ReVamp UMD

Authors
Brendan Fruin – bfruin@umd.edu - WebDesign, GoogleWeb Dev, and Google Maps API Usage
Chris Page - cpage3@umd.edu - Bug fixes, Database Manager, and Back-End Structure
Jared Greene – jegreene@gmail.com - GPS Coordinates, Documentation, and Prototype Design

Students, Faculty, and Visitors need a map system that will be quick, easy to use, and recognizable. It is imperative that the interface be recognizable and easy to use as though the product would be heavily used, if users do not see a recognizable interface they are very likely to use an older but more intuitive system. Thus in order to create a recognizable system, the Google Maps API was used to create a map of the University which would include bus routes, building locations, and available parking. As parking was such a highly requested feature valid parking locations would be shown to users based off of the type of user (student or faculty) and what building they would be at. Finally, as the map would be hosted on Google's Servers, finding directions would be much quicker than the past edition.
Introduction

Overview of the Problem

Currently the University of Maryland's website has two interactive maps for users. The older of the two maps is the “Interactive Map of College Park” is frequently the first choice by users. The map is fast, and easy to use, but lacks the basic map-navigation that many users expect after experience Google Maps. The second, newly released map, is the “TerpNav Pedestrian Mapping System.” Most map users are now familiar with the interface that Google Maps has set as the norm. The creators of “TerpNav” attempted to reproduce the Google interface by using a competitor of Google Maps, called OpenStreetMap. This system was chosen over competitors like Google Maps because the APIs were more readily available, and to follow mandated security issues. However, when TerpNav is compared to a competitor, like Stanford's Map it is obvious that UMD's mapping system needs to be redone.

There are several important features that TerpNav lacks which are critical to the success of creating a successful map system. This data is based primarily off of the teams own experiences of using the OpenStreetMap system and various other Systems, as well as data from various testers that had used both Google Maps and TerpNav. TerpNav was designed without taking into consideration that users engaged with the map do not wish to move their hands back and forth between the keyboard and mouse, and as such, users prefer to use the mouse’s right click option for shortcuts. Though TerpNav is heavily JavaScript based, and therefore has the ability to use the second mouse option, nothing was bound to it. Therefore, the map had an ultimate design that had the user moving back and forth between the mouse and keyboard. However, Google Maps has several options for the user primarily based off of where the user right clicked on the map. Another major problem with TerpNav is that it uses JavaScript that has been poorly optimized. JavaScript can have requirements on recourses from a machine, and although this problem is frequently avoided by modern machines, older computers cannot avoid the intensity of poorly optimized JavaScript. Thus, when TerpNav was explored with an older machine, the experience felt very rough and sluggish. It was difficult to drag and explore the map as the images did not immediately refresh, and dragging the actual map appeared to have a lag. Actually using the interface was very confusing, as there are several options to choose from, which are fairly similar at a glance. When the system actually searched for the directions it took a fair amount of time to be presented. Google Maps, is quick, as it uses Google's algorithm and their servers. Google's JavaScript is heavily optimized as they have a wide range of customers that must be able to use their product. Lastly, TerpNav design is unappealing containing poor use of translucent boxes with colors that heavily contrast on each other, ruining the aesthetics of the design.
Previous Work

As discussed, the current map system, TerpNav, is based off of the infrastructure of an open source mapping system OpenStreetMap. This system was chosen over competitors like Google Maps because the APIs were more readily available, and privacy issues. Particularly the entity of OpenStreetMap could be compiled and hosted directly on Maryland’s servers, allowing for Maryland to not be dependent on any third parties systems. However, when TerpNav is compared to a competitor, like Stanford's Map, it is obvious that our mapping system needs some work. Though the maps are similar, it is from a coder’s perspective we understand that OpenStreetMap simply is not capable of being optimized to compete with Google Maps.

