University of St Andrews

CS2003 W02
Web applications: an overview

Özgür Akgün

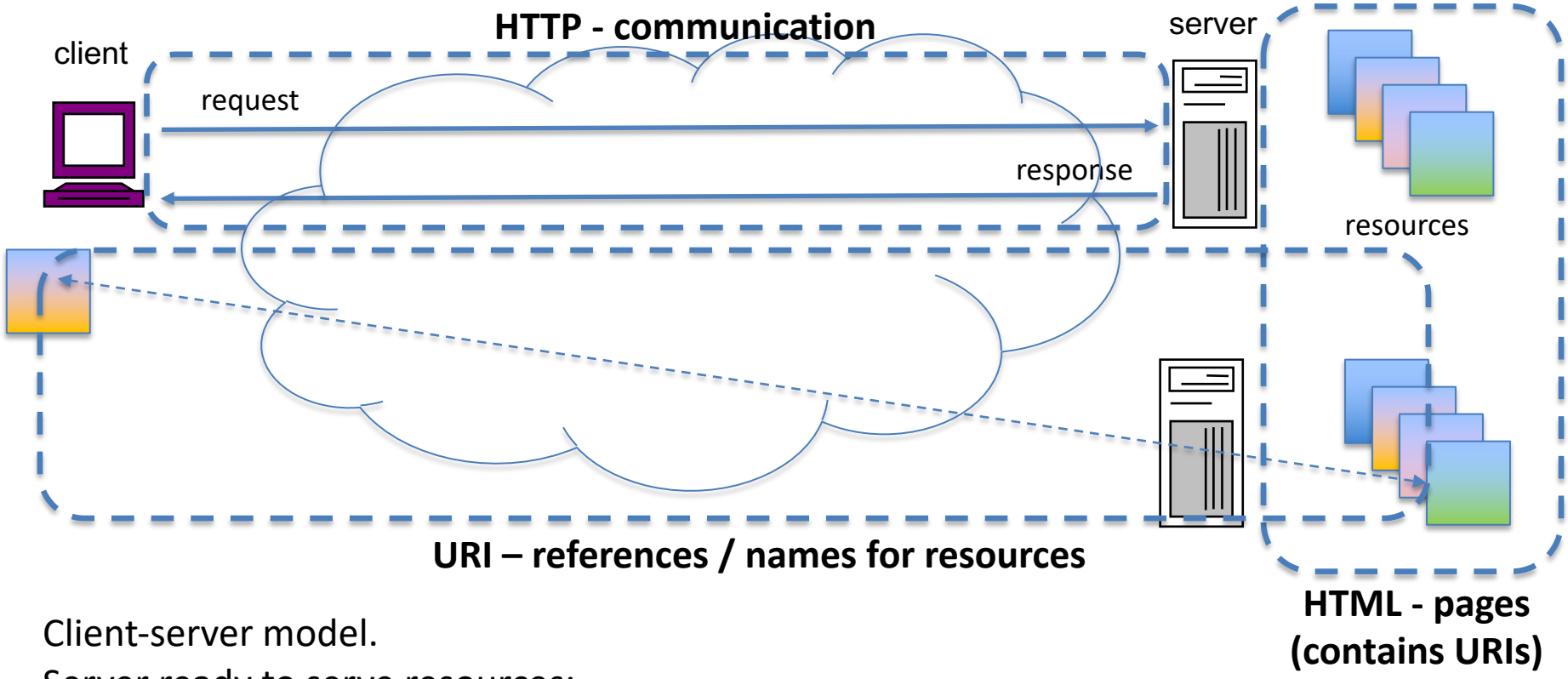
Slides adapted from Saleem Bhatti

A (very) short history of the WWW

University of St Andrews

- No consistent model of access to information:
  - different programs to access information.
  - information in different formats.
  - different ways of creating information for each system.
- Physicists wanted to share info:
  - (Sir) Tim Berners-Lee working at CERN in 1990.
  - **HTML – HyperText Markup Language**:
    define “pages” / “documents” of information.
  - **URI – Uniform Resource Identifier**:
    a reference (name) for a page or other resource.
  - **HTTP – HypterText Transfer Protocol**:
    messages allowing transfer of resources across the Internet.

