Mestrado Integrado em

Engenharia Electrotécnica e de Computadores

Programação de Sistemas — 2013/2014

Enunciado do projecto

O projecto de Programação de Sistemas consistirá na implementação de um **servidor WEB** (semelhante ao apache ou IIS).

Os servidores web, são aplicações que interagem com os *browsers* fornecendo-lhes o conteúdo de ficheiros .html ou de imagens. Através da Internet recebem os pedidos enviados pelos navegadores e respondem, se o pedido corresponder a um ficheiro existente, com o conteúdo desse ficheiro (.html ou .png).

Os servidores web também podem gerar conteúdos dinâmicos em resposta a *queries*.

Os alunos deverão desenvolver um servidor web que terá as seguintes funcionalidades, de modo a dar resposta aos vários pedidos feitos pelos navegadores:

- Tratamento dos pedido enviados pelos navegadores, com possível devolução do conteúdo do ficheiro (.html ou .png) pretendido;
- Armazenamento das informações relevantes acerca do pedido;
- Geração de relatórios acerca das páginas acedidas, dos erros, dos navegadores usados e dos padrões de navegação dos utilizadores;
- Execução de aplicações externas usando o protocolo CGI (*Common Gateway Interface*).

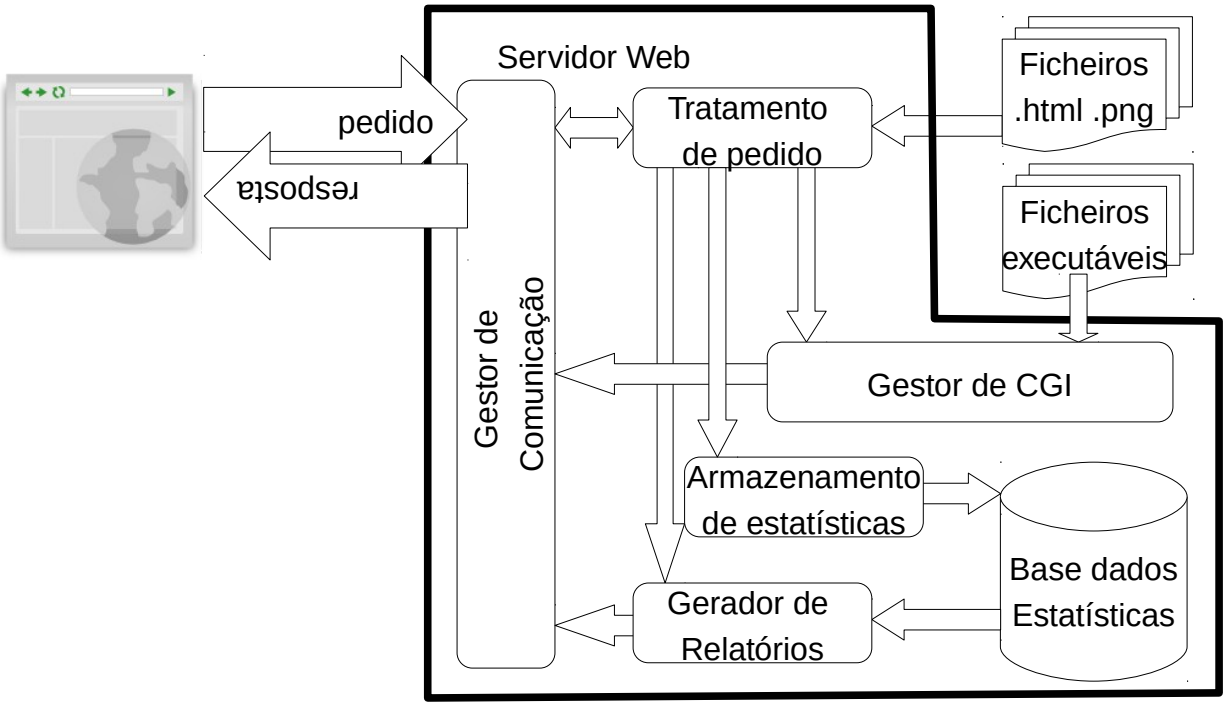
1 Arquitectura

O projecto será composto por uma aplicação, que será desenvolvida pelos alunos, e composta pelos seguintes módulos:

