# Software Tools: Web Technologies

Unit Director Dr David Bernhard
Dr Jon Bird

# Overview of the lecture

This is an introduction to the web and explains its relation to the internet.

It gives an overview of some key concepts that you'll need to understand when doing web development.

# The internet



# The internet

The internet is a world-wide computer network that interconnects millions of computing devices throughout the world.

These computing devices include desktops, laptops, tablets, smart phones, Alexa and smart home systems.

All these devices are known as hosts or end systems.

# The internet

End systems are connected by communication links that consist of different types of physical media e.g. coaxial cable, copper wire, fibre optics and radio waves, and these media can transmit data at different rates (bits/second).

Usually end systems are not directly attached to each other by a single communication link, rather they are indirectly connected to each other through intermediate switching devices known as routers.

Routers take information arriving on one of their communication links and then forwards that information on one of their outgoing communication links.

# Two Types of Host: Clients and Servers



A program or machine that responds to requests from others is called a server.

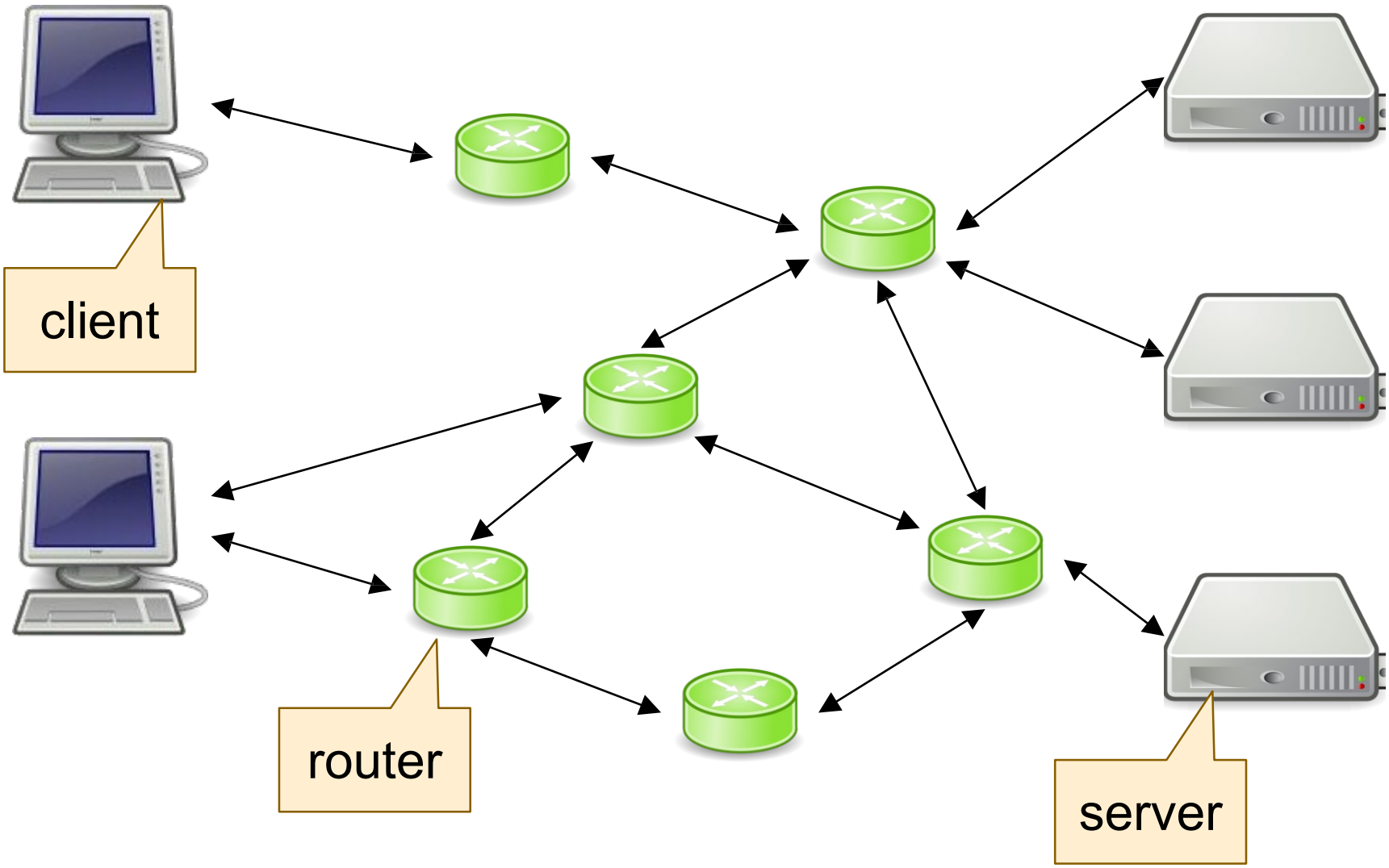
A program or machine that sends requests to a server is a client.

