Accessible Interactive Campus Mapping System For Syracuse University/SUNY-ESF

Aaron C. Knight

Accessible Interactive Campus Mapping System For Syracuse University/SUNY-ESF

Aaron C. Knight

Candidate for B.S. Degree
in History and Computer Science with Honors

May 2007

APPROVED

Thesis Project Advisor: ______________________________
Ernest Sibert

Honors Reader: ______________________________
Steve Simon

Honors Director: ______________________________
Samuel Gorovitz

Date: ______________________________

Abstract

This written component of my Capstone Thesis in Computer Science covers the conceptualization and realization, and the process in between those stages, of my project, the Syracuse University Accessible Interactive Campus Map system. The system in its present state is launched at http://map.syr.edu.

This project creates a usable and useful map and directions system for the Syracuse University and SUNY-ESF communities, which have long suffered without any properly oriented or fully functional map, much less an online one of any use.

What this project also does is turn the Syracuse University Accessibility Map, produced by the Office of Disability Services in 2004, into an updated electronic resource which the university as a whole can use. Bringing in directory information as well, this project provides a comprehensive directory, map and directions system which will serve the university community for many years to come.

Table of Contents

Acknowledgements

I'd like to acknowledge the efforts of this project's original team members: George Monioudis, Josh Killian and David Kazlman as well as its later addition, Yu-Pin Hsaio. I would also like to acknowledge the contributions, support, and refinement given this project by Dr. Ernest Sibert, Steve Simon and the Office of Disability Services, Joseph Stoll and the Syracuse University Cartographic Labs, the Office of Orientation and Transitions Services, Larry Quaglia and the Technical Services Group for the Division of Student Affairs, and the Office of Design and Construction.

Capstone Thesis

The Syracuse University campus has never reliably maintained a comprehensive and accessible map of the campus for students, parents, and campus visitors. Published maps, including those posted in the last few years on the campus grounds, are beset with inaccuracies, omissions and other problems.

In 2004, as the Office of Orientation and Transitions Services's web designer, I tried to fix some of the campus map issues, in particular the lack of any usable online map besides poorly-oriented print maps. This new system will replace it and all other online campus maps, with far superior capabilities and better base maps than those I had available in 2004.

In 2005, Syracuse University posted the maps mentioned earlier on the campus grounds, attempting to provide navigation tools for campus visitors. While these maps were a major improvement (they had no permanently posted maps previously), they were far from ideal. Those maps remain posted to this day, neither oriented with North at the top, as traditionally is the case, nor oriented in the direction one faces while looking at the map. Instead, depending on where the map is, one could easily be fooled into believing that the Carrier Dome is on the northwest, northeast, or southeast corner of the campus – while it is in fact located in the southwest corner of main campus. South campus doesn't even get the benefit of posted visitor maps; the sole map on South Campus (which is much more confusing to locate buildings from) is posted in the Goldstein Student Center.

With the lack of accessible mapping systems on this campus in mind, and the lack of any really useable online systems in particular, this project was born. The project was suggested to Dr. Ernest Sibert, professor for CIS 453-454, Software Specification and Implementation, by Steve Simon, Director of the Office of Disability Services. The Office of Disability Services had in 2004 created a printed Access Map and Guide of the campus, covering accessibility initiatives in place and information about such features throughout the campus.

George Monioudis, David Kazlman, Josh Killian and I took on the project. Our goal was to formally specify this project and then implement it during the following semester. The overall project would be an interactive campus mapping system which provided directions, maps, and campus directory information in one convenient online system. The system would also be designed to people first; specifically considering persons who are blind or users of wheelchairs.

Stage One: CIS 453, Software Specification

The first stage of the project was formally specifying what it would do. In the Computer Science industry, this is achieved via a Software Requirements Specification (SRS). SRSes define the concept and goals, anticipate problems, and define budgets – and does all of this in a method designed for non-technical audiences (managers, CEOs perhaps).

Working with our project's client, the Office of Disability Services, the project was fully specified. Modeled after MapQuest™ and other mainstream directions and mapping sites, it would provide written directions to and from any point on Syracuse University's campuses, North, South and Downtown. Searching options would be provided, allowing the directions it generated to accommodate various disabilities. Designing with web standards in mind, the system would also accommodate any screen reader software necessary for users to use the system.

There were two disabilities that we focused on. The first was mobility (providing directions that eliminated paths with steep grades or with stairs). This wasn't foreseen as a major problem; the fail-safe measure we created was that it would resort to busing options that were accessible to fill in any gaps left by Syracuse's hilly campus.