- **Gestor de Comunicação**
- **Tratamento de pedido**
- **Armazenamento de estatística**
- **Gerador de relatórios**

- **Gestor de CGI**

A arquitectura geral do sistema será a seguinte:



O navegador envia um pedido para obtenção de uma página que é tratado e recebido pelo **Gestor de Comunicação**. Após a recepção da mensagem, o módulo **Tratamento de pedido**, descodifica a mensagem e seleciona a ação apropriada:

- se for pedida a geração de um relatório este é produzido pelo **Gerador de relatórios**.
- se for solicitada a execução de uma aplicação (usando o protcolo **CGI)**, esta deverá ser executada e o seu output enviado para o cliente.
- se apenas for pedida uma página .html ou uma imagem .png estas são lidas do disco e enviados para o navegador.

Sempre que é recebido um pedido, toda a informação relevante acerca do mesmo é armazenada. Esta informação será processada mais tarde aquando da geração dos relatórios.

2 Funcionamento do Servidor Web

Quando desejam aceder a uma página ou ficheiro existente num **servidor web**, os navegadores enviam uma mensagem semelhante à seguinte:

```
GET /index.html HTTP/1.1
Host: pc-prog:5001
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.2.24) Gecko/20111107 Ubuntu/10.04 (lucid) Firefox/3.6.24
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

A primeira linha do pedido é sempre composta pela palavra **`GET`**, seguida do nome do ficheiro pretendido e terminada com a versão do protocolo (**`HTTP/1.1`**). Neste exemplo o navegador pretende aceder à página **`/index.html`** . As linhas seguintes contêm informação acerca do navegador, por exemplo, versão ou sistema operativo. O servidor a implementar também deverá aceitar pedidos enviados por navegadores antigos, onde a versão do protocolo é **`HTTP/1.0`** .

Qualquer pedido termina sempre com uma linha em branco (“\r\n”).

Dependendo do sucesso em encontrar o ficheiro ou do tipo do ficheiro assim a resposta enviada para o navegador varia.

2.1 Porto de escuta

O servidor web deverá esperar por ligações TCP/IP no porto **8080**.

2.2 Formato das respostas

O servidor web deve sempre responder ao cliente com uma mensagem bem formatada. Estas mensagens contêm as seguintes linha:

- código da resposta;
- tipo de dados enviados;
- linha em branco;
- dados da mensagem.

2.2.1 Resposta a pedido inválido

Se o formato do pedido enviado pelo navegador não cumprir o especificado na norma, a primeira linha da mensagem de resposta deve ser **`HTTP/1.1 400 Bad Request`** , seguida de uma linha em branco:

```
HTTP/1.1 400 Bad Request

```

2.2.2 Resposta a ficheiro inexistente

Se o pedido enviado pelo navegador cumprir o formato, mas o ficheiro não existir deverá ser enviado o código de erro: **`HTTP/1.1 404 Not Found`**

Associado a este erro pode também ser enviada uma mensagem textual (em html) para o utilizador ler. Assim, a primeira linha, deverá ser seguida da indicação do tipo dos dados enviados (por exemplo html), seguido de uma linha em branco, seguido da mensagem:

```
HTTP/1.1 404 Not Found
Content-Type:text/html

<HTML><HEAD> <TITLE>404 Not Found</TITLE></HEAD>
<BODY>  ficheiro %s nao foi encontrado.</BODY></HTML>
```

2.2.3 Resposta válida

Quando o ficheiro existe e não for um executável, o seu conteúdo será enviado para o navegador.

A primeira linha é indicativa do sucesso (**`HTTP/1.1 200 OK`**), a segunda linha contêm o

tipo do ficheiro enviado (html ou png), e a terceira linha vai em branco.

Após esta linha são enviados os dados tal qual se encontravam no ficheiro.

O seguinte exemplo mostra a mensagem de envio de um ficheiro .html:

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

<html>
<body>
<h1>Happy New Millennium!</h1>
(more file contents)
  .
  .
  .
</body>
</html>
```

O texto que aparece após a palavra **`Content-Type:`** depende do formato do ficheiro. A seguinte tabela apresenta o *Content-Type* associado a vários tipo de ficheiro:

Tipo de Ficheiro	**Extensão**	**Content-Type:**
Ficheiro javaScript	`.js`	`application/javascript`
Ficheiro PDF	`.pdf`	`application/pdf`
Ficheiro MP3	`.mp3`	`audio/mpeg`
Imagem GIF	`.gif`	`image/gif`
Imagem JPEG	`.jpeg` `.jpg`	`image/jpeg`
Imagem PNG	**`.png`**	**`image/png`**
Imagem TIFF	`.tiff` `.tif`	`image/tiff`
Cascading Style Sheel	`.css`	`text/css`
Ficheiro html	**`.html`** `.htm`	**`text/html`**
Ficheiro de texto	`.txt`	`text/plain`
Video MPEG4	`.mp4`	`video/mp4`

2.2.4 Resposta de tipo incompatível

O servidor a implementar só deverá servir e enviar para o navegador ficheiros dos formatos

listados anteriormente e marcados a **Negrito**.

Se o ficheiro não for executável e tiver uma extensão diferente de .png ou .html, deverá ser enviada a seguinte resposta: **`HTTP/1.1 415 Unsupported Media Type`** .

O seguinte exemplo mostra um pedido

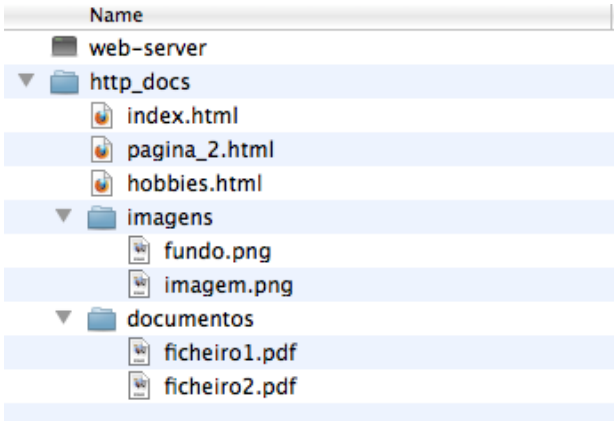
```
GET /exemplo.doc HTTP/1.1
Keep-Alive: 115
Connection: keep-alive
```

e a resposta correspondente

```
HTTP/1.1 415 Unsupported Media Type
```

2.3 Acesso aos ficheiros

Por omissão, os ficheiro a enviar para o navegador encontram-se numa diretoria chamada **`http_docs`** (a DOCUMENT_ROOT do servidor web), presente na mesma directoria do executável, como se mostra na seguinte figura.



Se o servidor receber o pedido **`GET /index.html HTTP/1.1`**, o ficheiro enviado será aquele presente na directoria **http_docs**: **http_docs/index.html** . Para aceder à imagem **fundo.png**, o navegador deverá enviar o pedido **`GET /imagens/fundo.png HTTP/1.1`** .

Só os ficheiros dos tipos listados na secção 2.2.3 (**a bold**) deverão ser servidos, se o navegador tentar aceder a ficheiros com extensão não listada anteriormente, deverá ser retornada uma mensagem com o código 415: **`HTTP/1.1 415 Unsupported Media Type`** .

Se o servidor receber um pedido terminado pelo carácter **/** (por exemplo **`GET / HTTP/1.1`**), deverá ser enviado para o cliente a lista dos ficheiros existentes na directoria

pretendida (caso esta exista):

- o pedido **`GET / HTTP/1.1`** devolverá a lista de ficheiros (com link) existentes na directoria **http_docs**.
- o pedido **`GET /documento/ HTTP/1.1`** devolverá um erro com o código 404: **`HTTP/1.1 404 Not Found`** , visto a directoria **http_docs/documento/** não existir.

Se existir um ficheiro de configuração, tal como descrito na secção 6, a directoria onde se encontram os os ficheiros (DOCUMENT_ROOT) pode ser diferente de **`http_docs`** .

3 Módulo Tratamento de pedido

O módulo tratamento de pedido recebe do **Gestor de Comunicação** um pedido enviado pelo navegador e responde com uma resposta adequada, tal como descrito na Secção 2.0"

Este módulo verifica a existência do ficheiro, verifica se o ficheiro corresponde a um programa executável e verifica o tipo de ficheiro pedido (extensão). Se o ficheiro pretendido existir e for compatível, o seu conteúdo é enviado para o navegador.

