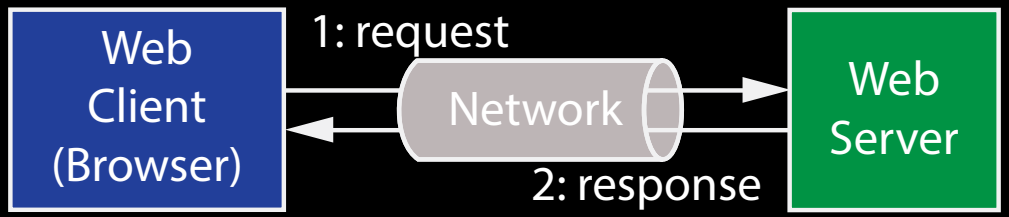
Web Application Architectures

Module 5: Middleware

Lecture 2: The Hypertext Transfer Protocol (HTTP) – Introduction

HTTP Protcol

- We've already seen that the HyperText Transfer Protocol (HTTP) is the foundation for data communication on the Web, and that it involves request/response interactions:



- HTTP is an application layer protocol used to deliver resources in distributed hypermedia information systems. In a Web application, the request initiates activities that are implemented over the middleware, and the response typically involves returning resources to the browser.

- In order to build and debug web applications, it's vital to have a good understanding of how HTTP works.

HTTP – Resources

The resources delivered as part of this protocol typically include hypertext, marked up using the HyperText Markup Language (HTML), cascading style sheets (CSS), hypermedia and scripts —

- Hypertext – text that can be displayed on a computer, or other display device, possibly styled with CSS, and containing references (i.e., hyperlinks) to other hypertext that the reader is able to immediately access, usually via a mouse click.
- Hypermedia – the logical extension of hypertext to graphics, audio and video.
- Hyperlinks – define a structure over the Web. Indeed, this is the structure that Google uses to determine the relevance of hyperlinks that are returned to you by a search.
- Scripts – code that can be executed on the client side.

HTTP – Background

- The HTTP protocol is extremely lightweight and simple – indeed, that's one of the main reasons for its success.

- Initially, with HTTP/0.9 (the first documented HTTP protocol), a client could only issue GET requests, asking a server for a resource.

  Ex. `GET /welcome.html`

  will cause the server to return the contents of the requested file (the response was required to be HTML).

- The HTTP/1.0 protocol, introduced in 1996, extended HTTP/0.9 to include request headers along with additional request methods.

HTTP – Background

The HTTP/1.1 extension followed soon thereafter, and included the following improvements:

- Faster response, by allowing multiple transactions to take place over a single persistent connection.
- Faster response and bandwidth savings, by adding cache support.
- Faster response for dynamically-generated content, by supporting chunked encoding, which allows a response to be sent before its total length is known.
- Efficient use of IP addresses, multiple domains can be served from a single IP address.
- Support for proxies.
- Support for content negotiations.

HTTP – Basics

- HTTP has always been a stateless protocol.
- This refers to the fact that the protocol does not require the server to retain information related to previous client requests.
- Thus, each client request is executed independently, without any knowledge of the client requests that preceded it.
- This made if very difficult for web applications to respond intelligently to user input, i.e., to create the interactivity that users expect when they use computer applications.
- Cookies, sessions, URL encoded parameters and a few other technologies have been introduced to address this issue, thereby allowing for the emergence of Web 2.0 and 3.0 applications.

HTTP Sessions

An HTTP session proceeds as follows:

1. An HTTP client (e.g., a browser) establishes a TCP connection to a particular port on a host server (typically this is port 80), and initiates a request. Establishing the TCP connection may first involve using an DNS server in order to obtain an IP Address.
2. An HTTP server listening on that port waits for a client's request message.
3. Upon receiving the request, the server processes it and sends back a status line, such as "HTTP/1.1 200 OK", along with a message of its own (i.e., a response), the body of which might be a requested resource, an error message, or some other information.

HTTP Sessions

In addition to using browser developer tools, you can also directly explore how a web server responds to client requests using telnet:

```
[~]$ telnet www.unm.edu 80
Trying 129.24.168.10...
Connected to wwwvs.unm.edu.
Escape character is '^]'.
GET /index.html
```

- This tells the server that you (the client) are making an HTTP GET request, asking for the file `index.html`.
- Hit <return>, and the HTML associated with the resource will be provided.