Some accommodations, however, seemed conceptually simple but certainly were not easy to implement. One particular example of this is the other disability we intended to accommodate: vision impairments. The intention was to make the system usable by both the fully blind and those with more minor vision impairments. It is with the disability of blindness that we would encounter the most challenging issue: how could we give directions spanning unnamed and unmarked paths without visual cues?

Working with a Michael Dillon, an Orientation and Mobility Specialist with the Commission for the Blind and Visually Handicapped (who the Office of Disability Services typically contracted with for orienting blind students to the campus), we determined our goal of assisting individuals with blindness was going to be much more limited in scope.

At his direction, we declared specific paths which could be considered usable for blind users, eliminating possibly confusing paths and those which would be hazardous (such as sidewalks crossing high-traffic driveways or that passed building loading zones). Designing for disabled users did place constraints on the design, culminating in what follows, the final design constraint statement from the SRS[1]:

> Common web technologies that are cross-platform are essential to the success of this project. The project's written outputs, such as directions, must be readable by screen-reader software such as JAWS, the common standard at Syracuse University.

Once the project was formally specified, we could begin work on implementing our solution. The specification was finished in

---

[1] Cited: "General Constraints," section 2.6. Full SRS is included as Appendix B.

December 2005. Work would commence on the project during the following semester.

Stage Two: CIS 454, Software Implementation

Upon returning from the January semester break, we took the framework from the previous map (using its original databases as a starting point for our own) and created the mapping base system. Intending to create a "useful and easy-to-use interface, that looks consistent across multiple browsers and platforms,"[2] I tried a few different ways to represent a useful portion of the map, with zoom and view panning controls, before reaching a final design which we demonstrated to the class in February 2006. This design would later be adapted in response to feedback received.

As the only team member well-versed in web design and specifically in PHP (the web scripting language we were programming the system in), my task was to make the entirety of the system work. George Monioudis worked with the professor to create a direction search algorithm that would perform the actual search. Josh Killion and Dave Kazlman were tasked with inputting the thousands of data points necessary for the system to function.

Our team quickly encountered problems. One of the first came in the graphics engine – the part of the system which generated the

---

[2] SRS, Section 2.5 (User Objectives). See Appendix B.

maps on-the-fly. To make the maps crop dynamically, so we didn't have to create a cropped map file for every single building and location on the campus, we intended on using the GD library[3] with PHP, the web language defined for the project. Our project was hosted by the Division of Student Affairs, and we needed this library installed on their web server. This process, due to problems on the Division's Technical Services Group's end, took almost a month just to get working so we could test the system.

The next problem we encountered was with the map files themselves. Late in the specification stage, we received (through the efforts of the Office of Disability Services) the original map files for the campus. I found errors in their maps, so several exchanges were made with the Cartographic Labs to arrive at a map we could use. Further, the amount of construction underway on campus lately has caused these maps to be revised every few months since we received them.

Once those issues were worked out, the basic map interface was created. A few features were quickly implemented, so that with the exception of the directions component, the system was functional. Data entry began at this point as we finalized the database model.

Here, again, we ran into problems. The sheer amount of data entry required would cause perpetual issues for the next year. Because

---

[3] Full documentation on the GD library is available on the web at http://www.libgd.org/.

the task is mind-numbingly boring, it was often put off. By the end of the semester, we had a full data set for buildings and offices (for the directory component of the project) but few properly-entered map data points for the directions system.

The directions system itself was fraught with challenges. By early April, several months after the directions algorithm was to be completed for implementation, the code (written in C++) was finally received. It proved nearly impossible to implement in PHP, however, due to differences in the object model and the differences between C++ (a desktop programming language) and PHP (which truly is a scripting language).

We spent eight months fighting with George's algorithm before finally changing to an entirely different algorithm. George's algorithm was a variant on Dijkstra's Algorithm[4], a common path-search algorithm, using a form of heuristics to guide the search. Dijkstra's Algorithm performs a search through all available vertices (the map itself would be represented by a set of vertices and a set of edges connecting those vertices – the two forming a directed graph of the entire campus) and finds all paths from a given point A to a given point B.

---

[4] The clearest pseudocode algorithm I've been able to find is located on Wikipedia at http://en.wikipedia.org/wiki/Dijkstra_algorithm#Pseudocode.

We later switched to the A* (A-star) Algorithm, a variant on Dijkstra's Algorithm which uses heuristics to guide it to the best single path between two points rather than all such paths, then returns it. This algorithm's implementation will be detailed later.

