This talks provides a history of the Internet and describes the main protocols. It summarizes:

- [RFC 2616](#) HTTP Protocol 1.1.

1 Timeline

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1970</td>
<td>Work on Unix begins.</td>
</tr>
<tr>
<td>1971</td>
<td>First FTP and Telnet implementations.</td>
</tr>
<tr>
<td>1973</td>
<td>TCP development begins.</td>
</tr>
<tr>
<td>1983</td>
<td>BSD Unix ships TCP-IP stack.</td>
</tr>
<tr>
<td></td>
<td>TCP-IP becomes a government standard.</td>
</tr>
<tr>
<td>1990</td>
<td>Object Linking and Embedding (OLE).</td>
</tr>
<tr>
<td></td>
<td>COM provided infrastructure.</td>
</tr>
<tr>
<td>1991</td>
<td>CORBA 1.1 released.</td>
</tr>
<tr>
<td>1992</td>
<td>Internet opened to commercial traffic.</td>
</tr>
<tr>
<td>1995</td>
<td>CORBA 2.0 released. IIOP standardized.</td>
</tr>
<tr>
<td>1996</td>
<td>COM is renamed ActiveX.</td>
</tr>
<tr>
<td>1997</td>
<td>Java RMI ships (JDK 1.1).</td>
</tr>
<tr>
<td>1998</td>
<td>ActiveX is renamed as visual controls.</td>
</tr>
<tr>
<td></td>
<td>COM is renamed as components.</td>
</tr>
<tr>
<td></td>
<td>DCOM appears.</td>
</tr>
<tr>
<td>2000</td>
<td>Microsoft announces .NET initiative.</td>
</tr>
<tr>
<td></td>
<td>Chooses SOAP for distributed programming.</td>
</tr>
</tbody>
</table>

- A more detailed [timeline on components](#).

Note:
The history of the Internet is interlaced with the history of Unix as it was on Unix that all the early tools (programs) for inter-machine communication were developed. First we saw tools for transferring data from one machine to another as that was the most pressing need. Once you could transfer data, it was a simple matter to make this data a text file and call it email. Programs first began to talk to each other using Remote Procedure Calls (RPC). RPCs are a simple extension on functional languages which, at that time, were dominant. As object-oriented languages (OOLs) became popular, with the advent of C++, the programmers needed a way to do what RPC did, but with an OOL. This need ushered lead the development of CORBA. Later on, Java re-implemented the same ideas, but in a simplified manner, in Java RMI.

Microsoft’s OS was only used in low-end personal computers which were rarely networked. As such, it largely ignored these developments. However, Microsoft faced another problem, it needed to develop a way for many programs to share the same functionality without having it replicated in each program. That is, it needed shared libraries which they named Dynamic Link Libraries (DLL).
DLLs were used to create components which were tied together by following specific guidelines, a process dubbed Object Linking and Embedding (OLE). OLE went thru several revisions and was then renamed COM. COM is a component model which allows one component to be used by other programs running in the same machine. Microsoft realized that they could add some networking infrastructure and distribute the invocation of component, and idea that gave rise to Distributed COM (DCOM). DCOM is very complicated to learn and many programmers shun it. In 2000 Microsoft announced the .NET initiative which replaces all the functionality of DCOM with a much simpler system based on open standards. We will talk about these later in the semester.

2 Influences

- In the beginning there was the Internet. In created a desire for computers to communicate and so TCP-IP was born.
- Everything we will study builds on top of TCP-IP.
- Object Oriented Programming gave rise to distributed objects (RMI, CORBA).
- Components (re-use deployed code) gave rise to COM, which was then distributed among machines (DCOM).
- AI research gave rise to agents, which use distributed objects.
- The web (port 80), XML, OOP, and the need to communicate gave rise to SOAP.

Note:
Meanwhile, Artificial Intelligence researchers had been trying to build machines that could achieve various tasks which were deemed to require intelligence, for example, playing chess, recognizing handwriting, etc. Once the machine could perform the task then everyone changed their mind and decided that the task did not, in fact, require intelligence (but, I digress). One of the tasks that has yet to be completely conquered is that of getting autonomous agents to coordinate with each other in order to solve a particular problem. This coordination is the focus of multiagent researchers, like myself. Of course, these agents are often software agents (although they can also be robots) which inhabit the Internet. As such, it was obvious that they should use the distributed programming libraries developed by others.

