Notes- Internet and Web

Web Programming (F28WP)

Introduction

Supplementary notes to complement the lecture material for the web programming course.

Nature of Internet
Internet is an inter network of Wide Area Networks

- with gateways between them
- based upon common use of TCP/IP protocols
- supporting standard application services - DNS, e-mail, web

Internet is organised by IETF, ICANN and ISOC.
Internet has no central operations room or global operations manager.
Each Internet host has an IP address e.g. 137.195.13.48.
Some Internet hosts have a domain name e.g. www.macs.hw.ac.uk.
TCP/IP Networks can be viewed as 4 layer structure:

Layer	Functionality
application	interfaces directly with user applications or users
transport	end to end (un)reliable deliver y of TCP or UDP packets
network	IP datagram deliver y, addressing, routing
link	delivers frames, handles errors, drives physical transfers

Examples of protocols on these Internet layers are:

application:	SMTP, IMAP, HTTP, DNS, RTP, SNMP, TFTP
transport:	TCP, UDP, SCTP
network:	IP, ICMP, IPsec, IGMP
link:	Ethernet, 802.11, DSL, ARP, L2TP, ISDN, GPRS, PPP

TCP carries reliable services - email (SMTP, IMAP), web (HTTP).
UDP carries best efforts services - DNS, media streams (RTP).

IETF, ICANN and Internet History
Inter net Engineering Task Force (IETF) started in 1986 and is

- open group of network designers, operators, vendors, researchers
- concerned with evolution and operation of Internet
- major developer of international IT standards

IETF does its work in 100+ chartered working groups which

- debate new standards via mailing lists and at IETF meetings
- produce documents - RFCs, Inter net-Drafts etc.

Internet Corporation for Assigned Names and Numbers or ICANN

- controls IP addresses, domain names, protocol parameters
- supervises root server system of DNS
- is not for profit US corporation founded in 1998

MIME Content Types

Media Types are classified by MIME scheme for Internet resources.

Initial list of official types was given in RFC 1521 in 1993 including:

Major Type	**Subtype**	**Description**
application	*octet-stream*	uninterpreted byte sequence
audio	*basic*	audible sound
image	*gif*	still picture in GIF for mat
	jpeg	still picture in JPEG for mat
message	*external-body*	message itself must be fetched over net
	rfc822	MIME RFC 0822 compliant message
multipart	*alter native*	same message in different for mats
	digest	each part is RFC 5322 message
	mixed	independent parts in specified order
	parallel	parts must be viewed simultaneously
text	*plain*	unformatted text
	richtext	text including for matting commands
video	*mpeg*	movie in MPEG for mat
	postscript	printable document in PostScript

3 major types have been added more recently

model RFC 2077 in 1997 e.g. model/vrml *example* RFC 4735 in
2006 e.g. example/hyper media *font* RFC 8081 in 2017 e.g.
font/woff

IANA maintains a large current list of recognised MIME (sub)types.

Web uses 2 MIME types for HyperText Markup Language *plain HTML* text/html

XHTML application/xhtml+xml, text/html

Uniform Resource Identifiers

Uniform Resource Identifier (RFC 3986) is

- compact sequence of characters (alphanumerics & a few symbols)
- identifier for online or offline abstract/physical resource

Uniform Resource Locator is a kind of URI that

- represents *location* and *access method* of resource
- has general form *<scheme> : <scheme-specific-part>*

RFC 1738 specifies URL formats for following access methods

file host’s file system *ftp* file transfer protocol
http hypertext transfer protocol *telnet* protocol

which share common syntax for scheme-specific part after //

```
<user>:<password>@<host>:<port>/<url-path>
```

where some or all of following parts may be excluded

```
<user>:<password>@ :<password> :<port> /<url-path>
```

Common syntax also applies to *mailto* and *news* URIs.

RFC 1738 regards some characters as *unsafe* for use in URLs

```
space < > " # % { } | ^ ~ [ ] `
```

They should be replaced with 3 octets *% <hex> <hex>*.

HTTP URLs give following characters a special meaning:

```
; / ? : = &
```

Only these non-alphanumeric symbols may occur unencoded:

```
$ - _ . + ! * ' ( ) ,
```

URI Access Methods

Since RFC 1738 newer URI access methods have been devised *data* immediate data in URL itself - RFC 2397 *dns* dns:[//dns-ser ver-host[:por t]][/domain] - RFC 4501 *geo* geographic location - RFC 5870 *ldap* LDAP directory protocol access - RFC 2255 *nfs* remote nfs file system - RFC 2224 *rtsp* Real Time Streaming Protocol - RFC 2326 *sms* shor t message service - RFC 5724 *tel* voice access via telephone - RFC 2806 *ws* websocket connection - RFC 6455 *xmpp*Jabber instant messaging - RFC 5122

Also var ious other proposed/proprietar y URL schemes exist *callto* callto:+int-area-local - callto URL using dial number *irc* irc://ser ver :port/chatroom - irc: URL scheme *rmi* rmi://[host][:por t]/[object] - RMI URL

Some out of date access methods include:

gopher Gopher distributed hyper media system - RFC 1738 *prospero* Prospero Directory Ser vice - RFC 1738 *wais* Z39.50 based Wide Area Infor mation Ser ver - RFC 1738 IANA has registry of URI schemes - permanent, provisional, obsolete.