By the close of the spring semester, we had implemented the map system without the directions search system. After this point, our team efforts would dwindle. The course had ended, and the remaining work would be completed in George's and my own free time. This later became solely my own free time.

MayFest 2006: Unveiled

The project was formally unveiled to the Syracuse University community during the campus MayFest (in April 2006). We presented our project throughout the day, fielding questions, taking suggestions, and talking about the future of the project.

At that time, we projected that the map would be completed and launched publicly in late May. We reasoned that we had a whole month to get the remaining work done (make the directions system work and finish working bugs out, among other things). We also believed that we could quickly resolve the data entry problem, just a few long nights and get it all finished.

As it would turn out, we had the right month but the wrong year. The project encountered many more delays, and wouldn't be launched (even without a complete directions system) until the following year.

Stage Three: Present

After the Software Implementation course ended, development progressed much more slowly. During the summer months, some time was devoted to it while I worked at Orientation and Transitions Services – I changed the design for better usability and completed the rest of the design. The site design was rearranged in the current form as depicted in Appendix A, to make the design more welcoming and clear. During this time, some work was devoted to troubleshooting the directions algorithm.

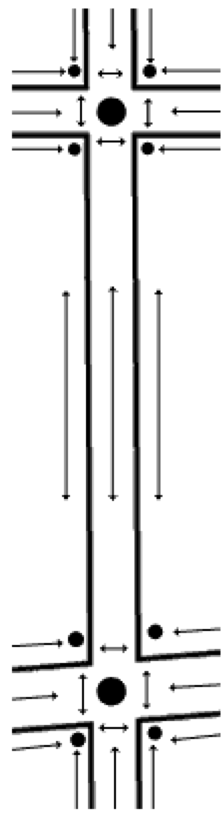
During the fall semester, weekly meetings were held between Dr. Ernest Sibert, George and myself to expedite the directions engine's completion. After months of collective efforts failing to make progress, we finally abandoned George's C++ algorithm in late December 2006.

During that same semester, work was done to make a desktop client version in another class. Yu-Pin Hsaio, a member of our class's desktop client development team, suggested and coded in the Java language an implementation of the A* (A-star) algorithm. This implementation was tested and working by the end of December, and I decided that we could solve our directions system woes by implementing this version in PHP. Working with Yu-Pin, the conversion was implemented over the semester break and completed by the end of January 2007.

With a now-working directions engine, attention once again went back to the data-entry components. In order to thoroughly test the directions engine, a decently complete set of data was required. Unfortunately, very little of the data entry had been completed. At the end of the implementation class, both campuses' edges and vertices were supposed to have been completely entered – some several thousand data points of each. In reality, there was about a third of one campus (South Campus) entered and the whole of North Campus in vertices finished – but, in the case of North Campus, inaccurately. So there was no testable set of data.

By this point, the team consisted roughly of me, myself, and I. Due to my own scheduling constraints and the sheer amount of data involved, the data entered so far is approximately five hundred vertices for North Campus, with the edges connecting them as well. This roughly covers the whole of North Campus north of about East Adams Street – hardly a comprehensive coverage of the campus.

The process of entering this data deserves some explanation. For every vertex, there is at least one edge leaving it and one entering. Now, if you imagine a four-way street intersection – both streets crossing are two-way streets, four corners and all of that – here is how that would be entered, data-wise (see figure at right for more clarity): five vertices – one street (driving) intersection point and one for each corner – and eight pedestrian edges: eight crossing edges between the four pedestrian vertices.



There are 13 rows of data to be entered for each intersection. And that neglects the edges connecting that intersection to others – as in the figure above. Every one of those arrows is double-ended – as in two edges. Each black circle is a vertex. Each of those arrows coming in off the edge of the diagram is connecting to another intersection. And that's for a simple four-way intersection; some of the campus path intersections have five, six or even seven paths intersecting at once.

So suffice it to say that there is a lot more data to be entered for the project and none of it's easy. For every vertex, the x and y coordinates of the point are needed; for every edge, the start and end vertex entries. In order to create the directed graph representation with the edges, the vertices being connected have to already be entered. What eventually happened was that the map would be segmented out, so that testable data sets could be created without finishing entirely with the vertices.

Conclusions

The demonstration from MayFest 2006 will be repeated (with more of the project completed) and formally announced at the 2007 event. What will become of the project is still somewhat unclear; what is known at this point is the completion of data entry will fall under the jurisdiction of the Office of Disability Services.

Several meetings over the Spring 2007 semester yielded a willing coordinator and some final components originally designed but never implemented – a full content management system, for entering and modifying existing data is under design at the present time and will be finished in order for Disability Services to take over the project. Enough data will have been entered to run the system; the data entry will be finished by the office.