Another example of a mapping comparison and how it affected users is comparing MapQuest to Google Maps. MapQuest had a system similar to that of the “Interactive Map” on the university web page. Although the actual map was difficult to navigate or follow, the directions were simple and easy to understand. However, as Google began implement satellite imagery, and made exploring the map quick and easy, users began to massively switch systems. Today the top competitors for online mapping are Microsoft’s Bing and Google Maps.

The movement in users and demonstrates how critical an excellent user interface is to launch a success mapping product. As the main interface and focus of the map, is the actual map itself, it is important to keep all outlying features close to map (in terms of distance, so users can see everything at once), as well as making the interface simple to use (so users don’t get overwhelmed and distracted), and quick (in order to keep the users attention).

Relevant Sources

Google Earth Creator Sees Many Potential Benefits - Brian McClendon creator of Google earth discusses its users. The site has the feel of an atlas and a flight simulator combined. Mountains and valleys are shown in high-resolution 3D. Demonstrates how in the mapping world simple is not always better, as many users enjoy the 3D interfaces.

Interactive Map of College Park - This map is the map that most students and faculty are currently the most familiar. It has a small display area and does not help the user find directions. Instead users are forced to find a corresponding building number that matches on the map with the building. It is hard to gain useful information from it and is our baseline for what we are trying to improve.

MobilEdu. Product Overview - MobilEdu creates mobile software for university systems. Observing their designs has helped in giving our team ideas in implementing a similar project for the University of Maryland. They designed the Stanford Map, and its interface looks quite appealing, aesthetically and seems simple to use.
Field Studies: The Best Tool to Discover User Needs - Field studies have an immeasurable value in designing an interface geared towards the user. This article outlines how you can use the information gained in a field study in order to help build your use cases and requirements.

TerpNav Pedestrian Map - This is a map that University of Maryland currently has that satisfies many of the goals that we are striving for. This includes being able to locate a building through a search box and finding the shortest distance between two on-campus locations. While this is a helpful map, almost all the people we questioned did not know of its existence, nor understand the interface.

HCL Maps Set in Stone - David Cobb and his staff in the Harvard Map Collection was requested to design a map for the North End parks that would illustrate how Boston had changed over the centuries. Cobb found that geo-referencing allowed maps to be accurately oriented so they line up with other maps.

Google Maps API Tutorial - The Google Maps API can be confusing especially with a limited knowledge of JavaScript. This page provides tutorials on how to embed Google Maps into your page and customize it to accomplish the task at hand. It helped the team to understand that with Google Web Development Toolkit, the process would be greatly simplified.

Presentation of Design

Overview of Approach and Solution

The goal of the project was to create a new map system for the University that was fast, easier to use, would keep some of the important features implemented in TerpNav, and add features to accommodate users that have parking and bus problems. The team decided to use Google Maps to implement the redesign as we felt it would best accommodate our requirements to complete our goals.

The Google maps infrastructure would help solve a lot of the initial speed and optimization problems that were reached when using the TerpNav system. Documentation on the Google Maps API is so readily available and thus would help our team with the deadline limitations. As Google Maps has around 90 million users, it is very likely users of our system that implemented the Google Maps API would have experience, which would greatly reduce the learning curve. This is important as although the frequency of use of our map would likely be very high, the amount of time users would wish to spend using the product is fairly low. A recognizable system/interface would appease these users, making them more likely to return in the future. Thus, our design similarly followed that of Google Maps, having a left pane navigation and top header.
This image shows the Transition Diagram of our Map. As our webpage is dynamic, the diagram shows that any time a user can clear the page (shown) and reach the help page (not shown). It is our belief that most users will use the map starting with an empty map before activating a feature. However, although the transition diagram does not show this (it would be impossible to follow) every feature can be reached while another feature is being used. Therefore the diagram is in reality a web of positions all linked with each other. Parking is the only feature that is dependent on another feature (find building). This is the case as during our user testing, individuals wanted to find parking close to the building. In future builds, with more parking locations mapped out, these feature would certainly be based off of location (in the last revision of the prototype all parking is shown).
This figure shows the information the user will initially see after loading the application. The help page link and reset button are located on the header in order for the user to easily notice the links, and be aware of their presence while running the application. The side navigation contains most of the tools to interact with the application. Our team organized the features in order of what we believed would be most used to least used. The map, of the campus which is outlined to aide the user in use, is by default centered on campus, and uses the satellite imagery.
Figure 2 – Find Building