3.1 Informação dos pedidos

Sempre que é recebido um pedido, o servidor web deverá escrever no écran a seguinte informação:

- nome do ficheiro solicitado
- código da resposta (200, 400, 404 ou 415)
- endereço do navegador
- tempo necessário ao processamento do pedido e envio da resposta

Exemplo:

```
/                          200  146.193.41.12   10
/documentos/               404  192.168.0.1     4
/documentos/ficheiro1.pdf  200  146.193.41.12   150
/index.html                200  146.193.41.12   15
/imagens/fundo.png         200  146.193.41.12   100
```

```
/exec-cgi                   200  146.193.41.12   200
/imagens/index.html         404  146.193.41.12   4
/documentos/proj.doc        415  146.193.41.12   4
/                           200  146.193.41.12   11
/                           200  146.193.41.12   9
```

4 Módulo Gerador de relatórios

Os relatórios acerca dos acessos às diversas páginas são gerados quando um navegador acede a páginas especiais:

- `GET /estatisticas/Pedidos HTTP/1.1`
- `GET /estatisticas/ClearAll HTTP/1.1`

Nestes casos, código especial deve ser executado. O código responsável pela geração de cada relatório consulta a lista criada no módulo **Armazenamento de pedidos**, formata a informação e envia-a ao navegador.

Ao receber um dos pedidos anteriores, o servidor deverá responder com o código 200, indicar que o resultado é html e enviar os dados formatados em HTML.

Para cada pedido, apresenta-se de seguida a sua descrição e o resultado da sua execução. O resultado é sempre apresentado na janela do navegador.

Exemplo da sequência de pedidos usados nos exemplos seguintes:

```
/                           200  146.193.41.12   10
/documentos/                404  192.168.0.1     4
/documentos/ficheiro1.pdf   200  146.193.41.12   150
/index.html                 200  146.193.41.12   15
/imagens/fundo.png          200  146.193.41.12   100
/exec-cgi                   200  146.193.41.12   200
/imagens/index.html         404  146.193.41.12   4
/documentos/proj.doc        415  146.193.41.12   4
/                           200  146.193.41.12   11
/                           200  146.193.41.12   9
```

4.1 GET /estatisticas/Pedidos HTTP/1.1

Este pedido deverá ser respondido com a listagem de todos os pedidos anteriormente recebidos pelo servidor por ordem cronológica (pedidos mais antigos em primeiro lugar).

Para cada pedido armazenado na lista, deverá ser apresentado:

- Data e hora do pedido;
- Nome do ficheiro pedido;
- Código de erro;
- Endereço do navegador;
- Tempo de processamento e execução do pedido.

Exemplo de resultado enviado para o navegador:

Lista de Pedidos

```
11/2/2014 11:01 /                          200       146.193.41.12   10
11/2/2014 11:02 /documento/                404       192.168.0.1     4
11/2/2014 11:11 /documentos/ficheiro1.pdf  200       146.193.41.12   150
11/2/2014 15:01 /index.html                200       146.193.41.12   15
13/2/2014 10:00 /imagens/fundo.png         200       146.193.41.12   100
13/2/2014 10:00 /exec-cgi                  200       146.193.41.12   200
14/2/2014 23:01 /imagens/index.html        404       146.193.41.12   4
15/2/2014 23:00 /documentos/proj.doc       415       146.193.41.12   4
15/2/2014 23:15 /                          200       146.193.41.12   11
16/2/2014 11:01 /                          200       146.193.41.12   9
```

4.2 GET /estatisticas/ClearAll HTTP/1.1

Este pedido limpa a lista de pedidos, eliminando todos os pedidos aí armazenados.

Envia para o navegador uma mensagem confirmando a limpeza da lista.

Exemplo de resultado enviado para o navegador:

Todos os registos eliminados

Qualquer pedido de estatística posterior não listará os dados eliminados.