At the behest of the Office of Disability Services some modifications have been recently made which will allow the map to be fully launched in May 2007. After two years of work, the map will finally be launched publicly and Syracuse University will finally have a useful map system.

If anything has been learned from this project, it is that it will take no less than twice the initial time estimate to complete the project.

Given more free time to devote to the map project over the last year, I might have completed it in one year; it now stands at almost two. We originally estimated that we would be able to complete the full implementation in the Software Implementation semester.

Another major lesson learned from this project is the importance of leadership and persistence. From the outset, we knew the result we wanted[5]:

> The Syracuse University Accessible Map project is designed to provide an accessible and usable map and directions system that ties together graphical maps as well as written directions to assist all users, especially those with accessibility issues.

Two years later, those results have been achieved and Syracuse University will finally have a campus map which is accurate, useful, and properly oriented. We were ultimately successful only through coordinating the schedules of many busy people and offices; with the assistance of several offices and many people who otherwise would not have thought such a project possible given each office's scarce resources, we generated a large-scale project for the entire university community. The pooled effort and resources of many offices, and the support of the university as a whole in the effort, through my own and others' leadership, culminated in a major success.

[5] Quoted from the full Software Requirements Specification, Section 2.1; see Appendix B.

These two lessons culminate in overall the most important lesson of this project: anything that can go wrong will go wrong, and that problems have to be taken in stride. This project faced many setbacks – problems within the team, between offices, and overall problems we hadn't considered when designing the project originally. While all of them were eventually overcome, they retarded progress.

With persistence, hard work and leadership, Syracuse University gained through this project a useful tool its individual units wouldn't likely have been able to complete independently, that will greatly benefit the university for years to come.

