

CS 190 Revised Project Specifications Proposal
Amy Simpson (asimpson)
March 6, 2006
GeoEvents

1. Title: GeoEvents

2. Description

GeoEvents is designed for Rhode Islanders looking for something to do. Often people want to go out and do something but lack the motivation, time, or energy to put in the effort to actually find that something. Newspaper listings lack the ability to adapt to any of the specific needs of the user and can be hard to use. GeoEvents will be able to mesh the current events listings from the newspaper (the Providence Journal since it is Rhode Island based) with the Google Maps API to provide a much more interactive and intuitive way to look for interesting activities. One current example of a similar project can be found at <http://api.local.yahoo.com/eb/demo/>. GeoEvents will go beyond, however, and provide additional useful features (specified in section 3 below).

3. Feature Set/Priorities (*Rated 1-6: 1 being the highest priority*)


(The priority rating is a result of responses to questions from possible users and an attempt to mirror similar existing interfaces in order to increase the intuitiveness of the project's use)

- a. Google Maps API with flags marking the location of local events that qualify for the filter parameters provided. [1]
 - i. This will be a frame on the page containing a Google Map of the area. When GeoEvents is first opened up, the default map should be of Providence and all filter types of events should be included on the map. All events are marked by a flags which can be clicked for details (more details in section c below.)
- b. Be able to enter in a 'base' address (the address the user will be starting from, for example their home address or the address of their office if they are leaving from work [2])
- c. Clicking on an event:
 - i. an event's flag should expand to show the brief details of the event in a concise, readable, and consistent manner.
 1. Display the event location [2]
 2. Display the event time [2]
 3. Display the price of the event [2]
 4. Show the distance from specified base address (if there is one, other wise this field should not be included in the flag information) [3]
- d. An interface for choosing the search dates [2]

- e. Ability to filter event type (e.g. choose only Theatre & Museum) [2]
- f. Be able to get directions to a specific event [3]
- g. Display below the map window a list of the filtered events [1]

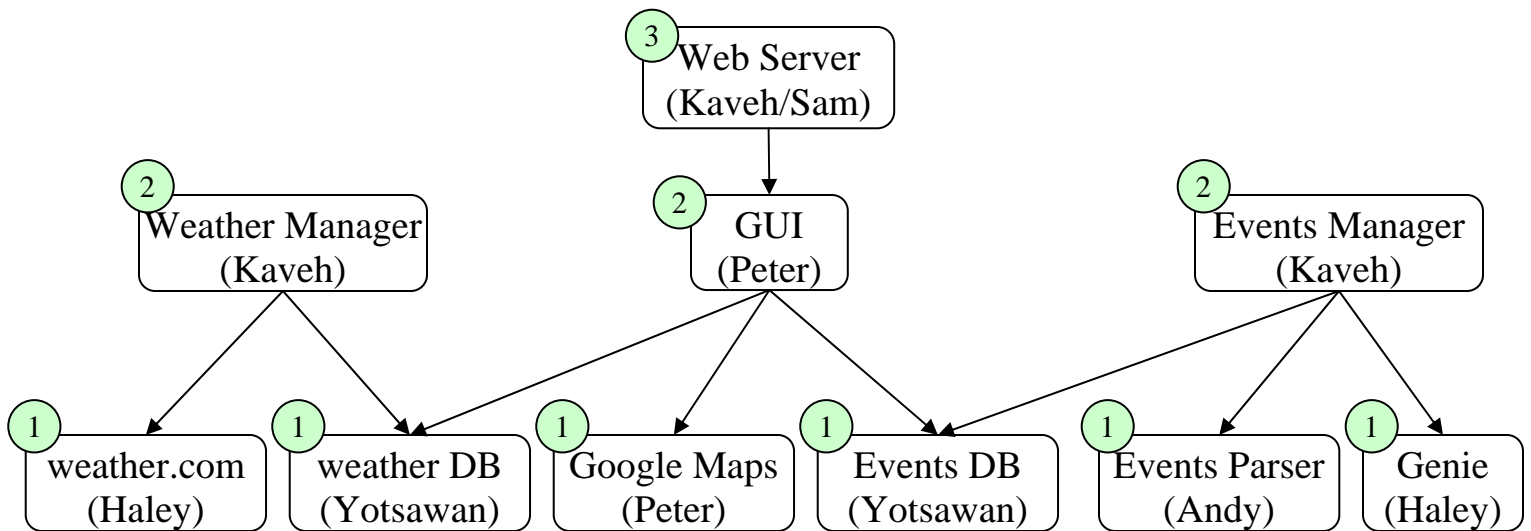
Optional Features (Priority four and over)