As a user begins to type a name of a building, or a building code, auto complete will allow the user to finish typing the building or once a user see's the building they are looking for, they can click the auto complete text and the text box will be filled. After the text is filled, the next time a user clicks or moves the mouse the map will put a red pin on the map to indicate where the building is located. This pin can only be removed with the clear page button.
Figure 3 Parking - Unrestricted after 4 p.m

When this checkbox is marked it will show blue “P” markers above all the parking that do not have restrictions after 4:00pm
When this checkbox is marked it will show maroon "P" markers above all the parking that have restrictions.
This option is similar to the find building option. The main difference being that after a user types the second address, directions will show (at the bottom of left the navigation bar) and a driving route will be displayed on the map.
This option is the same as the driving directions. The main difference being that if a user checks the walking bubble, a direct route (that follows walking paths) will be displayed on the map and the approximate time and distance is displayed on the left navigation. As the feature is still beta (within Google maps) so a warning is displayed to notify the user.
This screenshot demonstrates the color overlay that appears when a user checks a bus route. Users requested different colors for the paths, so our team attempted to have overlays that matched the bus line names.
This screenshot demonstrates how multiple bus routes appear on a single map. As the prior version had one color for all routes, it would be confusing to see multiple routes. The different colors allows for users to easily compare routes in order to determine one they would like to use.
This link takes the user to the official University of Maryland webpage that contains all the bus information. In case our map did not suffice for a user's needs, they will be able to find what they are looking for (as far as schedule and path details) on this page.
This feature will allow users to draw a path of their choosing on the map. It will then show the distance and approximate the time it will take to travel based off of a predetermined average walking speed and distance. Users can remove the path after they draw it, instead of having to clear the whole map.
This screenshot demonstrates how when a path is removed, only the path is removed, and the data that was already on the map remains.
Simple screenshot that should be used to compare with the next. This screenshot demonstrates a map that is full of points and data, and that is about to be cleared...
After the Clear Page icon is clicked, the map is cleared successfully and all data on the map will be removed. We chose to leave the navigation bar with the data as is, as we felt users might want to change minor details, and thus having the data remain would make it easier on the user.
This page explains all the features and details of the map in more detail for the user. It brings the user to a new page in order to not overwhelm the user and so that a user could potentially have both the help page and map page open at once. The Help page is explained in detail in the next section.
Help / Tutorial

In order to better serve our users, the help/tutorial is also accessible via the prototype.

UMD Maps is a tool designed to replace the University of Maryland's current parking map found here. This map allows you to find buildings and parking on campus, find directions between buildings, display bus routes/times and draw your own path. It is built using Google Maps which still has some features in beta testing. This means some of the direction features may not be fully implemented. This is only a prototype so many of the buildings/parking lots have not been added due to time constraints.

Find Building

"Find Building" helps you to find a building on the University of Maryland's campus by entering either the name of the building or the building code. (Building codes can be found here.) Once you begin to type the name of the building (or its building code), a drop down menu will be displayed with suggestions. Find and click the building name that you are attempting to locate. After clicking the building name, a marker will be displayed on top of the building selected. You can continue to locate buildings and add as many markers as you wish. To clear all markers click the "Clear Page" button located above the map.

Show Parking

"Show Parking" gives you the option of displaying lots for those with no permit and those with a teacher/staff permit.