Agents operate at a higher abstraction level than most of the protocols that we will be discussing in this class. Still, as we shall see towards the end of the semester, the tools being developed for the Semantic Web are merely re-statements of technologies developed for multiagent systems.

3 Abstraction Layers

<table>
<thead>
<tr>
<th>Abstraction Layer</th>
<th>Application.</th>
<th>Application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport.</td>
<td>TCP, UDP.</td>
<td>TCP, UDP.</td>
</tr>
<tr>
<td>Internet.</td>
<td>IP.</td>
<td>IP.</td>
</tr>
<tr>
<td>Host-to-network.</td>
<td>Ethernet, FDDI.</td>
<td></td>
</tr>
</tbody>
</table>

- The Open Systems Interconnection (OSI) 7-level model is overkill since many layers collapse into one or don’t exist.
- **Host-to-network** defines how a particular network interface (Ethernet card, PPP connection) sends IP datagrams over its physical connection. Examples: Ethernet, Token Ring, LocalTalk.
- A **gateway** can convert from one interface to another.

Note:
In this class we will focus on the application layer.
4 TCP/IP

- The Internet Layer breaks data into datagrams. Examples: IP, Appletalk, NetBEUI.
- Each datagram (packet) is sent independently of the others and travels the Internet on its own path.

<table>
<thead>
<tr>
<th>0</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>20</th>
<th>24</th>
<th>28</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Header Length</td>
<td>Type of Service</td>
<td>Datagram Length</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identification</td>
<td>Protocol</td>
<td>Flags</td>
<td>Fragment Offset</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Source address IP</td>
<td>Header checksum</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP: Source Port</td>
<td>TCP: Destination Port</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP: Sequence Number</td>
<td>TCP: Acknowledgment Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Function of the bits in an IP datagram packet, using TCP.

- The transport layer is implemented by either TCP or UDP.
  - The Transmission Control Protocol make sure that all packets are received, and in order. It uses numbering and re-sending. [RFC 761]
  - User datagrams allow lost and unordered packets. It is thus faster than TCP. [RFC 768]

**Note:**
Of course, you do not need to memorize the particular bit positions. However, it is important to know what information is stored in a packet since this tells you what can and cannot be done at the network layer. For example, at the network layer one cannot filter on content (data) without also knowing how the content is represented.

5 Firewalls and Proxies

- A router sits between two networks and routes packets between them by looking at the destination IP address. It knows which numbers are "left" and which are "right".
- A firewall is like a router but it can decide not to route certain packets. It often looks at the port when making a decision (e.g., block all telnet connections).
  - Most companies place a firewall between them and the Internet.
- A proxy is a go-between. Outside machines don’t see the IP of machines inside. It works at the application level, e.g., Internet Object cache.
- In a client-server architecture the server provides services which the client consumes (e.g., browser-webserver).
• An **n-tier** architecture expands this concept to multiple levels (e.g., browser-webserver-database).

**Note:**
We care about firewalls mostly when they prevent us from deploying a distributed program. If you are building a large distributed system you will need to coordinate with the network administrators to make sure that all the packets can get thru at all times.

## 6 DNS

• The **Domain Name System** (DNS) resolves host names (e.g., jmvidal.cse.sc.edu) into their corresponding IP numbers (129.252.11.88). It is described in a series of RFCs.

• IP numbers relate to the physical layout of the network. They are used for routing.

• It is hierarchical. Each domain (sc.edu) has its own authority. If it does not know the mapping it will ask a higher domain (.edu) ending in one of the eight root DNS servers. For example:

  – When your machine hadar.cse.sc.edu needs to know the IP for www.wastingtime.com it will ask your local DNS server (cse). If it does not have it cached, it will ask the next one higher up (sc).

• Domain names relate to ownership and responsibility. They are meant to be used by humans.