- (a) Have an interface that will learn from past usage of specific users and be able to 'suggest' events [6].
 - (i) Using past inputs (searches and events the directions were found for) from the user, GeoEvents would be able to find and 'suggest' similar types of events that the user might enjoy
- (b) Be able to select more than one date at a time [4]

The first priority (priority one) is having a map that shows and lists the events around Rhode Island. This is absolutely essential. For this project to have real usefulness, all items in priority two should be implemented. This includes event details including time, location, and price. The third set of priorities are extremely useful and what would make this project more useful than what is currently available since it would integrate the direction creating capabilities of Google Maps into the project, but is not essential to the overall functionality of the project. Beyond that, other items are helpful and added bonuses, but not what truly define the project. 



4. System Model Diagram



Web Server

This is the HTTP server that users will connect to via their web browser. This server will host the GUI code that generates HTML pages in response to user requests. The database(s) might also be stored on the same machine.

GUI

The GUI is responsible for formulating responses to user requests. This means that the GUI not only has to be able to take the input and decipher some meaning from it, but it also has to decide what to query for from the database(s) based on that meaning and then it has to use Google Maps to display the info it gets from the database(s). The GUI is basically the glue of the whole project ties all the separate pieces together.

Weather Manager

The Weather Parser Manager is in control of deciding when to gather new information about weather. When the time comes, the manager uses one or more parsers to collect weather data and update the Weather Database.

weather.com parser

The Weather.com Parser, when called upon, parses the weather.com website, collecting data about local weather.

Weather Database

The Weather Database stores information about weather for about the next two weeks

Events Parser Manager

The Events Parser Manager is in control of deciding when to gather new information about events. When the time comes, the manager uses one or more parsers to collect events data and update the Events Database.

Projo.com parser

The Projo.com parser, when called upon, parses the projo.com website, collecting data about local events.

TurnTo10.com parser

The TurnTo10.com parser, when called upon, parses the turno10.com website, collecting data about local events.

Events Database

The Events Database stores the most important information, the actual events. Locations should already be resolved and latitude/longitude coordinates already computed and stored in the database.



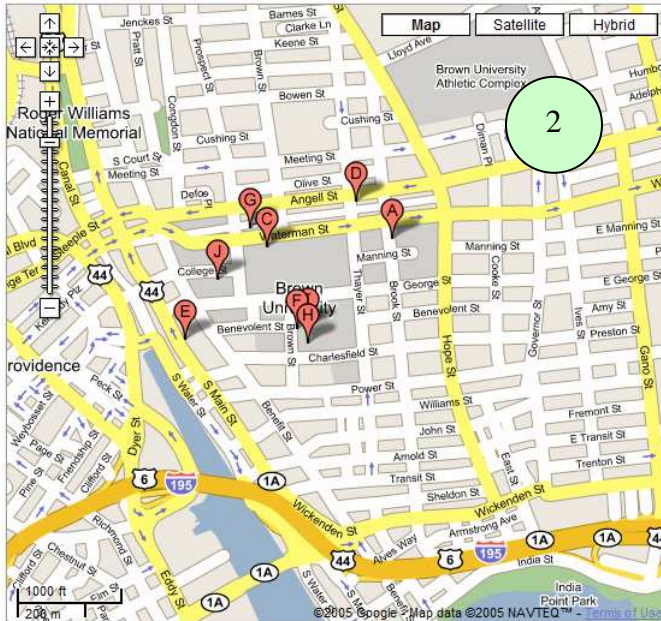
Genie

The Location Resolution Genie is the component that takes a “location” and returns both an address and a pair of latitude/longitude coordinates corresponding to that address. The first version of this should be trivial. Any location that is passed in that isn’t just an address that can be geocoded should just return null which will cause the Events Parser Manager to drop the event instead of add it to the database. Once this simple functionality is in place, if there are sufficient resources to do something cleverer, then a new strategy may be devised to replace this first one.