Simple server pseudocode:

```
while (1) {
   req = read();
   resp = serve(req);
   write(resp);
}
```

# The internet





# The internet



The internet is made possible by the development, testing and implementation of Internet Standards which are developed by the Internet Engineering Task Force (IETF) and their documents are known as RFCs (requests for comments).

They are technical and detailed and define protocols such as TCP, IP, HTTP (for the web) and SMTP (email). There are more than 2000 RFCs.

# Protocols



# What is a protocol?

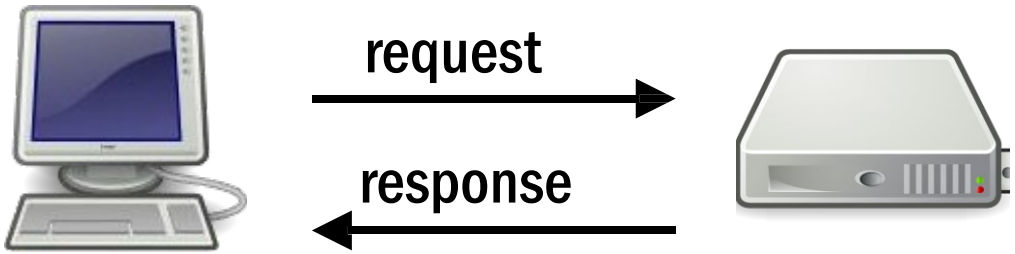


A protocol is an agreement between two computer systems about how they will communicate.

The agreement covers the format of messages that are exchanged and actions that are taken in response to received message or other events.