WWW: overview of operation [1]



- Client-server model.
- Server ready to serve resources:
  - pages, documents, and other objects (e.g. video, documents etc).
- Client instantiated when needed:
  - access via servers.
  - request-response model.

WWW: overview of operation [2]

University of St Andrews

- Client makes request for a resource to a server:
  - identifies resource.
  - makes request.
  - (may be subject to access control).
- Hyperlinks: a page or document can “refer” or “point” to another resource:
  - the resource can be local or remote.
  - the resource may or may not exist!
- A resource is a data object:
  - **content** – originally, **static**, text & images.
  - now, content can be **generated dynamically**.
- Today, many more technologies, e.g. Javascript.

Caching [1]

University of St Andrews

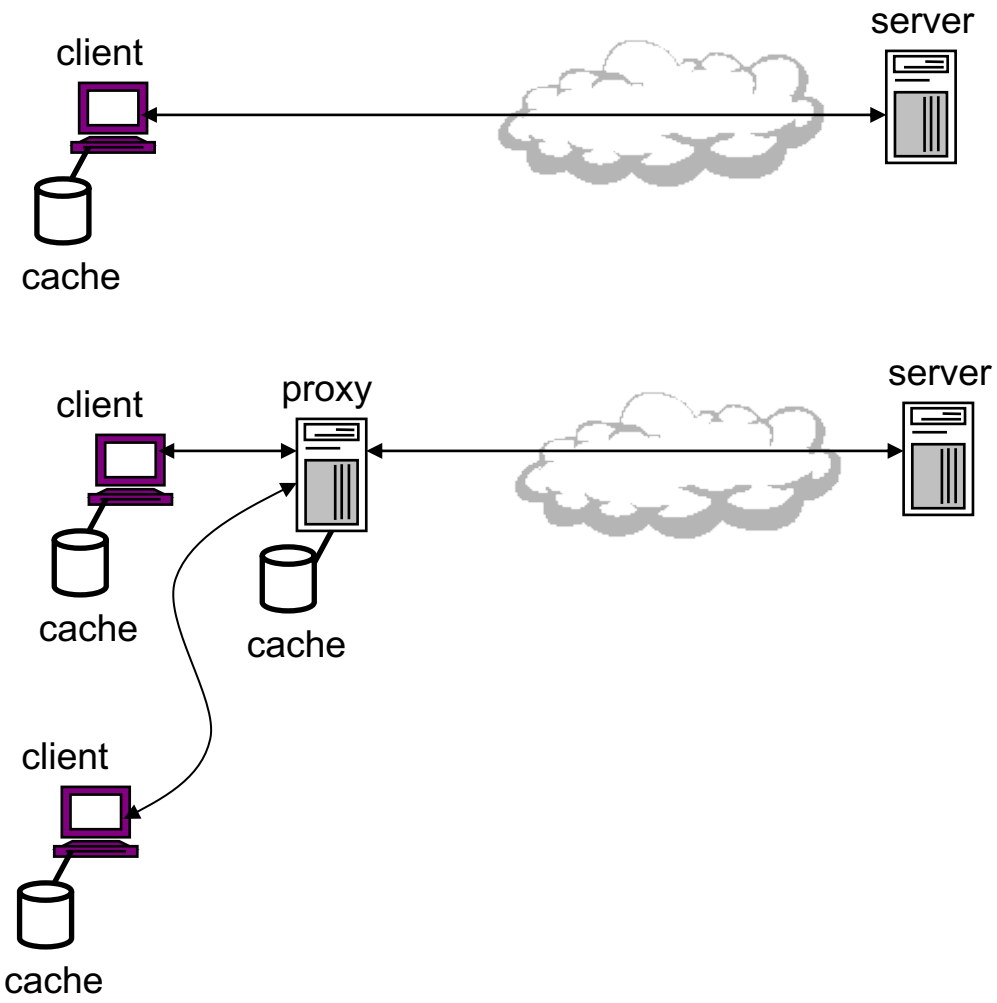
- A **cache** is a performance optimisation tool:
  - a cache is a **local** ("conveniently-placed") copy of data.
- A cache can improve performance:
  - data is more readily available because:
    - (i) it is "closer" to where it needs to be for use (**spatial locality**).
    - (ii) it can be copied to cache ahead of time (**temporal locality**).
- **Any data in the cache is a copy!**
  - if the original data changes, the cached version is "stale"!
  - stale data could perturb the behaviour of the application.
- What if content is **dynamically generated**?