Forms of URL

Scheme	Port	Common Form
data	-	data:<mediatype>[;base64],<data>
file	-	file:///<path>
ftp	21	ftp://<user>:<password>@<host>:<por t>/<path>
geo	-	geo:<latitude>,<longitude>
http	80	http://<host>:<port>/<path>?<searchpar t>
news	119	news://<host>:<por t>/<newsgroup-name>/<ar ticle>
sms	-	sms:+<phonenumber>?body=<message>
telnet	23	telnet://<user>:<password>@<host>:<por t>

Data URL encoding "No more secrets" in base64:

data:text/plain;base64,Tm8gbW9yZSBzZWNyZXRzCg==

Unix File URL for Iain McCrone's handler file for mimetypes file:///home/imcc/.mailcap

Geo URL specifying location on west coast of Scotland near Oban geo:56.3192,5.5825

HTTPS URL getting web hits for "og" on search engine Yahoo!

https://search.yahoo.com/bin/search?p=og

News URL for news articles in Dr Who news group news://news01.khis.de/uk.media.tv.sf.drwho

Set up SMS message to send to phone number by default service sms:+441314513427?body=Hello%20there

Telnet URL to Shadow Lands Multi-User Dungeon site telnet://mush.elendor.net:1893/

HyperText Transfer Protocol

HyperText Transfer Protocol (HTTP) is the web’s application protocol.

Key features of design of HTTP service include

client server	provider only delivers service to consumer *on demand*
stateless	each request is *standalone* with no context
synchronous	client connects to server, requests and waits for reply
caching	*recent stored* replies may be used to answer requests
layered	requests may be relayed via (chain of) intermediaries

Typical client-server interaction on web might be

- link *http://www.macs.hw.ac.uk/˜air/index.html* is clicked
- browser gets IP address of *www.macs.hw.ac.uk* from DNS
- browser makes TCP connection to port 80 on *137.195.13.48*
- browser sends *HTTP* request for given URL
- *www.macs.hw.ac.uk* server sends file *index.html*
- browser gets files referenced in page by same connection
- TCP connection is released
- browser renders *index.html* with extras - CSS, images etc.

HyperText Transfer Protocol

RFC 1945 defines HTTP 1.0. RFC 7230, 7231 define HTTP 1.1.

HTTP messages are in Internet E-mail for mat with MIME extensions.

HTTP 1.1 supports following request methods

CONNECT	method for use with SSL tunnelling
DELETE	remove web page
GET	fetch web page
HEAD	fetch HTTP header metadata on web page
OPTIONS	request data on comms options to given URI
POST	handle submitted web page with given URI
PUT	store submitted web page as given URI
TRACE	request remote loop-back of request message

Only the *GET*, *HEAD* and *POST* methods are widely used.

Ever y HTTP request gets status code response of following types:

Type	**3 Digits**	**Description**
informational	**1xx**	request is being processed
successful	**2xx**	request was received, understood, accepted
redirection	**3xx**	further action is needed to complete request
client error	**4xx**	syntax error in request
server error	**5xx**	server can't fulfil valid request

Common replies - 200 (OK), 401 (Unauthorized), 404 (Not Found).

RFC 7450 (May 2015) specifies HTTP's latest version HTTP 2.

HTTP 2 provides faster message syntax with same HTTP semantics.

GET

GET interaction between *Inter net Explorer 10* and Google UK

```
GET http://www.google.co.uk/ HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: en-GB
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1;
WOW64; Trident/6.0)
```

```
Accept-Encoding: gzip, deflate
Proxy-Connection: Keep-Alive
Host: www.google.co.uk
Pragma: no-cache Cookie:
PREF=ID=3ec33f874a5b2596:U=c6 ...

HTTP/1.1 200 OK
Date: Fri, 26 Jul 2013 14:17:28 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=UTF-8
Content-Encoding: gzip
Server: gws
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Transfer-Encoding: chunked

77a5
...
```

HTTP 1.1 request supplies cookie and gets compressed response

- *Accept-Encoding:* field permits gzip compressed reply
- *Content-Type:* field says reply is UTF-8 encoded HTML
- *Content-Encoding:* field says reply is gzip compressed
- *Transfer-Encoding:* field says reply is in hex sized chunks

POST

Filling in "Donald Trump" in simple HTML for m and submitting it

```
<form action="/cgi-bin/cgiwrap/hamish/na/test"
      method="post">
  <p> Name <input name="user" size="12">
      <input type="submit"> </p>
</form>  submits for m details by POST method much like the following:
```

```
host% nc -C www2.macs.hw.ac.uk 80
POST /cgi-bin/cgiwrap/hamish/na/test HTTP/1.1
Host: www2.macs.hw.ac.uk:80
Content-type: application/x-www-form-urlencoded
Content-length: 17

user=Unsername
HTTP/1.1 200 OK
Date: Wed, 18 Oct 2017 16:38:14 GMT
Server: Apache/2.2.15 (CentOS)
Transfer-Encoding: chunked
Content-Type: text/html
```

```
91
<!DOCTYPE html>
<html>
<head><title>CGI Inputs</title></head>
<body>
<h3>CGI Inputs</h3>
<ul>
<li>user = Username</li>
</ul>
</body>
</html>
```

"91" is size of chunk it precedes expressed as a *hexadecimal*.