Click the "Unrestricted after 4pm" checkbox if you wish to display the lots only limited to permit holders between 7 a.m. and 4 p.m weekdays. These lots are free to the public at all other times including weekends with the exception of football and basketball gamedays. These exceptions can be found on Maryland's athletic events website.

Click the "Staff/Faculty after 4pm" checkbox if you wish to display the lots limited to permit holders only between 7 a.m. and 4 p.m weekdays. Faculty/staff displaying Faculty/Staff permits are permitted to park in these areas at all other times.

Directions

The directions feature allows you to find driving and walking directions between two buildings on campus. Begin by typing the building name or building code into the text box labeled "A". (Building codes can be found here.) As you begin to type the building name (or building code), a drop down menu will be displayed with suggestions. Find and select the building name you are attempting to set as your start location. In the text box labeled "B" type the name of the building or the building code of the end location.
As you begin to type, drop boxes will appear with suggestions as they did in the text box labeled "A". From the drop down list, select the building name that you would like to set as the end location.

Walking directions will be displayed as the default. Walking directions will display the fastest path that Google can calculate along major paths and roads. The walking directions are still in beta testing which means that many paths will not be taken into consideration when calculating the fastest path.

Driving directions can be selected as an alternative to walking directions. After entering in a valid start and end location, selecting driving will display the directions from building "A" to building "B".

Both walking and driving directions will display a panel on the bottom left hand side of the page displaying step by step directions in plain English on how to get from the start to the end location.

**Bus Routes**

"Bus Routes" displays some of the evening shuttle routes available to students/faculty of the University of Maryland. Clicking any of the lines/colors will display the bus route in their respective color on the map. You can display multiple bus routes on the map at the same time if necessary. If you are unfamiliar with the stops or wish to find out bus times, a link is located below the different lines with the schedule.

**Draw Your Path**

Due to Google Map's walking directions being in beta testing, it may be necessary to draw your own path. If you wish to draw your own path, simply select the "Path" button. After selecting the "Path" button, the cursor that was once a hand will change into a target and the button will now read "Remove". Simply select the points on the map that you wish to draw a path between. After selecting two points, a line will automatically be drawn between the two points and the distance calculated. The total distance in meters and the estimated walking time will be displayed directly below the button. (Walking is estimated at approximately 1.2 m/sec and total estimated time is displayed in the format MM:SS.) You can continue to add points and extend your path for as long as you wish and the total distance and time will update automatically. When you are finished adding points simply double click on your final point and the target cursor will change into a hand cursor. To remove the line that you drew simply click the "Remove" button and the path you created will be deleted from the map.

**Clear Map**

If at any time it is desired to clear the map, clicking the "Clear Map" button will clear the direction line, paths or markers displayed on the map. All checked bus routes will also be unchecked.
Report on Development Process

Low-fidelity Prototype Screen

We created two low-fidelity prototypes based on two different map sites. One had most of the input above the map while the other had most of its input to the left of the map.

"Upper Content" Prototype

The first prototype we designed was referred to as the "Upper Content" prototype. This prototype is similar to the Stanford implementation in the sense that most of the input locations and feedback are located in the upper portion of the screen. The input locations are ordered with what we believed the users would find the most important on the top and the least important on the bottom. We believed Find Building / Show Parking would be the most important and placed this on the top of the page. Directly
underneath we put directions which we believed had the next most importance. At the bottom we put bus routes which we thought had the least importance in comparison to the other features.

"Left Content" Prototype

The second prototype referred to as "Left Content" has the majority of the input locations located in the left portion of the screen. We believed the Find Building / Show Parking features were going to be the most important and most used features of our map. We therefore separated these features from the left panel and placed them above the map for quick access. The placement of the directions panel on the left hand side is similar to Google Maps.
Based on user tasks and our expected users we decided that the "Left Content" prototype was the better choice. We reached this decision based on our belief that expected users would have some familiarity with Google Maps and would find our map intuitive if the layout was similar to Google Maps. We created several white board prototypes using the "Left Contents" prototypes in order to show the typical user interacting with our site.

