Web

Master 1 IFI

Andrea G. B. Tettamanzi

Université de Nice Sophia Antipolis

Département Informatique

andrea.tettamanzi@unice.fr

Unit 3

The Common Gateway Interface and Server-side Programming

Agenda

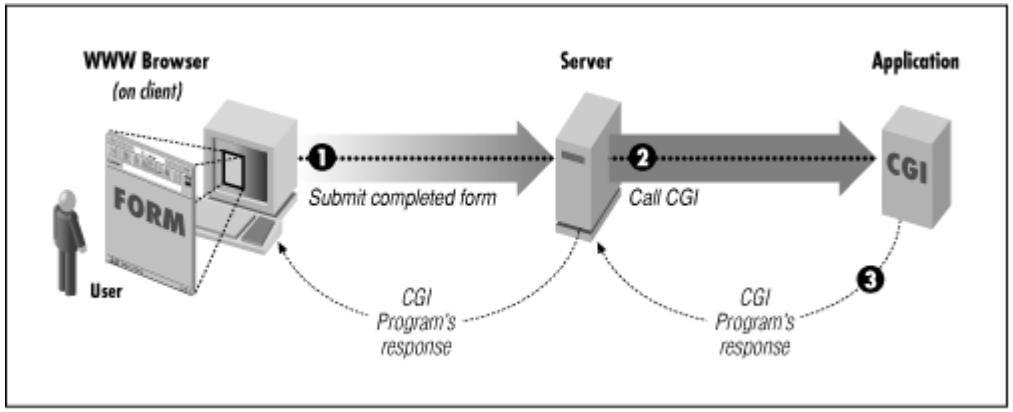
- The Common Gateway Interface
- Server-Side Programming

Introduction

- An HTTP server is often used as a *gateway* to a different information system (legacy or not), for example
  - an existing body of documents
  - an existing database application
- The Common Gateway Interface (CGI) is an agreement between HTTP server implementors about how to integrate such gateway scripts and programs
- It was typically (but not exclusively) used in conjunction with HTML forms to build database applications
- Nowadays largely superseded by dynamic Web content technologies such as PHP, ASP.NET, Java Servlets, and Node.js

The Common Gateway Interface

- The Common Gateway Interface (CGI) is a *de facto* standard protocol for Web servers to execute an external program that generates a Web page dynamically
- The external program executes like a console application running on the same machine as the Web server (the host)
- Such program is known as a CGI script or simply as a CGI



How Does That Work?

- Each time a client requests the URL corresponding to a CGI program, the server will execute it in real-time
  - E.g.: GET http://www.example.org/cgi-bin/add?x=2&y=2
- The output of the program will go more or less directly to the client
- Strictly speaking, the “input” to the program is the HTTP request
- Environment variables are used to pass data about the request from the server to the program
  - They are accessed by the script in a system-defined manner
  - Missing environment variable = NULL value
  - Character encoding is system-defined

The Environment Variables

- AUTH_TYPE
- CONTENT_LENGTH
- CONTENT_TYPE
- GATEWAY_INTERFACE
- HTTP_*
- PATH_INFO
- PATH_TRANSLATED
- QUERY_STRING
- REMOTE_ADDR
- REMOTE_HOST
- REMOTE_IDENT
- REMOTE_USER
- REQUEST_METHOD
- SCRIPT_NAME
- SERVER_NAME
- SERVER_PORT
- SERVER_PROTOCOL
- SERVER_SOFTWARE

REQUEST_METHOD

- The method with which the request was made:
  - GET, HEAD: see the QUERY_STRING variable
  - POST: see the request content variables and standard input

Request Content Variables

- CONTENT_LENGTH
  - The size of the data attached to the request, if any
  - Decimal number of octets (bytes)
- CONTENT_TYPE
  - The MIME type of the data attached to the request
    - `application/x-www-form-urlencoded` (HTML Form)
  - If the type remains unknown, assume
    - `application/octet-stream`