5 Módulo Gestor de CGI

A especificação CGI (Comon Gateway Infrastructure) permite um servidor web executar aplicações externas e usá-las como geradores de conteúdo a enviar para os *browsers*. Esta especificação define de que modo os dados são enviados para o **cgi** (script ou aplicação desenvolvida previamente) e de que modo o executável formata o resultado e o envia para o servidor web.

Quando o servidor recebe um pedido cujo ficheiro associado é executável, deverá executá-lo. Se o ficheiro pedido não existir ou não for executável, deverão ser seguidos os passos e verificações descritas anteriormente.

Nesta versão do projecto o **cgi** não recebe nenhum argumento do servidor web (nem do *browser*). O resultado que o o **cgi** deseja enviar para o browser deverá ser escrito no **stdout**. Será responsabilidade do servidor web redireccionar essa informação para o browser. O formato da informação gerada pelo **cgi** é o seguinte:

- primeira linha: deverá conter o tipo da resposta (por exemplo **Content-Type: text/html\n**)
- a segunda linha: deverá estar em brando (só deverá ter o **\n**)
- a partir da terceira linha deverá ser escrito o resultado a enviar para o *browser*.

Deverá ser responsabilidade do servidor (e não do **cgi**) inserir a linha **HTTP/1.1 200 OK\n** antes da resposta do **cgi**.

Se não existir um ficheiro de configuração, qualquer programa executável dentro da **DOCUMENT_ROOT** pode ser considerado um **cgi** e invocado através do *browser*. O ficheiro de configuração poderá limitar a subdirectoria onde se encontram o **cgis**.

cgis que demorem muito tempo a executar deverão ser terminados passados 5 segundos do seu início.

Informação adicional acerca de CGI:

- http://en.wikipedia.org/wiki/Common_Gateway_Interface
- http://httpd.apache.org/docs/current/howto/cgi.html
- http://www.ietf.org/rfc/rfc3875

6 Configurações

Se na directoria do servidor existir um ficheiro de configuração (chamado **www.config**) este deverá ser carregado. Este ficheiro contem as seguintes configurações:

1ª linha: nome da directoria DOCUMENT_ROOT

2ª linha: nome da subdirectoria CGI_ROOT (dentro da DOCUMENT_ROOT) de onde podem ser executados **cgi**

Se este ficheiro existir e estiver correcto, serão estes os valores a usar aquando do acesso a ficheiros e execução de cgi.

Após o carregamento do **www.config**, se se tentar executar um **cgi** fora da directoria **CGI_ROOT** deverá ser enviado para o browser o erro 403:

```
HTTP/1.1 403 Forbidden
```

Se o servidor receber o sinal **SIGUSR1**, o ficheiro de configuração deverá ser lido. Os valores de **DOCUMENT_ROOT** e **CGI_ROOT** deverão ser actualizados.

7 Funcionamento

7.1 Funcionamento paralelo

O servidor web deverá tratar os vários pedidos concorrentemente. Para tal deverão ser usadas técnicas de multi-programação (threads e/ou vários processos).

O servidor deverá ser escalável permitindo um número elevado de pedidos concorrentes.

Os alunos devem a decidir a melhor divisão dos vários componentes pelos diversos processos ou threads de modo a garantir o desempenho mais elevado possível.

O código para a manutenção da lista com a informação dos pedidos deverá ter em conta que será atualizada concorrentemente.

O desenvolvimento do projeto deverá ser guiado desde o início tendo em atenção os requisitos

de desempenho e escalabilidade.

7.2 Gestão de recursos

O servidor deverá executar durante muito tempo sem paragens. Devido a tal, a gestão de recursos (memória, sockets, ficheiros, threads) deverá ser cuidada de modo a garantir que em funcionamento normal esses recursos não esgotam.

7.3 Terminação

A terminação do servidor poderá ser feita através do CTRL-C (no teclado) ou através do envio do sinal SIGTERM. Em ambos os casos deverá ser executado código de limpeza e libertação de todos os recursos.

7.4 Dúvidas e omissões

Em caso de omissões deste enunciado, os alunos podem decidir por uma solução válida, lógica e que não contrarie o escrito neste enunciado.

Em caso de dúvida os alunos devem contactar o corpo docente.

8 Entrega e Visualização do Trabalho

Informações cerca do processo de entrega, avaliação e discussão dos trabalhos serão oportunamente disponibilizadas.