# Request-response

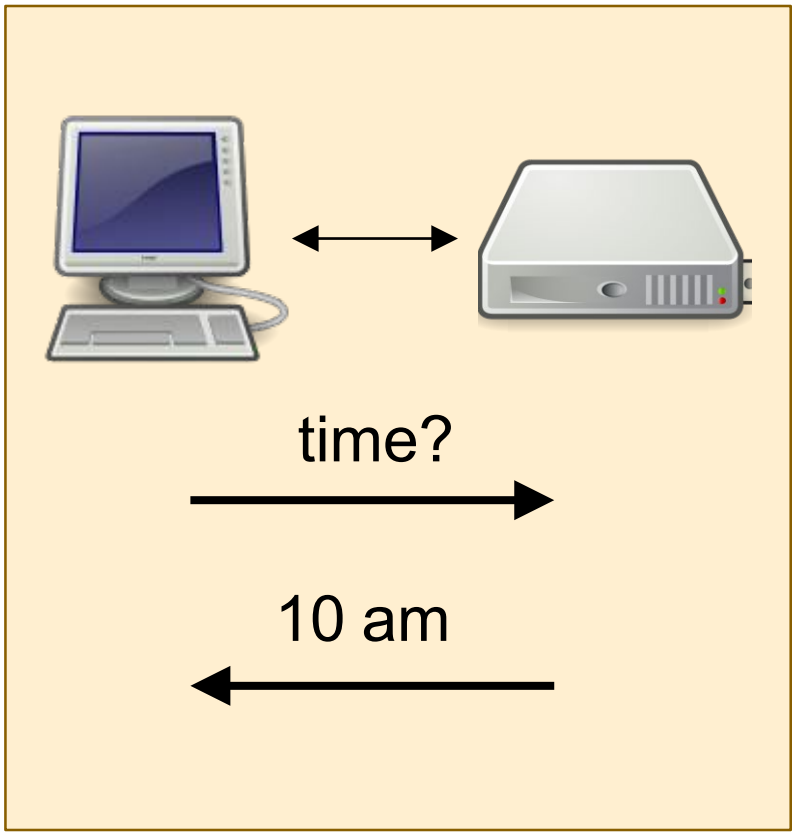





A simple protocol: one machine (the client) sends a message called a request and another machine (the server) replies with a