QUERY_STRING

- The query string of an URI is anything that follows the “?”
- Example:
  - http://www.example.org/cgi-bin/add?**x=2&y=2**
  - Here, the query string is “x=2&y=2”
- The QUERY_STRING variable contains the query string of the CGI program’s URI
- In principle, it is up to the CGI script to parse the query string to extract the parameters, e.g.:
  - x = 2
  - y = 2

Path Variables

- Assume a CGI program called “script” is invoked via a request
  - GET http://www.example.com/cgi-bin/script/a/path/x.html?q
- The string “/a/path/x.html” is called an extra-path
- The extra-path is anything between the name of the CGI program and the “?” that introduces the query string
- This extra-path is passed to the CGI program via the **PATH_INFO** environment variable
- The **PATH_TRANSLATED** variable contains the operating system path corresponding to PATH_INFO

  - CGI programs using this variable may suffer limited portability

REMOTE_* Variables

- REMOTE_ADDR: The IP address of the agent sending the request. Not necessarily that of the client
- REMOTE_HOST: The fully qualified domain name of the agent sending the request
- REMOTE_IDENT: The identity information reported about the connection by a RFC 931 request to the remote agent, if available
- REMOTE_USER:
  - If AUTH_TYPE = "Basic", then the user-ID sent by the client
  - If AUTH_TYPE = NULL, then NULL
  - Otherwise, undefined

HTTP_* Variables

- HTTP_COOKIE: all the set cookies
  - It is up to the CGI program to parse the cookie string
- Cookies can be set in the header of a response:
  - Set-Cookie:<key> = <value>;
  - Set-Cookie:Expires = <expiration date>
  - Set-Cookie:Domain = www.example.org
  - Set-Cookie:Path = <path>
- Before the Content-type header!

Accessing Environment Variables

- In C:
  - #include <stdlib.h>
  - char *getenv(const char *name);
  - Example: browser = getenv("HTTP_USER_AGENT");

- In Python:
  - import os
  - os.environ[<varname>]
  - Example: browser = os.environ["HTTP_USER_AGENT"]

The Command Line

- Most systems support a method for supplying a array of strings to the CGI programs (i.e., command-line arguments)
- This is only used in the case of an “indexed” query, i.e.:
  - A "GET" or "HEAD" HTTP request
  - with a URL search string not containing any "=" characters
- For such a request, the Web server will break the search string into words
- Each word is URL-decoded
- Then, the argument list is set to the list of words

The Standard Input

- For some methods, there may be data attached to the request
  - This is the case for POST, PUT, and PATCH
- Therefore, there must be a system-defined method for the CGI program to read these data
- The standard way to pass data is via the “standard input” file descriptor (`stdin` in C)
- There will be at least CONTENT_LENGTH bytes available for the program to read
- The program is not obliged to read (all) the data
- … but it must not attempt to read more than CONTENT_LENGTH bytes, even if more data is available

Output

- A CGI program will always return some data:
  - An HTML page
  - An image
  - A file of some other type (e.g., a PDF file)
- This is via the “standard output” file descriptor (`stdout` in C)
- The program must return a **complete** HTTP response message!
  - Header (status, content-type, etc.)
  - Body (the actual content, in the declared format)

Error Handling

- CGI programs should reject unexpected methods (such as DELETE etc.) with error
  - 405 Method Not Allowed
- If the script does not intend processing the PATH_INFO data, then, if PATH_INFO is not NULL, it should reject the request with
  - 404 Not Found
- If the output of a form is being processed, check that CONTENT_TYPE = "application/x-www-form-urlencoded"
  - Presumably 400 is the appropriate error code in this case...

Server-Side Programming

- The CGI is interesting for historic reasons
- … but also because it is at the origin of server-side programming
- Today's frameworks all build upon the basic concepts of CGI
- The main improvements with respect to the CGI are
  - Better integration with the HTTP server
  - Basic APIs and utilities allow for faster development
  - The script runs in the same process as the server
  - The script can be embedded in HTML pages providing a static skeleton, where the generated parts can be inserted

Example: PHP Hello World

```
<html>
 <head>
  <title>PHP Test Page</title>
 </head>
 <body>
 <?php
    echo '<p>Hello World!</p>';
 ?>
 </body>
</html>
```