5. GUI Diagram

GeoEvents¹



Address:

City:

State:

Zip Code:

3

☐ Theatre

☐ Music

☐ Arts & Museums

☐ Etc...

4

Date:

6



Event #1

Time and Place
Cost



Event #2

Time and Place
Cost



Event #3

Time and Place
Cost

5

The GUI is organized so that the most noticeable item is the map itself. Easily accessible on the right are the options for changing and filtering the events. They are in a standard clean format so that they do not distract from the map. Below this main part of the GUI, the list of events is displayed.

The items in the map, corresponding to the green number tag, are:

1. **Title:** The page title.
2. **Map:** This is the map that will display the event locations with markers. Users can pan around the map and zoom in or out. Clicking on a tag will provide the user with more complete information about the event including time, location, cost, and distance (see details in the features section above)
3. **Address Frame:** This area is where the user inputs the address to center around and base directions from. After entering the address data, the user can click “set address” set this option.
4. **Filter Frame:** This box contains all the filter options for event types.
5. **Date Frame:** Allows the user to select a date to look at events for.
6. **Results Frame:** All events and their corresponding details are displayed here.

6. Usage Requirements

- a. Since this product would be of public interest it would need to be reliable under a heavy usage (500).
- b. Any longer of a response time than 5-10 seconds would be inadequate since multiple searches or refinements of searches (different types of events, the zoom factor on the map) could be made in short succession.
- c. Depending on connection speed, loading should take less than 30 seconds or a minute
- d. GeoEvents must be consistent.
 - i. If a user searches around a base address and certain type filters and finds an event they like, if they come back an hour later to get directions, they should easily be able to find it with the same search parameters.
 - ii. Essentially, searches with the same parameters should return the same results.

7. Non-Functional Requirements

- a. Testing
 - i. Besides testing to ensure that the project responds in the specified amount of time, it is essential that it continues to do so as the user usage increases.
 - ii. Data displayed must match events included in the database
 1. This should be tested under a high load for reliability
 2. Compare located events to actual HTML sources; ensure that addresses don't get confused and that all events claimed by parser really exist.



3. Compare located events to actual HTML sources; ensure that addresses don't get confused and that all events claimed by parser really exist.
 4. Compare located events to actual HTML sources; ensure that addresses don't get confused and that all events claimed by parser really exist.
- b. GeoEvents must be accessible from any computer through the web
 - c. Must be written and ready to demo by the end of April 2006
 - d. Must have thorough documentation on the use of the program and possible extensions for the future that were not implemented in this version
 - e. Ease of Use
 - i. This project should be usable for anyone capable of surfing the internet.

8. Divisibility

- a. This project should end up being sufficiently divisible for the projected size group for this year's class. Different elements include
 - i. Overall GUI
 - ii. Google Maps API interaction
 - iii. Parsing of the Providence Journal's Events Page information and other events listings
 - iv. Filtering and searching of the event information once parsed

9. Specific Challenges/Issues for the Project

- a. External Dependencies
 - i. It is unclear at the moment how well organized the Providence Journal's Events information is (it is not expected to be highly so). The largest difficulty could lie in deciphering this language so that it is in a useful enough of a format to use.
 - ii. **Neither the Google Maps API nor the Yahoo Maps API interfaces with their directions creating capability.**
 - iii. Google forces the use of an external geocoder (switches an address to longitude/latitude that can be used in the API). Yahoo has an internal geocoder.