Appendices

Appendix A: Screen captures demonstrating project

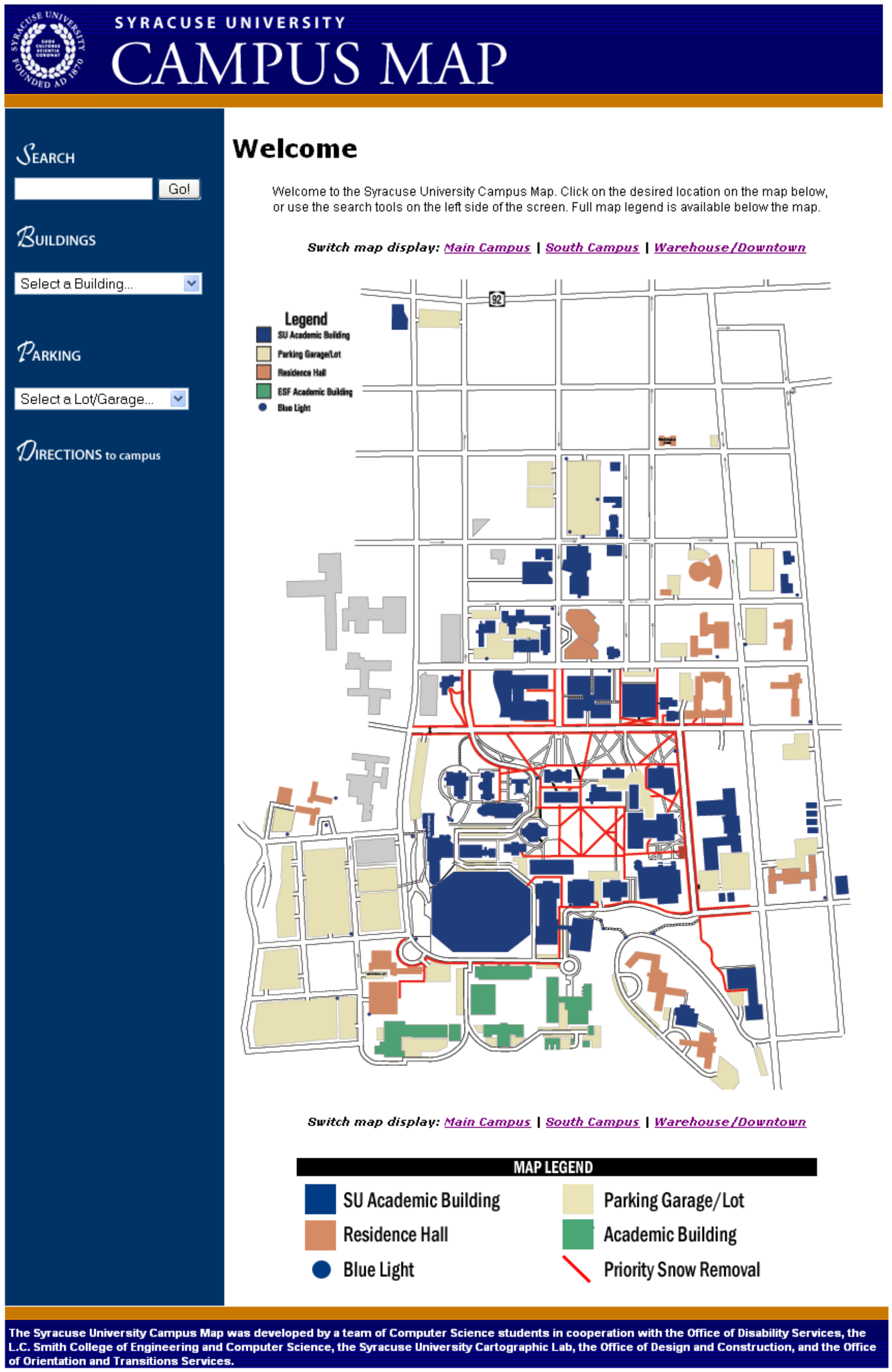


Fig. A.1: The entry screen for the project. The map is clickable.