response. This can be repeated.

# Protocols and Data Formats





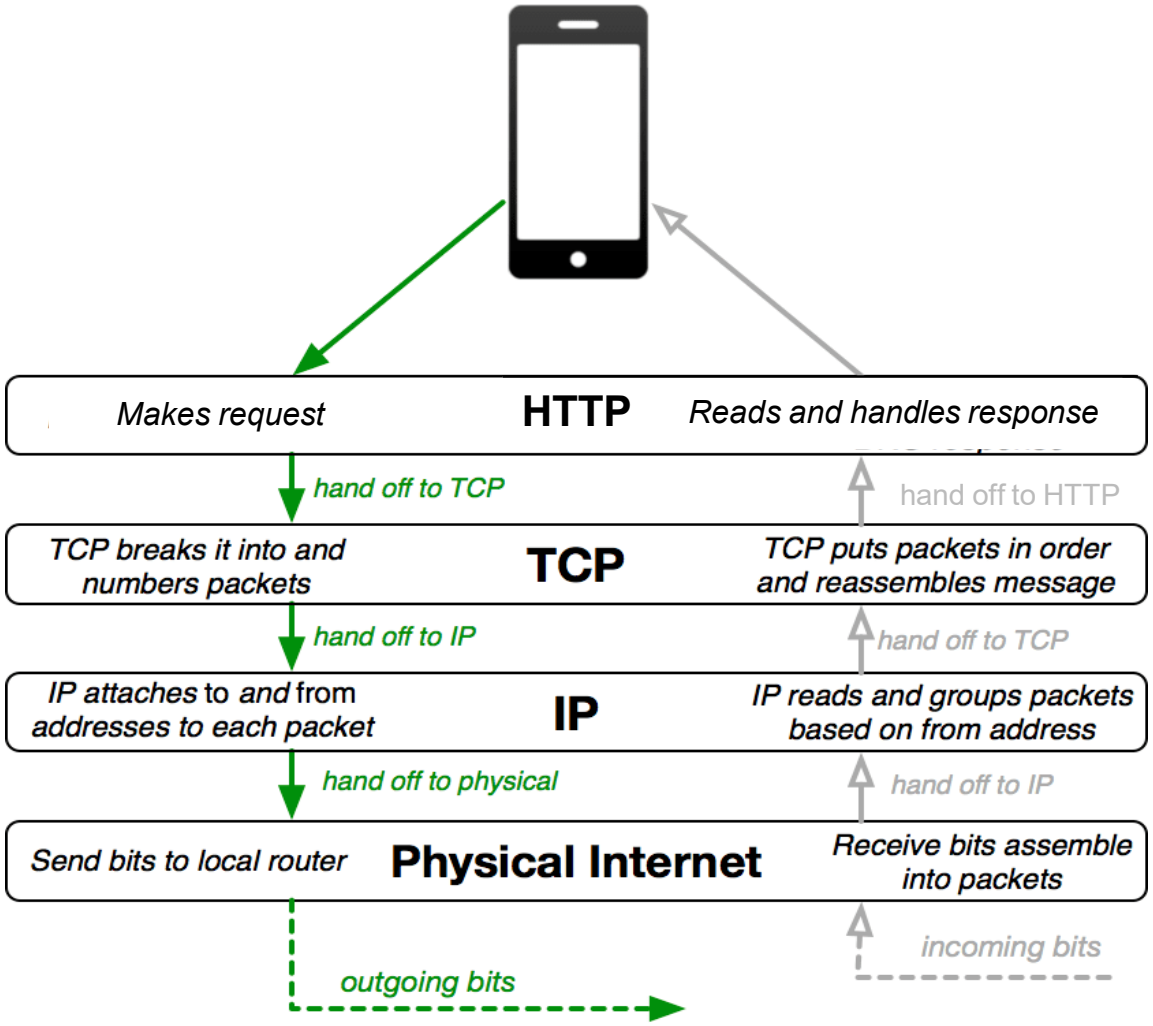

```
<html>
<body>
  <p>The time is:
    <i>10 am</i>.
  </p>
</body>
</html>
```

