (source)

- `GET` *`request-URI?query-string`* `HTTP-version`
  - autres headers
  - ligne vide
  - corps optionnel
- `POST` *`request-URI`*
  - `Content-type:` *`mime-type`*
  - `Content-length:` *`number-of-bytes`*
  - autres headers
  - ligne vide
  - *query-string* url-encoded

rali