• In Unix the **whois** command can be used to get ownership information:

```
whois sc.edu
```

Registrait:
University of South Carolina (SC6-DOM)
1244 Blossom Street
Columbia, SC 29208
US

Domain Name: SC.EDU

Administrative Contact:
Yarbrough, Stan (SY1779) syarb@GWM.SC.EDU
University of South Carolina
1244 Blossom Street
Columbia, SC 29208
803-777-7474 (FAX) 803-777-1900

Technical Contact:
Mowery, Bill (BM13149) bill.mowery@SC.EDU
University of South Carolina
514 Main Street
Columbia, SC 29208
803-777-4636 (FAX) 803-777-8644

Billing Contact:
Wider, Ella (EW1183) wider-ella@SC.EDU
USC DLIS
Thomas Cooper Library
Columbia, SC 29208
(803) 777-2498

Record last updated on 14-Dec-2001.
7 Internet Services

- Each host can establish a different connection on one of its ports. There are 65536 ports. Some are reserved:
  - telnet: 23
  - http: 80
  - ftp: 21
  - smtp: 25
- A class C address reserves the first 3 bytes (e.g., 199.1.2.*), while a class B reserves the first two (e.g., 127.1.*.*).
- 10.*.*.*, 172.1.*.*, 172.31.*.*, 192.168.*.* are nonroutable.
- 127.*.*.* is the loopback number.
- We are running out of IP numbers. IPv6 will add more bytes.

8 Example

1. Blossom wants to send a packet to buttercup at port 80.
   (a) Blossom queries the local DNS server for buttercup’s IP address.
   (b) Blossom broadcasts an IP datagram with destination address of 137.22.11.1 on the local Ethernet.
   (c) No other computer on the local Ethernet reads the packet because it is not addressed to them (unless they are running a sniffer).
   (d) The router recognizes that this IP will not find its destination of that subnet. It re-broadcasts the packet on the appropriate sub-net. The source and destination fields are kept the same.
(e) Buttercup sees the packet addressed to it and reads it.

2. Blossom wants to send a packet to bubbles on port 25.

(a) Bubbles does not have a real IP number, so it does not exist on the Internet.

(b) Blossom somehow (offline) determines that it should instead send a packet to 62.11.12.3. on port 25, hoping that the NAT will do the right thing.

(c) Repeat the first four steps of the previous scenario, except that this time the other router picks it up and forwards it to the Internet.

(d) The packet goes thru any number of routers on the Internet until it is seen by 62.11.12.3 who then changes the destination IP of the packet to 198.169.0.1 and places it on the local subnet.

(e) Bubbles sees the modified packet and reads it.

Note:
The computers "outside" an NAT see it as just one machine. The computers inside the NAT think that they are on the open Internet. The NAT has the job of remembering who on the inside is talking to who on the outside and re-write the Destination-IP fields of all the packets.

One drawback of using a NAT is that a computer on the outside cannot make first contact with a computer on the inside. Computers on the outside can only reply to messages sent from the inside. That is unless the NAT is set up to specifically forward new packets to some machine inside. For example, it could be set up to forward all new packets to port 80 to a particular machine which servers as the company’s web server.

9 Internet Standards

<table>
<thead>
<tr>
<th>Internet Engineering Task Force (IETF)</th>
<th>World Wide Web Consortium (W3C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Democratic. Open to anyone.</td>
<td>Vendor organization led by dues-paying corporations.</td>
</tr>
<tr>
<td>TCP-IP, MIME, SMTP</td>
<td>HTTP, HTML, XML, CSS</td>
</tr>
</tbody>
</table>

| 1. Experimental                       | 1. Note                        |
| 2. Proposed standard                 | 2. Working draft               |
| 3. Draft standard                    | 3. Candidate recommendation    |
| 5. Informational- not required       | 5. Recommendation              |
| 6. Historic- obsolete                |                                 |

10 Uniform Resource Identifier

- The URI is defined in RFC 2396[^10].
- It looks like scheme:scheme-specific part. For example, the scheme part can be: data, file, ftp, httpd, gopher, mailto, news, telnet, wais, etc. The scheme-specific part usually looks like //authority/path?query
- It can be further classified as either a location, a name, or both.
• A Universal Resource Name (URN) is required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable. \[\text{RFC 2141}\]^{11}.

  - \texttt{urn:namespace:resource-name}
  - For example: \texttt{urn:isbn:1234-98754-09}
  - Not widely used, yet.

• A Universal Resource Locator (URL) identifies resources via a representation of their primary access mechanism. \[\text{RFC 1738}\]^{12}.