Protocol

Data Format

# Internet layers






## Application layer

different protocols to support network applications

## Transport layer

transports application layer messages between clients and servers

## Network layer

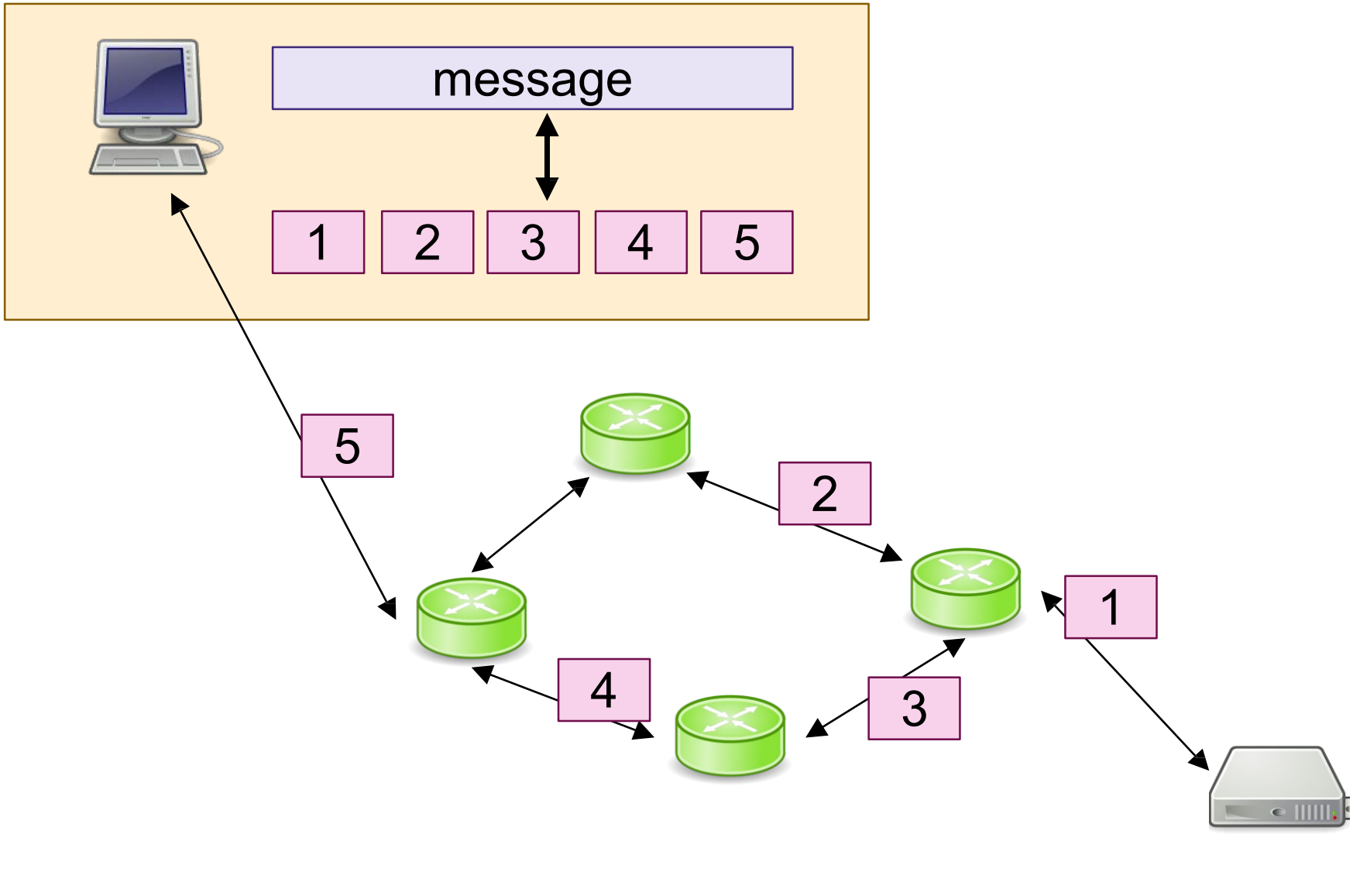
delivers messages based on source and host addresses

