Web applications and PHP

15-415 Database Applications

Spiros Papadimitriou
spapadim+@cs.cmu.edu

In the beginning…

- Hypertext
  - Documents with embedded links (i.e., pointers) to other documents/resources
  - A very simple and old idea (Vannevar Bush?)
- Hypertext on the Internet
  - Well, all this is very nice…
  - How do we get to documents on different machines?
  - Tim Berners-Lee (CERN) "hack"
  - Historical note: Gopher…

In the beginning…

- URLs, HTTP and HTML
- URL: Unique, universal resource names
  - E.g., `http://www.cmu.edu/path/index.html`, `file:///path/index.html`
- HTTP: Simple file access protocol
- HTML: Markup language
  - Based on SGML
  - Presentation and hyperlinks (aka. anchors)

HTTP

- Request

```
GET /path/index.html HTTP 1.0
Host: www.cmu.edu
```

- Response

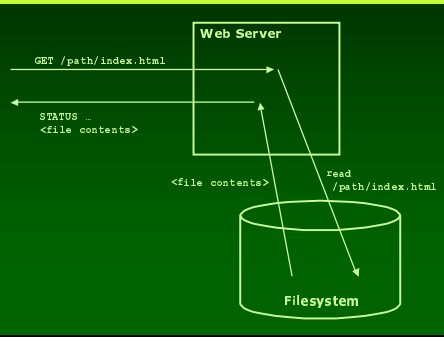
```
HTTP 1.1 200 OK
Content-Type: text/html

<file contents> …
```

Static content

- Static webpages
  - Plain files stored in the filesystem
  - Webserver accepts pathnames
  - Returns contents of corresponding file
- Boring!
  - Can't generate customized content—always serve the same static pages…

Static pages



Dynamic content

- Need some way to
  - Pass arguments
  - Generate output
- HTTP: support for "arguments"
  - In general: KEY=value
  - Also: multipart encoding (e.g., file uploads)…

Dynamic content – Arguments

- **GET** requests: part of URL

```
GET /cgi-bin/query?KEY=value HTTP 1.0
Host: www.cmu.edu
```

- **POST** requests: part of request body

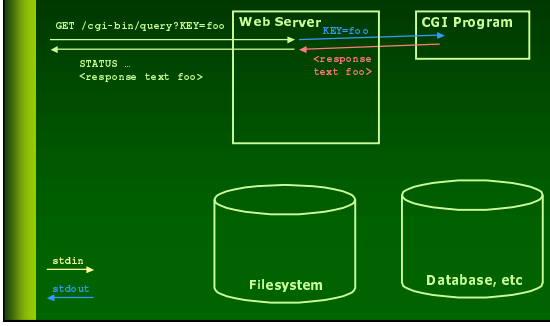
```
POST /cgi-bin/query HTTP 1.0
Host: www.cmu.edu

KEY=value
```

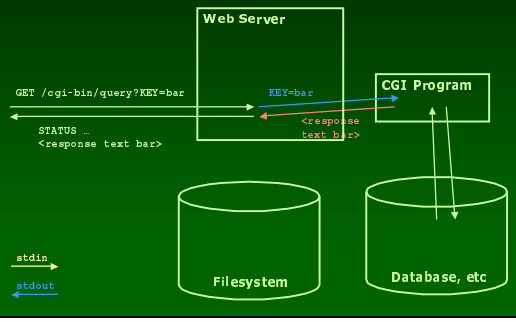
Dynamic content – CGI

- CGI: Easy "fix" (yet another "hack")
  - Common Gateway Interface
  - Oldest standard
  - But at least a **standard**!
  - Inefficient
  - No persistent state
- Forward requests to external programs
  - Spawn one process for each new request (ouch!)
  - Communicate via
    - Standard input/output
    - Environment variables
  - Process terminates after request is handled

