Steven Myers ([stelmyer@indiana.edu](mailto:stelmyer@indiana.edu))

A290 Final Project Phase 0 & 1 – Project Proposal

Phase 0 Ideas:

-Create a simple IDE with simple syntax highlighting, ability to save work, and very minimal code snippet completion.

-Create a form entry application that inserts rows into a SQL database, and pulls records and displays them in an Excel-like format. Also, have some data-views like charts.

-Create a wire frame tool for web development that allows drag and drop components to quickly mock up a design for a web page. Features would include the ability to export it as html, CSS, and JavaScript and add logic workflows behind each graphical element, much like Visual Studio.

**-\*Create an application that scrapes web pages for links and builds a visual graph of interconnected websites. Add functionality that would allow the user to specify recursion and visitation depth, so that a target website’s links get visited and added to the graph of connected websites. Perhaps add functionality for page rank displays and such as well.**

Phase 1 (chosen idea):

I chose the last, bolded idea to create an application that scrapes the web for links. My original inspiration was to visualize the connections (like a roadmap) of websites linking to and from each other. This would be a fun project to learn a little bit about graphing algorithms, as well as some minimal work in parsing the html. I have already found a decent C# library called HTMLAgilityPack (located here: <https://htmlagilitypack.codeplex.com/>) that I can use to parse certain html elements out using XPath expressions (yuck!). It works quite well and I am already able to scrape the links from a target web page and I’ve even made a simple form to allow a user to search for a webpage, then fill in a textbox of the all the attribute values of all the anchor tags (<a href=”THIS STUFF”).

So, the finished product might be a windows form application with a simple search textbox for a target HTML page, submit button, and a combo box to specify recursion depth. On submit, the application goes out and grabs all the links from the given webpage, and for each of those links, it recurs and perform the same operation several times specified by the user in the combo box for the recursion depth. The links will be stored in a data structure for graphs, likely an *Adjacency List*, of nodes (target websites), and their links to other sides (edges, child nodes). This will be a directed graph data structure, since links have one direction (they can reference back to create a cycle, though).

Once I’ve built up our graph representation, I would like to visually display the information (if chosen by the user) as nodes and edges on a “canvas”. I am yet to do more research on the proper element that this would be displayed in – preferably something that the user would be able to interact with via the mouse and keyboard (e.g. they can drag around the map, zoom in, zoom out, click on nodes to get more information). This would be a useful application for visualizing the interconnectivity of a website. For example, if I create a website, it’s important that my users can access several pages at any given page so that navigation is easy. By running this application, I can see how well each page connects to all my other pages (its all-to-all connection). It’s nice for user experience designers so that they can reevaluate how statistically (and visually) easy it is to reach any other page given one page from a website. For external websites (“neighbor sites”), you could also see how many references back to your target website exist. In summary, the target audience would be web developers, graphic designers, and user experience designers.

As for the timeline, I’m confident I can get this done by the end of the semester because I know enough about graph data structures to come up with something efficient. I already have a working application that can scrape the anchors from a webpage and get the links. I need some ways to validate user input to prevent html parsing errors or invalid URI’s. The tough part of the project will be the actual visualization, but I am sure I will be able to find the proper medium in C# to pull it off. From the simple examples using the *Graphics* and *Pen* in class, you could already draw simple nodes and edges and just distance them by some amount.