## Physical layer

how bits are moved from one router to the next

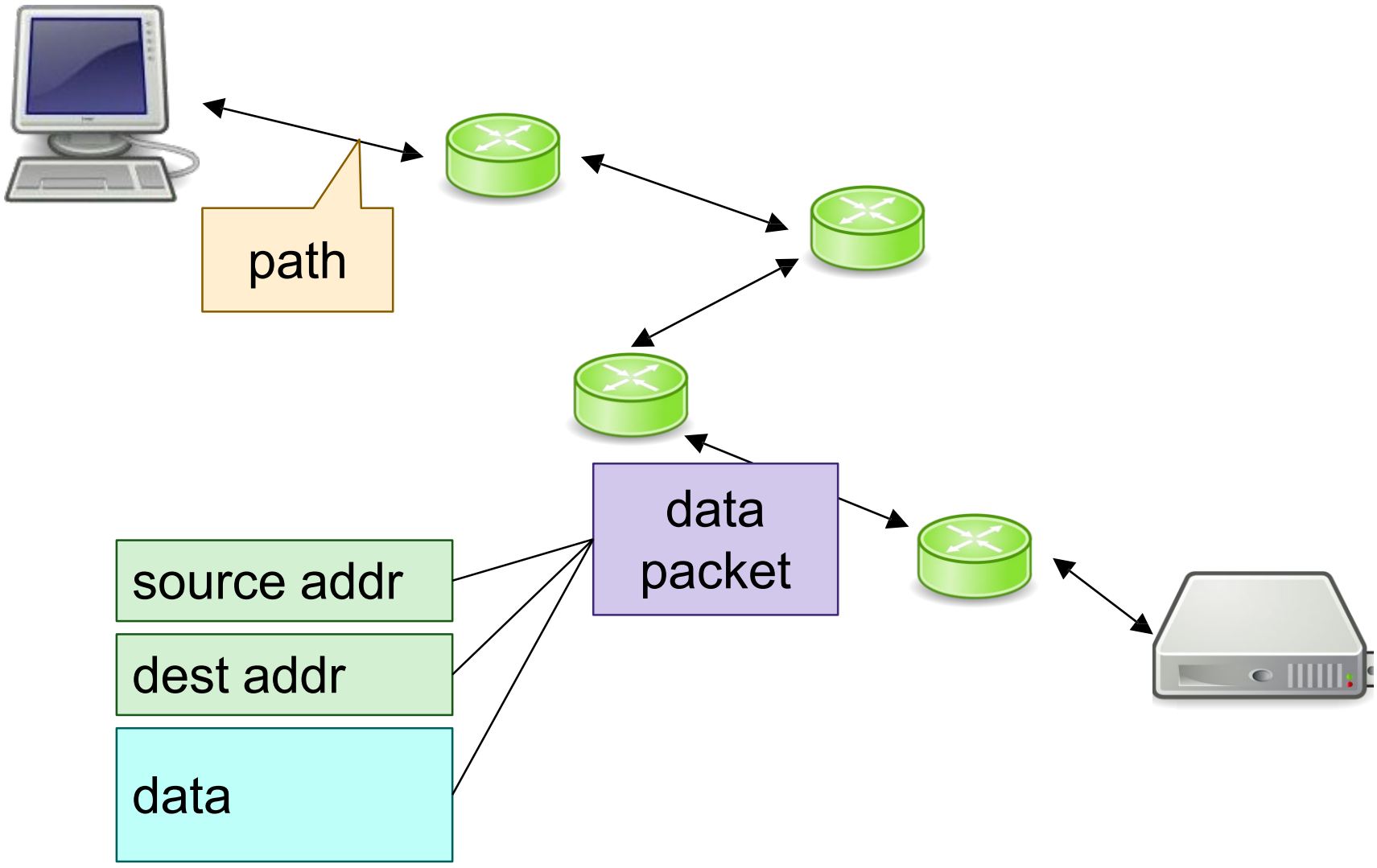
# Transmission Control Protocol





# Internet Protocol Layer





# HTTP



# HTTP



HyperText Transfer Protocol (HTTP)

Nowadays HTTP is the most common application protocol on the internet. There are other protocols for other applications, such as email, file transfer and video streaming.

Version 1.1 is in RFC 7230-7235.

Version 2.0 is out too (RFC 7540) but we'll concentrate on 1.1 for this unit.

# Brief history of HTTP



The HyperText Transfer Protocol was developed between 1989 and 1991 by Sir Tim Berners-Lee when he was working at CERN. Initially, it was designed to exchange files in a laboratory environment.

Berners-Lee proposed building a hypertext system over the internet using the existing TCP and IP protocols.

# Brief history of HTTP



The four building blocks of Berners-Lee's proposal were:

A text format to represent hypertext documents: HyperText Markup Language (HTML)
A protocol to exchange hypertext documents: Hypertext Transfer Protocol (HTTP)
A client to display hypertext documents: the first browser
A server to give access to the documents

# Brief history of HTTP



The requests were very simple and consisted of a single method followed by the path to the hypertext document:
GET /mypage.html

The response was also very simple and just consisted of the file:
<HTML>
A very simple HTML page.
</HTML>

# Brief history of HTTP



Only HTML files could be requested.

There were no status or error codes – any problems were described in the HTML file that was sent back.

As the client was connected to the server there was no need to specify its address nor to state the protocol being used.

# CRUD and HTTP Methods



CRUD is an acronym for basic operations that can be carried out on data (aka resources): create; read; update; delete

**create**

The create operation creates a new resource in a server assigned location. The create operation is typically performed by an HTTP POST method.

**read**

The read operation accesses the current contents of a resource. The operation is performed by an HTTP GET method.