**White Board Prototypes**

The following movies give different scenarios of a user interacting with our prototype map site. Several schemes were created and performed on a white board so we could better understand how well our design worked. After numerous attempts and changes made to our prototypes the following design maintains the most simplicity, while accomplishing a users task. The primary tasks described below each corresponding to a recording.

*Note:*

Person in white will correspond to a person using the interface.

Person in black will carry out the actions of our site based on the user input.

**Find Building 1**

*FindBuilding1.mov*

The user is interested in only locating a building on campus and finding the closest parking area to it. It was difficult using the whiteboard to show how we will implement displaying reserved parking for user events. Our plan is to have a text area in the bottom left corner of the screen that will show possible conflicts.

**Find Building 2**

*FindBuilding2.mov*

The user decides to locate another building. The building and parking locater are updated.

**Directions Walking**

*DirectionsWalking.mov*

The user is acquiring the path from one building to another (DEFAULT Walking option). The shortest path is displayed.
Directions Driving

DirectionsDriving.mov

The user is obtaining the path from one building to another via an automobile. The road(s) to the closest parking area of the destination are outlined. Next the walking path is displayed from the parking lot to the destination.

Something not implemented in our demo, but which we hope to implement if possible, is the ability to change recommendation for parking zones based off of events.

Display Bus Routes

DirectionsBus.mov

The user wants to display two different bus routes while the driving directions still appear on the map. The user first clicking the "Blue" line checkbox and it then being displayed on the map shows this. The user then selects the "Green" line checkbox, which is then also displayed on the map. This shows that directions can be displayed along with multiple bus lines.

Bus Schedule

BusSchedule.mov

The user has directions and two bus lines displayed, but is unfamiliar with the bus arrival times for the green line. To figure this out, the user simply clicks the "Green" text next to the checkbox, which opens a separate window displaying the green line's schedule.

Process for Arriving at the High Fidelity Prototype

After deciding that we were going to use the "Left Content" prototype, it was difficult in deciding exactly how we were going to implement it. We first had to choose from the various free map APIs currently available. We quickly narrowed it down between OpenStreetMap and Google Maps since these were the only two free map APIs that we had heard of. Due to our experience with Google Maps and its extensive library of functions, we decided that we would implement a map running on Google Maps. After researching various implementations of Google Maps, we learned that we could employ Google Maps using Flash, Javascript or GWT. Flash was quickly eliminated as an option since none of us had any prior experience with it. Google Maps using Javascript appeared to have the most documentation and tutorials. On the other hand, we knew that we wanted building information to be easily maintained and updateable
on a database and that the Google App Engine using GWT would be a great way to implement such a
database. It was decided that GWT was the best option for creating our map and we were successfully
able to turn our "Left Content" low-fidelity prototype into a testable high-fidelity prototype.

**High Fidelity Prototype**

Our high fidelity prototype has most of the functionality that we outlined in our "Left Content" low-fidelity
prototype. It has directions and bus routes to the left of the map and find building and show parking above
the map. Markers are placed on the building locations as they are typed in and parking icons are
displayed above all the different parking lots. Clicking the different bus lines will display at least part of the
bus route (full bus routes visible in final prototype). At this point we were unable to implement the walking
directions option. The only directions that the user can get in the high fidelity prototype are driving
directions. The feature of the bus schedule being displayed when clicking the bus color is also not in this
prototype. We believed this could potentially lead to the user being accidentally redirected to bus
schedule pages. Nonetheless, this high fidelity prototype was very useful in the usability tests for finding
the strengths and pitfalls of our map.

**Usability Testing Process**

For our usability test, we asked friends and neighbors to help us test our high fidelity prototype. The
individual was first asked to complete a pre-test. The pre-test asked important questions such as their
relation to the University of Maryland, familiarity of campus and the current map, and whether they were familiar with Google Maps.

