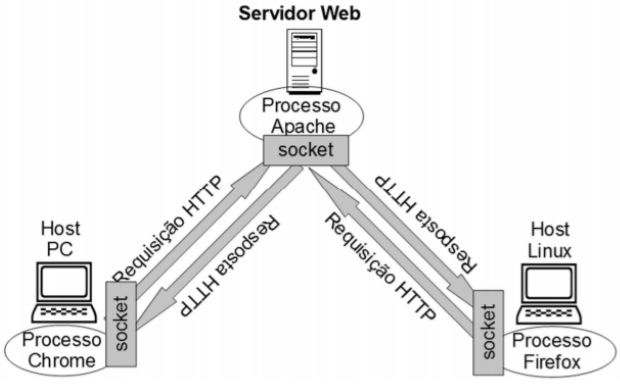
Network - Web Service

Web - HTTP (Hypertext Transfer Protocol)

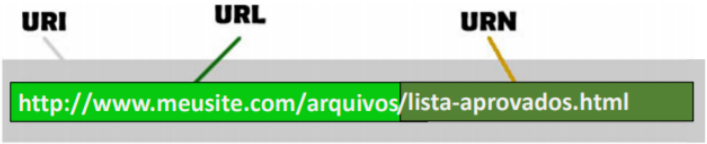
- Application Layer Protocol;
- Runs over TCP;
- Implemented in the form of a client program and a server program;
    - **Customers:** request an object;
    - **Server**: transfer the object to clients;
- Does not save state between different connections, that is, each connection is new to the server;
    - Stateless x statefull.

➢ **Web** - HTTP (*Hypertext Transfer Protocol*)



URIs and URLs

- URI (Uniform Resource Identifier);
- URN (Uniform Resource Name);
- URL (Uniform Resource Locator).



General syntax of a URL:

- <protocol>://<server>:<port>/<path>/<resource>
- The port is optional for services on default ports;
    - Default ports: 80 (HTTP), 443 (HTTPS), 21 (SSH), 23 (Telnet);
- Path and resource can be omitted (partial URLs);
- URLs can contain data after the resource name.

x-www-form-urlencoded format

- Format for data transfer over HTTP;
- sed to encode the data produced by the customer;

Syntax

- There can be no white space in the data;
- Data is grouped into name=value pairs;
- Pairs are separated by &;
- White spaces are encoded with +;

HTTP Header - Common Data

- **User-Agent** - contains a string that identifies the browser making the request;
- **Host** - server address;
- **Date** - date of request;
- **Accept** - there can be multiple accept fields. Each contains a MIME type (type/subtype) that is accepted by the requesting browser;
    - Ex.: Accept: text/plain, text/html;
- **Accept-Encoding** - types of encoding accepted by the browser;
    - E.g.: UTF-16, ISO-8859-1;
- **X-Powered-By** - Specifies the technology used by the server (e.g., PHP, ASP).

HTTP request message - GET

- GET request method is used to request a resource from the server;
- Request data is visible in the URL;
- Eg: return codes: 200 (OK) or 404 (Object not found);

HTTP request message - POST

- POST request method is used to send data to server;
- sually used to upload file and HTML form data;
- Request data is not visible in URL like GET;
- However, just like GET does not protect data!
- E.g.: return codes: 200 (OK)

HTTP request message - PUT

- PUT request method is used for the user to create an object in a web server directory;
- If the objective does not exist, it is created or changed when it already exists;
- E.g.: return code: 201 (Created);

HTTP request message - DELETE

- DELETE request method is used for the user (or application) to delete an object in a Web server directory;
- Eg: Return code: 202 (Accepted).

HTTP Request Message - HEAD

- The HEAD method is similar to the GET method;
- The server that receives HEAD, returns the request header, but does not return the data;
- Typically used by HTTP server developers for debugging purposes:
- Check the file's last modified date, file size,...
- Eg: Return code: 304 (Not Modified).

HTTP response message

- 200 OK: successful request and delivered in response;
- 304 Not Modified: Object not modified (useful for Web proxy);
- 400 Bad Request: Request not understood by the server;
- 403 Forbidden: Access to the requested resource is prohibited;
- 404 Not Found: Requested object does not exist on the server;
- 505 HTTP Version Not Supported: HTTP version of the request is not supported by the server.

Cookies/Sessions

- Remember, HTTP is a stateless protocol;
    - Stateless;
- Many application actions need to maintain state when navigating between pages, how to do it?
    - Eg shopping cart on a website.

Applications that need to maintain state

- Status information can be saved in the CLIENT's browser (using cookies);
    - Users can disable cookies in browser;
- You can save status information on the Web SERVER (using sessions).

- Kind of variables that are saved in the browser;
- Cookies are sent by the web server through the HTTP protocol;
- Cookies are sent on future browser requests to the web server

Cookies - Working example

1) John always accesses the web from his PC through Chrome;
2) João accesses the website www.sitey.com for the first time;
3) When the request arrives at the sitey server, it creates a unique identification number for John;
4) In the header of the HTTP response from sitey's web server to João's browser, there is Set-cookie: 1234 (the identification given by the server to João).
5) John's browser sees the Set-cookie header and attaches a line to the cookie file he manages;
6) Server host name and its identification number;
7) Every time João navigates through sitey, his browser consults his cookie file, extracts his identification number for this site and inserts it in the HTTP request (Cookie: 1234) so that sitey knows that it is João who is accessing it ;
8) The sitey thus knows which pages John visited recommending free movie sites

Sessions

- Session variables have to be stored on the web server (e.g., database);
- Sessions need a “timeout”;
- Otherwise, the web server will not know if a session is active;
- Security reasons too.