**update**

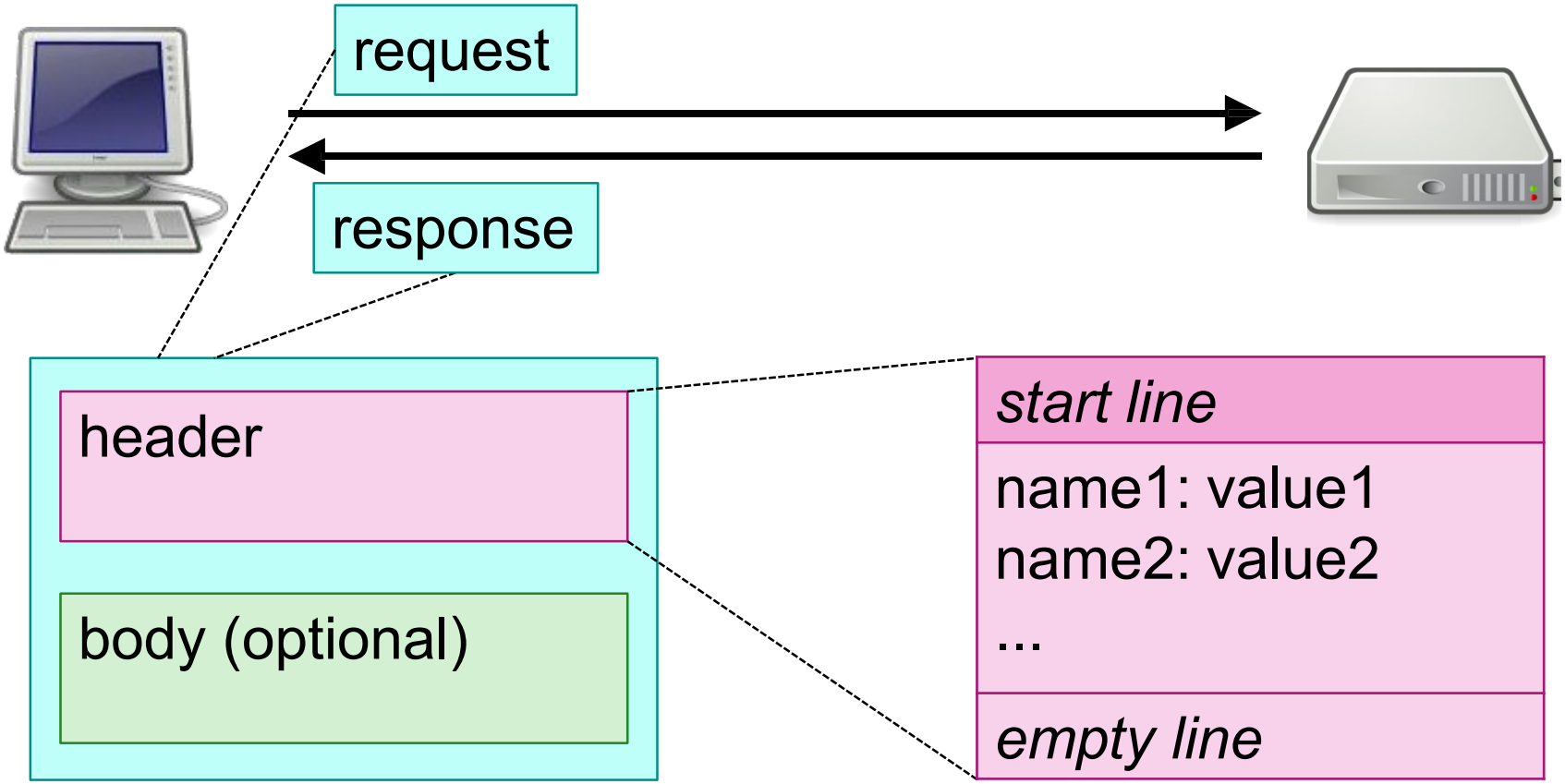
The update operation creates a new current version for an existing resource or creates a new resource if no resource already exists for the given id. The update operation is performed by an HTTP PUT method.

**delete**

The delete interaction removes an existing resource. The interaction is performed by an HTTP DELETE method.

# Structure






HTTP is line-based (lines end with CR LF).

# Request



method

GET /home/bernhard/personal.html HTTP/1.1

Host: www.cs.bris.ac.uk
Connection: close
...

*empty line*

The ‘Host’ parameter is required in HTTP 1.1.