Size and number of chunks in *chunked transfer* are up to server.

POST request should be used instead of GET request when

- parameter details passed to web server are quite large
- browser can’t safely repeat request (e.g. e-purchases)

HTTP lets browser repeat GET but not POST without user say so.

Representational State Transfer

Roy Fielding argues networked systems exhibit architectural styles: *mobile code* moves to close gap between processing and data *pipe and filter* inputs are filtered to produce transformed outputs

Roy Fielding asks *What is the architectural style of web applications?*

He analyses *well designed* web application as

- *network* of web pages
- user progressing through application by *selecting* links
- link selections cause *fetching* and *render ing* of next page

Under lying architectural style is of

- network of web pages or *virtual state-machine*
- link selections causing *state-transitions*
- transitions causing *transfer* of next *state* of application

Fielding calls this style *REpresentational State Transfer* or **REST**.

Web's REST style uses

network protocol	HTTP
address scheme	URIs - HTTP URLs
resource representations	XML, HTML, PNG etc.
resource types	MIME types - text/xml, text/html, image/png
hyper links	XML's xlink attributes, HTML link tags

REST Architectural Style

To *Null* style Fielding's REST model for Web adds 6 more:

Client Server *server* fulfils *client* requests only on demand

Stateless

communication is stateless in nature

server doesn't look beyond request to know its details

Cache

local copies of replies are *saved* from previous requests local copies can be *reused* to answer further requests

Uniform Interface

uniform interface is presented

generic CRUD operations supported for interface

Layered System *inter mediate* components may handle and return replies
they may *relay* requests or process them from *cache*

Code-On-Demand clients *download* and *execute* scripts or applets optional constraint of REST style

Pros and **Cons**

Client Server

efficient needs driven use of resources

inept at supporting *push services* like event notification

Stateless

improves *visibility*, *reliability* and *scalability* makes sessional use *awkward* - cookies

Cache improves *performance*, reduces *use of bandwidth*
may decrease *reliability* if resource copies are stale

Uniform Interface eases use by different clients

can decrease *efficiency* as data is passed in standard form

Layered System improves *scalability* and *robustness* can decrease

efficiency in worst cases

Code-On-Demand offloads processing from server to clients on demand increases client user's *security risks*

REST supports generic CRUD operations (*Create*, *Retrieve*, *Update*, *Delete*) on resources via uniform interface.

History of the Web

Web is steered by 2 key organisations

W3C	World Wide Web Consortium
IETF	Internet Engineering Task Force

IETF defines HTTP and URIs. Other web standards belong with W3C.

Tim Berners-Lee (TBL) is the inventor of the World Wide Web.

Seminal early years in the history of the Web include:

1945	Vannevar Bush's paper "As We May Think"
1965	Ted Nelson coins word *HyperText* at ACM conference
1980	TBL implements single host, shared user hypertext app *Enquire*
1989	TBL proposes distributed hypertext system for CERN
1990	invention of *WorldWideWeb* name early work on HTTP and hyper linked document support
1991	HTML is developed based on SGML line mode browser and HTTP server in use at CERN
1992	Erwise and Viola GUI HTML browsers run under X Window
1993	NCSA Mosaic web browser developed by Marc Andreessen CERN announces WWW technology is freely usable by anyone 200+ known HTTP servers in October
1994	Andreessen et al leave NCSA to form what became Netscape World Wide Web Consortium founded on October 1 1500+ known HTTP servers in June
1995	HTTP traffic takes over from FTP as top Internet app RFC 1866 - HTML 2.0, T Berners Lee and D Connolly
1996	RFC 1945 - HTTP/1.0, T Berners Lee, R Fielding, H Frystyk Today 1.7 billion web servers are in use on the Internet.

Distributed hyper media system Gopher was superceded by Web.

Things To Do

Read "As We May Think"
Vannevar Bush, Atlantic Monthly, May 1945

Read Chapter 1, The Web: An Over view
Dynamic Web Programming and HTML5 Paul S Wang,
CRC Press, 2013.

Read A Little History of the World Wide Web
Robert Cailliau, W3C Consortium, 1995.

Key Points

○ A Uniform Resource Identifier is a compact sequence of characters (alphanumerics and a few symbols) that identifies an online or offline, abstract or physical resource. A Uniform Resource Locator is a type of URI that identifies a resource through an access method and location.

○ HTTP is a stateless, synchronous, readable, client server application protocol running on top of TCP, supports GET operations to fetch resources and POST operations to invoke services, and can operate through intermediaries and use caching.

○ The Web was invented by Tim Berners Lee around 1990, replaced Gopher from 1994 onwards as the Internet's foremost distributed hyper media application, and has grown to support well over a billion active web servers.