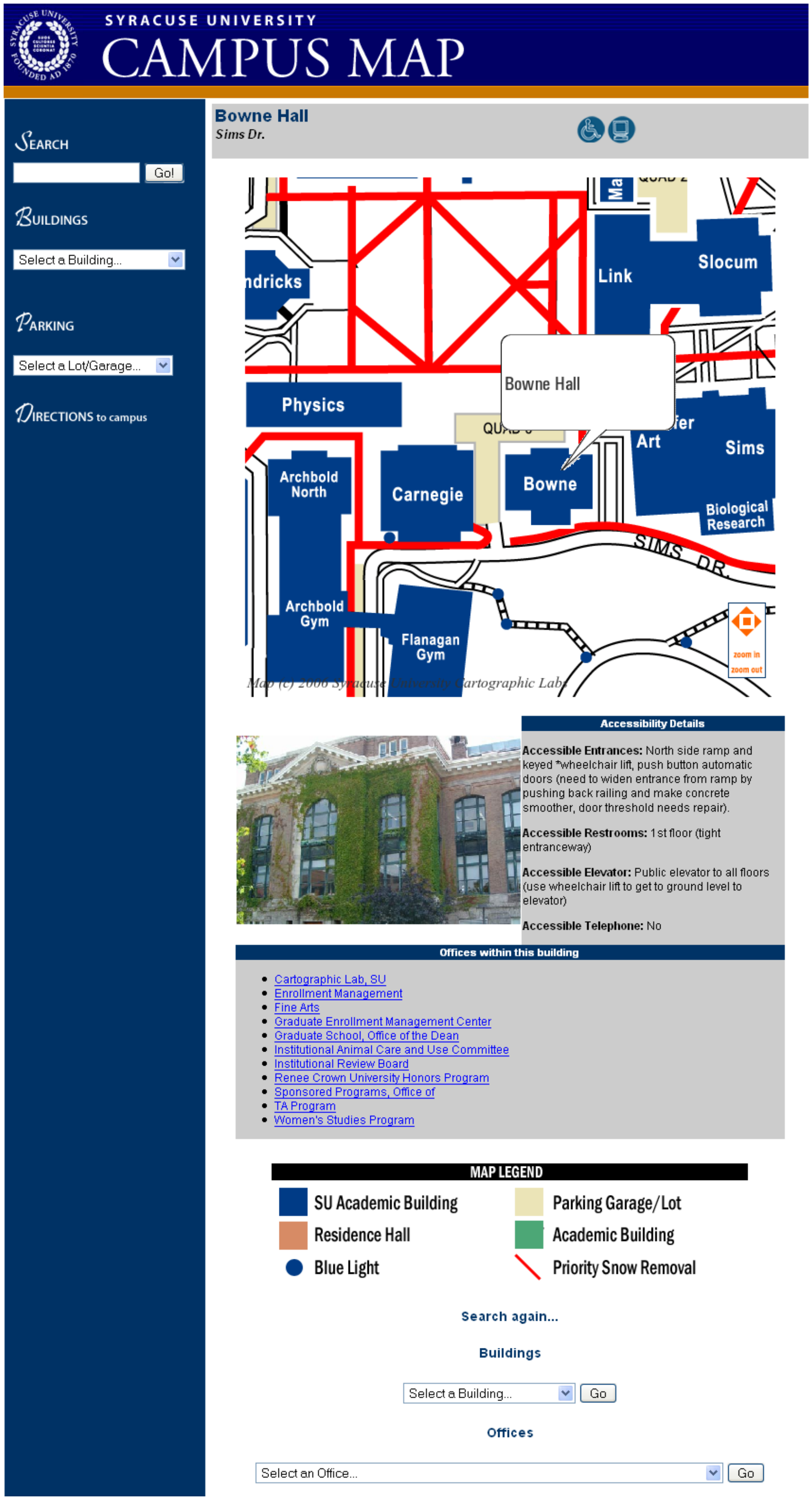


Fig. A.2: Results page depicting Bowne Hall.

SOFTWARE REQUIREMENTS SPECIFICATION

Accessible Interactive Campus Map

Dave Kazlman, Josh Killion, Aaron Knight, George Monioudis
CIS 453, Syracuse University, Fall 2005
Version 1.0

1. Introduction

1.1 Purpose of this document

Outlining the specifications of an interactive campus map of the Syracuse University campus, specifically geared to be accessible to the whole community with consideration of wheelchair access and web interfaces for the visually impaired. An interactive directory will allow the user to find exactly where they intend to go.

1.2 Scope of this document

To outline and specify the requirements of the Syracuse University Accessible Interactive Campus Map, with features recommended by the client, the Syracuse University Office of Disability Services.

1.3 Overview

The planned outcome of this project is an online interactive campus map that will provide directions from given points (similar to MapQuest or Google Maps) for people walking the campus. Grade of paths will be considered in options for those directions for mobility-impaired users. Special consideration will be given as well for visitor parking (i.e. for those visiting the university – driving directions from common entry points to parking lots, and walking/bus route information from those lots to other points).

1.4 Business Context

Funding for the accessibility characteristics of this project, including the procurement of a map from the Syracuse University Cartographic labs, are to be underwritten by the Syracuse University Office of Disability Services.

2. General Description

2.1 Product Functions

The Syracuse University Accessible Map project is designed to provide an accessible and usable map and directions system that ties together graphical maps as well as written directions to assist all users, especially those with accessibility issues.

2.2 Similar System Information

Outcome product will be designed to be platform-independent, only requiring a modern graphic-enabled web browser (such as Microsoft Internet Explorer, Mozilla Firefox, or Apple Safari) for the viewing of the maps themselves.

2.3 User Characteristics

Software is designed for the general university community, including visitors of the campus. Specific resources will be provided for general accessibility as well as those who are vision-impaired and mobility-impaired.

2.4 User Problem Statement

A major obstacle is to come up with a way to write directions for visually-impaired users without giving visual references - i.e. other buildings.

2.5 User Objectives

- A useful and easy-to-use interface, that looks consistent across multiple browsers and platforms
- Ability to give directions from any campus position to any

other on-campus position

- Ability to give directions useful to those bound to wheelchairs (i.e. excluding steep inclines and stairways)
- Ability to provide directions that are useful to those who are visually impaired
- Un-cluttered interface that is feature-filled but not confusing
- Ability to provide directory information for buildings, and provide at least simple directions within buildings (or appropriate wheelchair-accessible entryways for those locations).

2.6 General Constraints

Common web technologies that are cross-platform are essential to the success of this project. The project's written outputs, such as directions, must be readable by screen-reader software such as JAWS, the common standard at Syracuse University.

3. Functional Requirements

3.1 Graphical Database Search

3.1.1 Description

This function is designed to search the database of graphical area top down and find the one the user has selected by using helper functions to determine which element of the SQL database is to be selected. The input comes from the user's selection, Point and Click, Drop Down Menu, or Keyword Search. The output will be the graphical area chosen by the user in a new larger sized image.

3.1.2 Search Methods (Helper Functions)

3.1.2.1 Point and Click

This is a helper function to the Graphical Database Search, which allows the user to directly point and click onto the large-scale map to bring a larger image of one specific area of the campus. The input is direct clicking of the mouse on the area to enlarge. The function will return to the Graphical Database Search the location of the keyword associated with the area of the large-scale map selected by the user.

3.1.2.2 Drop Down Search

This is another helper function to the Graphical Database Search, which allows the user to select a location from a pre-defined drop down menu on the Interface of the webpage to allow the user to bring up a large-scale map of the area selected. This function will then return to the Graphical Database Search the location of the area to be displayed.