# HTTP response status codes



| | | |
|---|---|---|
| 1xx information | 100 continue | ... |
| 2xx success | **200 OK** | 202 Accepted |
| | 201 Created | ... |
| 3xx redirect | 301 moved permanently | |
| | ... | |
| 4xx client error | 400 bad request | **404 not found** |
| | 403 forbidden | ... |
| 5xx server error | **500 internal** | ... |

# Content-type



Content-type: text/html; charset=UTF-8

Content-type (aka MIME or media type, but not in HTTP)

text/html, text/plain, text/css, text/javascript, image/jpeg, application/pdf, video/mp4,

application/octet-stream

For a browser, the HTTP response content-type  overrides any information in the resource itself.

# URLs



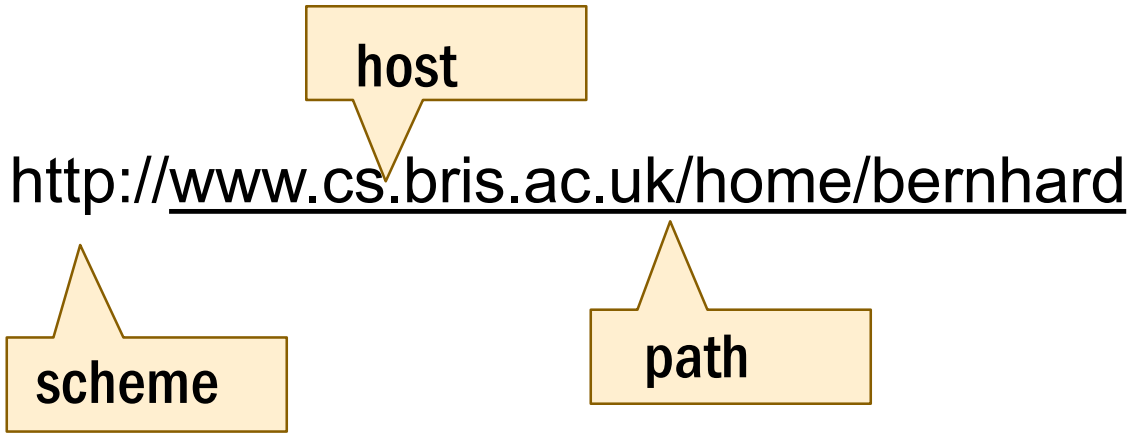
# URLs



URL stands for  Uniform Resource Locator (RFC 3986)

A resource is some media type such as a web page, image, file or video.

A URL tells you where a resource is located – aka its “address”



# Scheme (aka protocol)



https://www.cs.bris.ac.uk/home/bernhard

https is http with extra security provided by the TLS protocol.

# Host



https://www.cs.bris.ac.uk/home/bernhard

Every machine on the internet has an IP address, e.g. 137.222.0.38.

However, that's not very human-friendly.

A host (domain) name is a sequence of words separated by dots.

# Addresses



IPv4 (RFC 791):

address = 4 bytes, e.g. 137.222.0.38

IPv6 (RFC 2460):

128 bit addresses

# Path



https://www.cs.bris.ac.uk/home/bernhard

The format and meaning of the path is mostly up to an individual server.

In the early days of the web, all servers did was serve files and the path component was simply the path to a file starting at a particular folder.

Nowadays, there are several conventions for different kinds of applications.

# Query (aka parameters)



https://www.cs.bris.ac.uk/home/bernhard

?name=David&action=view

When you fill in a form on a website, how does the data get to the server?

There are several ways to do this; one is to put the data in the URL (if using the GET method).

The query part of a URL is optional.

It starts with a '?' character and contains name=value entries separated by '&' characters.

# Key concepts introduced



The internet

Client

Server

Router

The (world wide) web

Protocols, in particular HTTP

URL

# Excellent resource for web development



An excellent and detailed resource for information about web technologies is:

https://developer.mozilla.org/en-US/docs/Web