  - protocol://username@host:port/path/file#fragment?query

11 HTML, SGML, and XML

• The Standard Generalized Markup Language (SGML) was established in 1986 as an ISO standard for describing a text’s semantics, rather than its appearance.

  - For example, saying that "The title of this document is RFC 123", rather than "Center RFC 123 on this page with an 18pt bold font".
  - SGML was championed by librarians and the publishing industry.
  - The idea was to have one source document which could then be automatically transformed into many formats.
  - SGML is a meta language. Over time it became very hard to write a parser for it.

• The Hyper Text Markup Language (HTML) is an instance of SGML.

  - It looks like:

    \[
    \begin{aligned}
    \text{<ul>}
    \text{<li>The <b>URI</b></li> is defined in <a href="http://www.ietf.org/rfc/rfc2396.txt">RFC 2396</a>.</li> \\
    \text{</ul>}
    \end{aligned}
    \]

  - When Tim Berners-Lee was looking for a way to markup the text for his hypertext application he decided to base it on the well-known SGML.
  - The original intention was to make all tags describe semantics (e.g., H1, STRONG, EM) but over time tags started describing display attributes (e.g., B, I, BLINK, HR).
  - HTML is now at version 4.0. Many of the tags have been added by vendors. The syntax has become corrupted.

• The eXtensible Markup Language (XML) is the W3C’s effort to create a new meta language that is simpler than SGML.

  - XML is very easy to parse. There are many parsers available for every programming language.
  - XML does not define any tags. It is up to the user to define the tags that he will use. For example:

    \[
    \begin{aligned}
    \text{<question>}
    \text{<text>Who created the first Web client and server?</text>}
    \text{<choice answer="1">Tim Berners-Lee</choice>}
    \text{<choice>Marc Andreessen</choice>}
    \text{<choice>Bill Gates</choice>}
    \end{aligned}
    \]
The set of tags allowed by a particular XML document are defined in its **DTD** (Document Type Definition) or **schema**.

XML is very popular.

### 12 MIME

- The Multipurpose Internet Mail Extensions (MIME) standard is defined in **RFC 2045**. 
- It was originally developed as a way to add (binary) attachments to what was at the time only plain-text email.
- It has become a widely used technique for describing a file's content.
- It supports almost a hundred predefined types of content.
- Content types have a type, and a subtype. The type gives a general idea of what the data represents (e.g., a picture, a video, a movie), which the subtype specifies the specific encoding (e.g., gif, jpeg, mpeg). Some examples are:
  - text/html
  - text/plain
  - image/gif
  - image/jpeg
- MIME also allows the user to define non-standard types, by prefixing them with an x-
  - application/x-framemaker
  - audio/x-wav

### 13 Hyper Text Transfer Protocol

- HTTP was originally defined in **RFC 1945**. HTTP 1.1 is in **RFC 2616**.
- HTTP is a stateless protocol for fetching MIME-encoded information from another machine in the Internet.
- The steps are:
  1. Make connection.
  2. Make request:
     ```
     GET /index.html HTTP/1.0
     Accept: text/html, text/plain
     User-Agent: Mozilla/1.0
     \r\n     \r\n     ```
     A POST request can include data before the
     ```
     \r\n     ```
  3. Get a response:
HTTP/1.1 200 OK //response code
Date: Fri, 28 Dec 2001 14:59:11 GMT
Server: Apache/1.3.20 (Unix) PHP/4.0.6 mod_ssl/2.8.4 OpenSSL/0.9.6b
Last-Modified: Fri, 28 Dec 2001 14:55:00 GMT
ETag: "a5a99-17f5-3c2c87c4" //entity tag
Accept-Ranges: bytes
Content-Length: 6133
Connection: close //one-shot
Content-Type: text/html

\r\n\r
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
<html LANG="en"><head>
<TITLE>University of South Carolina Home Page</TITLE></head>
......


• Closing the connection every time is costly. HTTP 1.1 does not close the connection.

• Cookies are opaque strings that are sent by the browser to the server based on its IP number.

• Under the Common Gateway Interface (CGI) the server rather than returning the named file will run it as a program, passing it the arguments and values in the URL path.