3.1.2.3 Keyword Search

This is the final helper function to the Graphical Database Search, which allows the user to type in a keyword location to search for. The keyword will then be sent to an error check method, which will search through the database for a matching keyword associated with an area to be displayed. If the keyword is found in the database the location of the associated keyword will be returned to the search function to be displayed. If the keyword is not found the function will return the closest matching keyword.

3.1.3 Criticality

This function is critical to the success of the project, as the search is the primary interface for generating an appropriately detailed map for the user.

3.1.4 Cost and Schedule

To be implemented Spring 2006. Cost will be negligible unless a surveyor is required to obtain data. Adaptable functions are already available.

3.2 Direction Search

3.2.1 Description

The directional search is a guide allowing user's to select two different locations, the first being a leaving point and the other being an arrival point, by means of a drop down menu or a keyword search. If the user uses the drop down menu, the database will be searched for the textual directions from one location to the other. If the user uses the keyword search, each keyword will be sent to the error check method and use the returned location to search the database for the location from one location to the other and display the Best Route or Handicap Accessible Route.

3.2.1.1 Method One: Best Route Direction Search

This will be a ranked map, which will determine the best route based on information that the user's input has provided and also depending on the date (to emphasize priority shoveling routes and avoid seasonally closed paths and stairways), path slopes, and shortest paths.

3.2.1.2 Method Two: Handicap Accessible Route Search

This will prioritize routes which are easier for those with a disability and will also be a ranked algorithm, weighing features such as date, path slope, shortest paths, building access ramps, and safety features. It will also select the appropriate accessible entrance (if a location within a building is provided by the user) for the mobility-impaired or visually-impaired user to use.

3.2.2 Criticality of Function

Vital to the success of this project.

3.2.3 Technical Issues

Algorithm should be easy to implement. Most difficult portion of requirement will be building of database with topographical data to get the slopes of all campus paths.

3.2.4 Cost and Schedule

To be implemented Spring 2006. Cost will be negligible unless a surveyor is required to obtain data. (Unlikely)

3.3 Error Check

3.3.1 Description

Uses the Keyword sent from the search functions to search the database for a match and return the matching location name if found, or return the closest matching location name. Matches to a degree of error the search keyword provided to entries in the database.

3.3.2 Criticality of Function

Important for the success of this project, although will be difficult to implement given allotted time and vast array of possible errors in user textual input fields.

3.3.3 Cost and Schedule

To be implemented Spring 2006. Cost will be negligible.

3.4 Zoom Functions

3.4.1 Description

This method allows the user to enlarge the area map on the graphical website, give the larger context of a detailed area, and re-center displayed maps based on click input.

3.4.1.1 Toolbar

A toolbar on the side of the map will allow the user to control zoom levels of the displayed map and re-center position. The toolbar will have a limited scope which will allow the user to zoom in or out up to the pre-defined range.

3.4.2 Criticality of Function

This function is critical to the success of this project. Use of Google Maps APIs and Source Code (available freely on the internet; see References [section 10.2]) may make this a simple function to implement.

3.4.3 Cost and Schedule

To be implemented Spring 2006. Cost will be negligible unless a surveyor is required to obtain data. (Unlikely)

4. Interface Requirements

4.1 User Interfaces

4.1.1 Graphical User Interface

The user interface will consist of a graphical web page. The webpage will consist of a single graphical map. The cross-sectional map displayed will have a single click attribute, which will allow the user to navigate to a directional, and graphical representation of the location of the specific building. The interface will also allow the user to navigate from one specific location to another. The interface will be database driven, and scaled to be available across different platforms.

4.1.1.1 End User Interface Requirements

4.1.1.1.1 Display Map

- Initially, the user will be presented with an overview map of the main campus. As the user searches for buildings and directions, the displayed map will change to represent a closer view or a traced path of the directions that the user has requested.

- Display directions from one location to another.
- A direction engine will analyze possible paths for the user to take from locations that they select. Options will be provided for the user to provide parameters – such as lower-grade paths, paths that are shoveled clear of snow with higher priority, or paths that go via other buildings. Directions to specific entrances (for example, wheelchair-accessible entrances) will be provided as an option.

4.1.1.1.2 Option to display best route for mobility-impaired users.

- This option will provide lower-grade path directions and/or directions to nearby CENTRO bus stops (with bus schedules) for campus shuttles.

4.1.1.1.3 Option to display best route for visually-impaired users.

- This option will provide path directions that include physical landmarks that a visually-impaired user would find helpful.

4.1.1.1.4 Option to display a picture of the building in question for visual reference.

- This option will provide a photo of each building, such that the user will be able to recognize a where they are going.

4.1.1.1.5 Option to Search for an office to verify the location of an office.

- A maintained database will be searchable by the user in order for the user to make sure that they are going to the correct building or buildings.