Image:Revamp-pretest.pdf

The individual was then asked to complete the following task list.

1. Find one of the following buildings: Adele H. Stamp Student Union, Clarice Smith Performing Arts Center, Computer & Space Sciences Building, Computer Science Instruction Center or Tydings.
2. Show parking close to the building found.
3. Get directions between any of the two buildings listed above.
4. Display the blue line.

After the user had finished this task list they were given a post-test. The post-test asked the person to rate each item along a 1-7 Likert scale (7 being very pleased):

1. The speed of the prototype
2. The appearance of the prototype
3. The "Find Building" feature
4. The "Show Parking" feature
5. The "Directions" feature
6. The "Bus Routes" Feature

The user was then asked to rank Find Building, Show Parking, Directions and Bus Routes in order of importance to them.

The final question was a free response for any recommendations or general suggestions that would help to improve the usability of the site.

**Subject’s Experience and Outcomes**

**Participants**

Test Subject #1

Test Subject #1 is a senior dance major at the University of Maryland. She transferred to the University of Maryland as a sophomore. She has never lived on campus, rarely gets lost on campus and sometimes uses the campus bus. Though she is familiar with the parking map, she is not familiar with Google Maps.
Test Subject #2
Test Subject #2 is a junior at the University of Maryland. She is a female and a graphics design major, and transferred to the University of Maryland as a second semester freshman. She has never lived on campus, rarely gets lost on campus, and often use the campus bus. She is not familiar with the parking map, but is very familiar with Google Maps.

Test Subject #3
Test Subject #3 is a freshman history major at the University of Maryland. He has lived on campus in the dorms, and in a fraternity house, rarely gets lost on campus, and rarely uses the campus bus. He is both familiar with the parking map and with Google Maps.

Test Subject #4
Test Subject #4 is a mother of two students at the University of Maryland. She is 59 and has been on campus for various activities with her children that currently attend the University. She has never lived on campus, gets lost quite frequently when she visits, and only rarely had used the campus bus. She is not familiar with the parking map, but is very familiar with Google Maps.

Test Subject #5
Test Subject #5 is Teacher at the University of Maryland. He is 42 years old and has been teaching for more than five years. He has never lived on campus, rarely gets lost, and rarely uses the campus bus. He is very familiar with the parking map and Google Maps.

Test Subject #6
Test Subject #6 is a senior Finance major at the University of Maryland. He has lived on campus for four years, living in the dorms his first two years and moving into the South Campus Commons for his Junior and Senior year. He only gets lost when he looking for buildings he has never been to, uses the bus frequently at night, but rarely uses the bus during the day. He is familiar with the parking map and Google Maps.

Graphs displaying important information on participants:
Participants Comments

The following are the comments made during the test by each participant, the problems that they ran into while completing the task and their suggestions for improving the page.

Test Subject #1

- Typed cspac for find building and nothing came back because PAC is the building code
- “Whoa that’s cool! I could actually use this except I already know most of campus.”
- Believes find building should be on the left
- Wants direction color to be changed.
  - Recommended directions line be another color like red
- Wants logo to be added to the top of the page.

Test Subject #2

- Didn’t understand the whole “find building” feature, tried to point on map with finger where stamp was, but eventually realized the directions and saw the feature
- Typed parking into find building at first then saw checkbox because she did not see the P icon at all after clicking the checkbox.
- Wants a perimeter around the campus.
Test Subject #3

- “I like how quick and easy it found my building, also the auto complete is very cool and impressive.”
- Liked the layout of the parking, but suggested a larger more obvious marker.
- Initially had a bug when attempting to find directions (typed in a nonexistent building), but the user refreshed the page and the second time was able to complete it.
- Stated that he felt the icon for parking was too small.
- Did not like how bus routes were the same color as directions
- Wants to see reverse look-up (clicking the building and finding it)