University of St Andrews

Caching [2]

- Need mechanism and protocol to manage cache:
  - "local" storage: **privacy**!
- Should not use stale information:
  - need to check when original has changed.
- Share cached information?
  - improve availability of data.
  - reduce load on network and server.
  - **security and privacy**!
- Use pre-emptive caching?
  - reduce wait times for users.

Caching, computation & processing [1]



- Client and server need to communicate:
  - input from user
  - feedback to user
- Data has to be processed:
  - client side processing
  - server-side processing
  - depends on application
- Proxy does not normally undertake processing:
  - but may be used for site-wide policy.
- **Privacy and Security**!

Caching, computation & processing [2]

University of St Andrews

Client-side

- Local processing , fast response.
- Takes load off server.
- Reduces network load.
- May incur local storage and processing overheads.
- Potential security and privacy issues.

Server-side

- Caching provides benefit for whole site with access to data.
- Common content generated on demand.
- Higher overhead at server for each page/document for content generation.
- Potential security and privacy issues.

Try things out …

First(?) web page - outline

1. Login in to your ‘host’ server
2. Create a web page
   a. Add some text
3. Modify a web page
   a. Add an image
   b. Add a link
4. Try things out

- You will need to use:
  - ssh for remote login from a shell/terminal
  - a text editor (e.g. nano, vim) in the shell/terminal
  - a Web browser (e.g. Firefox) on your desktop machine

ssh

- The lab machines should give you access to your home directory for the School directly.
- If not then you will have to use ssh ...
- ssh (Secure Shell) allows you to remote login to a server where you have an account.
- In the School everyone has a server:

  `username.host.cs.st-andrews.ac.uk`

- So, login in ...

Set up a filespace for the web server

University of St Andrews

- Make sure you are in your home directory:

```
$ cd
```

- Create a directory and sub-directory:

```
$ mkdir nginx_default
$ cd nginx_default
$ mkdir cs2003
$ mkdir cs2003/wk02/
$ cd cs2003/wk02/
```