4.1.1.1.6 Option to Keyword search the database in order to allow for quick access; minor error correction in this Keyword search will be necessary to assist the user in the case of typos and the like.

- This will allow the user to search for various services, without knowing the proper name of the office or the building. Similar to how search engines such as Google and Yahoo!, this search engine will allow for minor error correction (for example, a search for “teh” would return results containing “the”).

4.1.1.2 Administrative Interface Requirements

4.1.1.2.1 Easy to use interface which allows quick updating.

- The interface will be designed such that the administrator need not have extensive web development experience, simply the ability to use a web browser and understand normal computer skills. The administrator will be able to edit office information, add new buildings as they are constructed (with the addition of an updated map file, of course, containing the new building itself).

4.1.1.2.2 Option to change season for directions – e.g. Closed stairways.

- An option will be provided where paths such as stairways can be marked as seasonal, so that directions aren't provided in winter months that require them. A notable example of such a stairway is the set of stairs on the north side of Crouse College, which is closed during the winter months.

- This feature should be automated if possible for easy maintainability.

4.1.2 Map Requirements:

- Single cross-sectional map displaying all buildings located on campus. It will likely be in the form of two maps, one each of main and south campus. A small inset map will likely be required for directions to the new Downtown campus building.
- All roads as well as walking paths and sidewalks displayed. Curb cuts will also be shown for mobility-impaired users benefit.
- One-click interaction
- Up close graphical views
- Handicapped-accessible entrances (and, for buildings without direct accessible entrances, indications of connected buildings with such entrances).
- Precise accuracy in order to generate accurate directions.

5. Performance Requirements

5.1 User Performance Requirements

The user's performance will be dependent on the speed of their Internet connection as well as the performance of their computer. The project will be designed to be compatible with all major Internet browsers and for a low common denominator of computer performance, so as to be widely accessible.

5.2 Server Performance Requirements

The server limits the numbers of connections based on available bandwidth and processing power. The page will be fully loaded at one time, so the time involved here will depend on the user's system. The actual test response time, the time between user action and browser response, will be uniformly responsive. For example, when the user makes his/her selection, the time it takes the browser to respond should be somewhat uniform between system configurations.

6. Design Constraints

6.1 Standards Compliance

Should be compliant to federal web page accessibility guidelines, Office of Disability Services accessibility standards, and World Wide Web Consortium (W3C) standards if possible.

6.2 Hardware Limitations

Must be designed for common computing platforms. The end product is to be designed in order to be usable on common browsers such as Microsoft Internet Explorer, Netscape (v. 4+), the Mozilla family of browsers, Apple Safari, and Opera.

7. Other non-functional attributes

7.1 Security

No security measures are expected to be involved excepting the web modification interface for maintenance, which will be protected. The database will be password protected, but otherwise not accessible except via the Administrative Interface.

7.2 Binary Compatibility

Web-based application, will not require client-side software save a modern web browser and an active Internet connection of some kind.

7.3 Reliability

Based on up-time of server on which application resides. Regular backup of database, or backup server for application, may be required to ensure up-time and reliability.

7.4 Maintainability

A web-based database management engine will be developed for the maintenance of the database.

7.5 Portability

As a standards-driven web-based application, portability should not be an issue.

8. Preliminary Schedule

Specification and design will occur Fall Semester 2005. Implementation will occur in Spring Semester 2006.

9. Preliminary Budget

Funding provided and approved by the Syracuse University Office of Disability Services as necessary.

10. Appendices

10.1 Definitions, Acronyms, Abbreviations

W3C – World Wide Web Consortium, an international web standards organization.
mobility-impaired user – Wheelchair-bound students and visitors to Syracuse University
vision-impaired user – Minimally a partially blind student or visitor to Syracuse University. These users require the ability to use text-to-speech software such as Freedom Scientific's JAWS or the Apple Computer's VoiceOver utility for viewing web pages.
JAWS – Screen-reading software from Freedom Scientific. The Windows platform software of choice for screen reader as defined by the Office of Disabilities Services.
SQL – **S**tructured **Q**uery **L**anguage. Namesake of a variety of database systems, including the free and open-source mySQL database system.

10.2 References

Document format adapted from the Software Requirements Specification Document Template, which is publicly available at http://www2.ics.hawaii.edu/~johnson/413/lectures/5.2.html

Google Maps APIs and documentation are available online: http://www.google.com/apis/maps/documentation/

An open-source implementation of Google Maps: http://www.clasohm.com/blog/one-entry?entry_id=13663