Test Subject #4

- Parking icons are too small
- Subject liked how parking showed you what time it was open, thought “P’s” were too small
- Did not like the similarity in color of bus routes and directions.
- Did not realize you could zoom. Interviewer informed her she could and she realized it and “wanted to take this critique back”
- Likes interface, simple and straightforward.
  - Suggests a list of the buildings for users, or maybe if you click a building you can know what it is.
  - Liked how it was like “facebook” and found your buildings with autocomplete
- Critique as an average user towards bus routes (does not understand what the bus routes are)
  - Can she use them
  - Does she need an ID
  - What are the details
- Finally wants directions to say driving directions

Test Subject #5

- Liked the layout of the parking, noticed how it did not show restricted lots “I like how it shows parking, and the times if you click on it. However I have a parking permit, it should show the location that my parking permit works in”
- Likes Interface but wants everything on one side if it can still remain simple.
- Did not like how Bus routes were same color as directions.
- Wants to see reverse look-up (clicking a building and finding it)
Test Subject #6

- Test subject went through the test very quickly without any problems.
- Stated that he thought the parking icon was too small.
- Stated that when he started dragging the map around he did not like how he could not center it back at the University of Maryland
- Wanted to remove the building finder icons, but was unable to unless he refreshed.
- Likes interface but suggested that everything be moved to one side.

Summary/Results

Recommendations

- Need bigger parking icons
- Need more information contained in the map
- Need to outline the campus
- Reverse look-up of buildings
- Different color for directions and blue line
- More information on restricted parking lots
- More information for buses such as bus schedule
- Move “Find Building”/“Show Parking” to left hand side

Ranking of Features/Performance

In the post-test each participant ranked the speed, appearance and each feature of the high fidelity prototype. The following graphs show the results:
The following shows the average rating of speed, appearance and each feature that the participant tested.

<table>
<thead>
<tr>
<th>Issue</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Speed</td>
<td>6.83</td>
</tr>
<tr>
<td>Overall Appearance</td>
<td>3.67</td>
</tr>
<tr>
<td>Find Building</td>
<td>5.50</td>
</tr>
<tr>
<td>Show Parking</td>
<td>4.83</td>
</tr>
<tr>
<td>Directions</td>
<td>5.50</td>
</tr>
<tr>
<td>Bus routes</td>
<td>5.83</td>
</tr>
</tbody>
</table>

Average Ranking of Importance of Features (Each user ranked the issues from 1 (most important) to 4 (least important))

<table>
<thead>
<tr>
<th>Feature</th>
<th>Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find Building</td>
<td>1</td>
</tr>
<tr>
<td>Directions</td>
<td>2</td>
</tr>
<tr>
<td>Show Parking</td>
<td>3</td>
</tr>
<tr>
<td>Bus routes</td>
<td>4</td>
</tr>
</tbody>
</table>

**Revisions**

We received many recommendations on how to improve our map in order to make it easier to use and more valuable to the user. While we were not able to implement all recommendations due to time constraints, we were able to implement many of the features recommended.

**Recommendations Implemented**

**Make the parking icons larger** - We increased all parking icons displayed to be 50% larger than their original size. This has made a huge difference in the visibility of the parking icons as seen by recent usability tests.
Outline the campus - The campus is now completely outlined by a transparent red polygon.

Different color for directions and blue line - The blue line and the directions line are now two very different shades of blue.

More information on restricted parking lots - While we did not add a feature to display/find a particular restricted parking lot, we did add the ability to show parking lots restricted to only faculty/staff after 4 p.m.

More information on buses such as bus schedule - A link is now found beneath the bus lines taking the user to the various bus schedules.

Move "Find Building"/"Show Parking" to left hand side - "Find Building"/parking options have now been moved above the directions panel.