- Start your editor (e.g. nano, vim):

```
$ vim hello-world.html
```

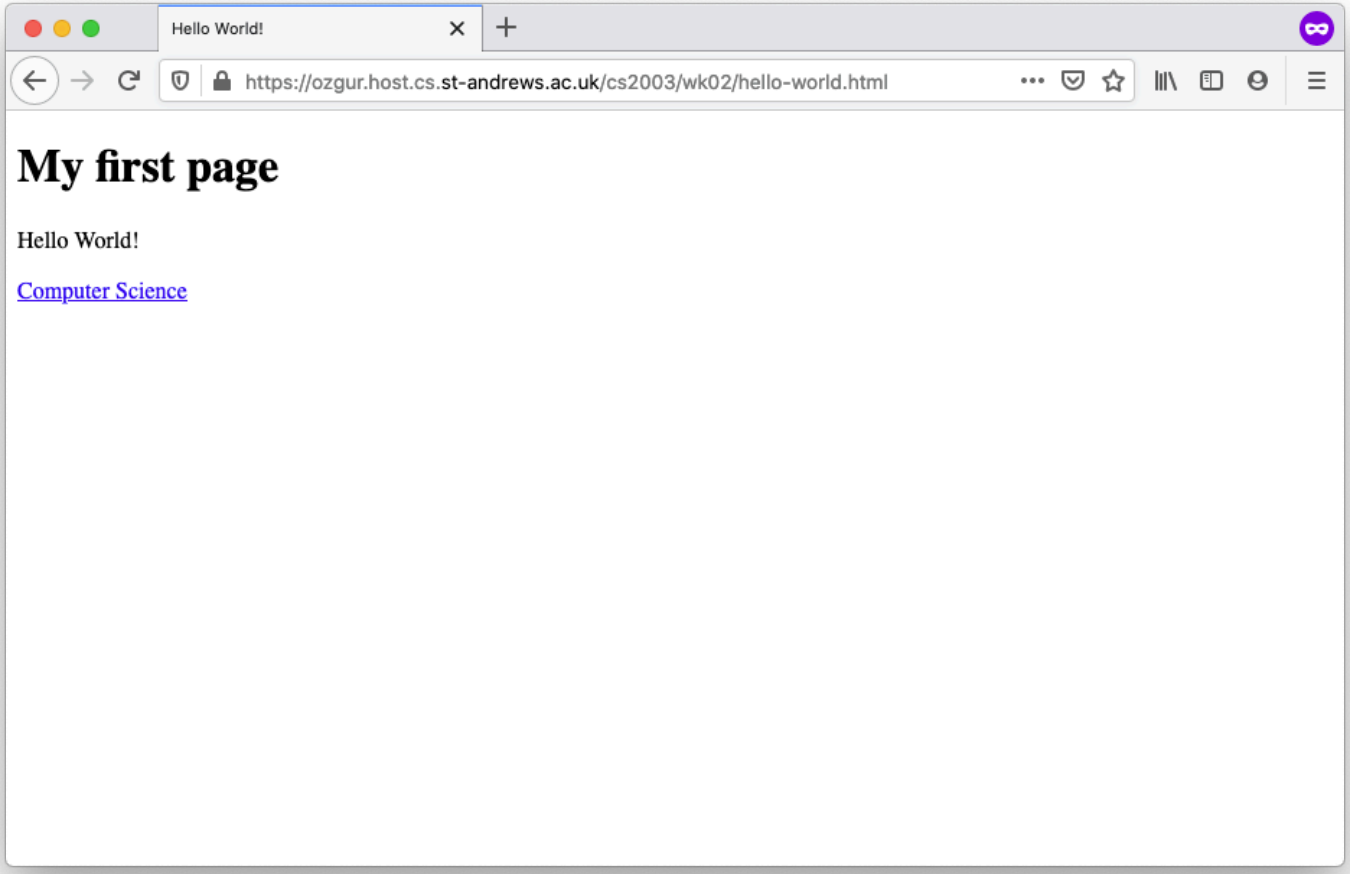
Create a web page ... [1a]

University of St Andrews

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Hello World!</title>
</head>

<body>

<h1>My first page</h1>

<p>Hello World!</p>

<a href="https://www.cs.st-andrews.ac.uk/">Computer Science</a>

<body>

</html>
```

Create a web page ... [1b]

https://username.host.cs.st-andrews.ac.uk/cs2003/wk02/hello-world.html



Hyper Text Markup Language (v5)

University of St Andrews

```
<!DOCTYPE html>
<html>

<head>
<meta charset="UTF-8">
<title>Hello World!</title>
</head>

<body>

<h1>My first page</h1>

<p>Hello World!</p>

<a href="https://www.cs.st-andrews.ac.uk/">Computer Science</a>

<body>

</html>
```

start tag

end tag

attribute

HTML tags

- **HTML tags** describe the **structure** of the page.
- The page must have the following tags:

```
<!DOCTYPE html><html> </html>      "envelope"
<head> </head>                     "header"
<body> </body>                     "content"
```

- I say "must have", but browsers are forgiving ...
- Comments:

```
<!-- begin comment          end comment -->
```

  – comments **do not** nest

Some examples ...

- From studres, copy the sub-directories of CS2003/Examples/ :
  - wk02/01a/
  - wk02/img/
  - wk02/video/

  to the relevant place in your home directory, e.g.:

  ~/nginx_default/cs2003/wk02/
- Have a look at the the .html files in:
  - https://*username*.*host.cs.st*-andrews.ac.uk/cs2003/wk02/