Web Browser Concept Map

The System that I chose for this Concept Map is a web browser. There are a few reasons that I decided to choose this topic for the map. First, it is a tool that I (and most people in the 21st century) use on a daily if not hourly basis. A web browser is a complicated yet simple piece of software that allows a user to make complex calls to any server hosting itself on the internet, all within a self-contained interface that makes it easy to understand what is happening without really knowing what is going on under the hood. A user can keep track of many tabs' worth of information at a time, siphoning streams of data from multiple locations at once. At the moment, I am developing a web application for Firefox, my browser of choice, which also played into my selection of this topic.

Algorithms are a sequence of instructions, often computer readable, that make a complex problem more understandable by using smaller subroutines to define it. **Decomposition** is the process of breaking a complex problem into a more manageable and easier to understand system of parts. **Abstraction** refers to the removing of information and details that are otherwise nonessential to the user that is experiencing and handling the given system. Finally, **Communication** is a large idea that encompasses how we share messages and data with each other, as well as how other systems complete the same process; for instance, how computers communicate with each other across the web.

HTTPS -> Algorithms: The HTTP protocol must ensure that it is able to correctly establish and maintain a connection to a server [1], and on top of that it might need to guarantee that it is secure as well. The need to break this routine into smaller steps according to the type of connection being established, the types of devices making the connection, and a variety of other factors is certainly a given.

Algorithms-> Decomposition: Nobody should deny these two go hand in hand. Oftentimes a complex problem must be decomposed into more manageable and understandable problems for a developer to comprehend, and thus the algorithmic approach is well suited to the task.

HTTPS -> Abstraction: There are a plethora of reasons why a connection to a server could fail, and for this reason one of the most well-known abstractions on the web has been created. Clients have been getting 404s for decades, but a suite of other error messages help abstract away the complex details of connection failure into more manageable subgroups of errors [2].

Data Presentation -> Abstraction: Resources must be retrieved from more than a single server for many webpages on the internet now, and the task of combining them all and making them look pretty is a process the web browser does completely behind the scenes to make the UX as clean as possible.

Tabs -> Chrome: The Chrome of a browser is the many visual design elements that surround the user's data and provide tools to the user, given to them by the browser itself [3]. Without these tools we could hardly navigate even simple data, and thus we understand that the software provides us powerful tools to manage the data we receive, most notably in the form of tabs, the most interacted with chrome function.

Works Cited

1. "How Does HTTPS Actually Work?" *Robert Heaton*, https://robertheaton.com/2014/03/27/how-does-https-actually-work/.
2. "Status Code Definitions." *HTTP/1.1: Status Code Definitions*, https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html.
3. World Leaders in Research-Based User Experience. "Browser and Gui Chrome." *Nielsen Norman Group*, https://www.nngroup.com/articles/browser-and-gui-chrome/.