Conclusion

Final Status

Many of our primary task goals were implemented, however only to a moderate degree of completion. The primary reasons for not fully completing the project was time, current GWT Map limitations we were unaware of, and changes we had made throughout the design.

First let's get into the cool stuff that we actually implemented. As one views the website for the first time they will see the University of Maryland Map outlined thanks to a suggestion by a reviewer. To the left is a vertical panel with all of the primary functionality. It is structured from the more important features to the lesser based on the user evaluations. The first feature is find building. This will display markers on the map based on buildings selected by the user. The second feature implemented is directions. Users can select their travel mode as driving or walking in order to obtain more appropriate routes. The third feature implemented is the Bus Routes. On the click of a check box the route of a bus line is displayed on the map. Users can click multiple lines to see which bus route may better suite them. The final primary feature is draw path thanks to a suggestion by Professor Bederson. In this scenario a user can draw their path on the map from one place to another. It is not limited to a start and end and the path can incorporate sidewalks not recognized by the directions feature.

Future Work Possibilities

Below is a list of the elements we were unable to finish:

- Display places to eat with appropriate icons
- Lookup of any parking lot by name
- Have events synced into the parking lot Info Window
- Reverse lookup by clicking place on map and having it say which building it is
- Implementing all campus buildings, parking lots, and bus routes

Some of these elements would be quite easy to add. For example the building names and locations are bulk loaded to Google's appengine database. Without modifying any code future developers could just add more locations to a file and upload it to the server. Several elements are limited by Google maps. For example the walking directions are still in beta and not all of the paths are recognized. This leads to the walking directions been inefficient. Fortunately the project is implemented so as Google betters their walking implementation it will do so on our map. For now the draw path feature can aid in users getting the most efficient of paths.

**Recommendations to Future Developers**

Further developers could refine all of the unfinished features mentioned. Unfortunately the documentation for GWT and GWT maps API is currently lacking. It is primarily in Java Script and there are significant differences. In time the documentation should improve. The alignment in the left panel could be slightly adjusted from visual and usable improvement. The alignment in the upper panel with the help and clear button needs the most improvement. It just does not seem to flow well.

**Acknowledgements**

We would like to thank our friends, neighbors and family for helping with testing and improving UMD Maps. A special thanks to Ben Bederson for helping us resolve the GWT bug when using Safari 4.04. Without all your help, we would have not been able to evolve UMD Maps into the polished final prototype seen today.
**References**

*Google Maps 1.0 API.* Google. <Google Maps 1.0 API>

*GWT 1.6 API.* Google. <GWT 1.6 API>


*Interactive Map of College Park.* University of Maryland College Park <Parking Map>

Sielge, Del. "Likert Scale." University of Connecticut <Likert Scale>

**Surveys**

**Pretest**

*Relationship to the University of Maryland: Student Faculty Visitor*

*Complete the appropriate questionnaire based on your relationship to the University*

**Student**

What is your major?

How long have you attended UMD? semester(s)

Have you ever lived on campus?

Yes No

How often do you get lost on campus?

Very Often Often Sometimes Hardly Ever Never
How often do you use the campus bus?

Very Often       Often          Sometimes            Hardly Ever             Never

Are you familiar with the University's current interactive parking map?   Yes   No

Are you familiar with Google Map?

Yes                         No

**Faculty**
Which building/buildings do you work in the most?

How long have you worked for UMD?    year(s)

How often do you get lost on campus?

Very Often       Often          Sometimes            Hardly Ever             Never

How often do you use the campus bus?

Very Often       Often          Sometimes            Hardly Ever             Never

Are you familiar with the University's current interactive parking map?

Yes                         No

Are you familiar with Google Map?

Yes                         No

**Visitor**
What is your primary reason for visiting the University of Maryland?

Have you ever visited the campus?
Would you need assistance finding parking on campus?

Yes       No

Are you familiar with the University's current interactive parking map?

Yes       No

Are you familiar with Google Maps?

Yes       No