Note:
The fact that HTTP is stateless was both one of the reasons for its widespread early adoption as well as one of the biggest headaches when using it for complex applications. Because it is stateless it is very easy to implement, so basic web servers could be written in a page of code. However, it also means that if the user is involved in an interaction that requires more than one step then some sort of cheat must be used. The first attempts extended the URL with state information. This created some large ugly URLs and gave rise to problems if the user decided to bookmark that URL. The next attempt was the standardization by Netscape of Cookies.

13.1 HTTP Request Methods

• The line GET /index.html HTTP1.0 is a request, where GET is the method

• Some of the most common methods are:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET URI</td>
<td>Retrieve information pointed to by URI.</td>
</tr>
<tr>
<td>HEAD URI</td>
<td>Identical to GET but server must not return a message body.</td>
</tr>
<tr>
<td>POST URI data</td>
<td>Request that the server accept the following data as a subordinate of the resource pointed to by URI.</td>
</tr>
<tr>
<td>PUT URI data</td>
<td>Replace the data pointed to by URI with the following data</td>
</tr>
<tr>
<td>DELETE URI</td>
<td>Delete the data pointed to by URI.</td>
</tr>
</tbody>
</table>

13.2 HTTP Request Headers

• After the request line one can add any number of request headers.

• Some of the most common headers are:
| **Accept:** | Specify media types which are acceptable for response (e.g. pdf, gif, png, flash, etc.) Can also give ordered preferences using the q quality value: 1 is most preferred, 0 is least. | Accept: audio/*; q=0.2, audio/basic |
| **Accept-Charset:** | Specify what charsets are acceptable for response. | Accept-Charset: iso-8859-5, unicode-1-1; q=0.8 |
| **Accept-Encoding:** | Restricts the content encodings that are acceptable. | Accept-Encoding: compress, gzip |
| **Accept-Language:** | Restrict the language. | Accept-Language: da, en-gb; q=0.8, en; q=0.7 |
| **Authorization:** | Include the credentials (username and password) needed to authenticate with the server. | |
| **From:** | Contains the email address of the human issuing the request. | From: president@whitehouse.gov |
| **Host:** | Specifies the Internet host and port number of the resource being requested. | GET /pub/WWW/ HTTP/1.1 Host: www.w3.org |
| **If-Modified-Since:** | Only return the document if it has been modified after the given date. | If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT |
| **User-Agent:** | Specifies the user agent (web browser) being used. | Mozilla/4.0 (compatible; MSIE 5.5; Windows 98) |

### 13.3 HTTP Response

- The server returns a **response** which begins with a status code number, such as **404: Page Not Found**, and **200: OK**.

- There are 40 defined status codes.

- The response can also include any number of response headers
### 14 CGI

- The [Common Gateway Interface](#) is a method for generating web pages dynamically.
- Instead of interpreting the argument of GET (or POST) as a filename, its interpreted as a program name with arguments.
- The URL for a GET is interpreted as `programname?parameter1=value1&parameter2=value2&...`. The values are optional. The parameters and values are passed as command-line arguments to the program.
- For a POST, all the parameters and values are passed to the program in the standard input. That is, as though a person had first run the program and then typed the information on the keyboard.
- The program is run and whatever it prints out is sent back to the client.
- CGI programs run as independent processes and so have many drawbacks:
  - They consume a lot of system resources.
  - They suffer from startup delays.
  - It is hard to maintain state between successive calls (must write to file).
- It is because of these reasons that server-side systems were introduced. These include Java servlets, PHP, Zope, IBM’s Webshpere, etc.

### 15 Applets

- Applets are programs that run on the client.
- They are included by a special HTML tag:

  ```html
  <applet codebase="http://jmvidal.cse.sc.edu/applets/Circle"
    code="Circle.class" width="200" height="100" />
  ```

- Upon seeing this tag the browser opens a connection to the codebase and downloads the classfile.
- The client then runs this program.
• Of course, since the program is running on the client, we must be careful about what actions it can take. Java applets run on a virtual machine which limits what they can do.

• Applets continue to be used in situations where a small program is the best way to communicate with the user (e.g., mortgage payment calculators).

Notes


This talk is available at http://jmvidal.cse.sc.edu/talks/internet

Copyright © 2004 Jose M Vidal. All rights reserved.