CGI



CGI



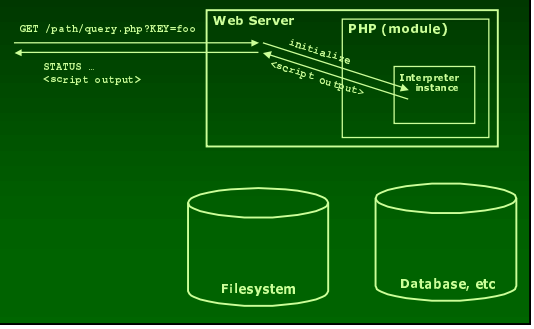
More "modern" solutions

- Do things *in the webserver*
  - mod_perl (Perl interpreter)
    - See also: ePerl
  - mod_python (Python interpreter)
  - Servlets (JVM)
    - See also: JSP
  - mod_php (PHP interpreter)
  - Also: custom modules…

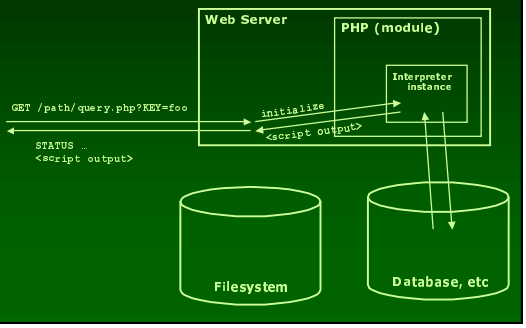
Side note – Other alternatives

- Dynamically loadable server modules
  - Web server offers standard API for *request handlers*
  - Request handlers are dynamic libraries
  - Loaded into server address space (security?!)
  - Very efficient!
  - Better suited for generic server extensions than application-specific logic
- CGI flavours
  - Persistent CGI (PCGI) / FastCGI
  - Too little, too late
- Modern web publishing systems
  - Essentially databases that "speak HTTP" (e.g., Zope)

PHP



PHP



PHP

- Different requests can be handled by the same or a new interpreter instance
- Request parameters (PHP auto-globals)
  - `$_GET["KEY"]`
  - `$_POST["KEY"]`

PHP – Hello world

```
<html>
<head>
<title>Hello world!</title>
</head>

<body bgcolor="#FFFFFF">

<h1>Hello world!</h1>

<p>Your query key was <?php echo $_GET["KEY"] ?>
</p>

</body>
</html>
```

PHP – PostgreSQL

```
<html>
<head>
<title>Hello world!</title>
</head>
<body bgcolor="#FFFFFF">
<?php
// Handle both POST and GET
$key = $_POST["KEY"];
if (!$key) $key = $_GET["KEY"];

$dbconn = pg_connect(…); // Use db_connect()…
$sql = "SELECT a, b FROM table WHERE key='$key'";
$result = pg_exec($dbconn, $sql);
?>
```

PHP – PostgreSQL (cont'd)

```
<?php
$num_rows = pg_numrows($result);
if (!$result) {
?>
<h1>Error</h1>
<p>Query failed!</p>
<?php
} elseif ($num_rows == 0) {
?>
<h1>Results</h1>
<p>Query returned no results!</p>
```

PHP – PostgreSQL (cont'd)

```
<?php
} else {
?>
<h1>Results</h1>
<p><table border="1">
<tr><th>A</th><th>B</th></tr>
<?php
  for ($i = 0;  $i < $num_rows;  $i++) {
    $row = pg_fetch_row($result, $i);
    echo "<tr><td>{$row[0]}</td>";
    echo "<td>{$row[1]}</td></tr>\n";
  }
?>
</table></p>
```

PHP – PostgreSQL (cont'd)

```
<?php
} // end if
?>
</body>
</html>
```

PHP – Feature list

- Look at online manual
  - http://www.php.net/manual/en/
- Functions
  - Note that variables are local, unless explicitly declared `global`
- Classes
  - You probably won't need them
- Rich library
  - Also: session management (persistence)
  - Extensions (Zend)

PHP – Some debugging tips

- PHP error reporting…
- Use `echo`
  - Make sure to remove them later
- Look at `/var/log/httpd/error_log`
- Run from command-line

  `$ php script.php`
  - Need to get args from `$argv[]`