<u>Seattle Area 911 Call Map Visualization</u>

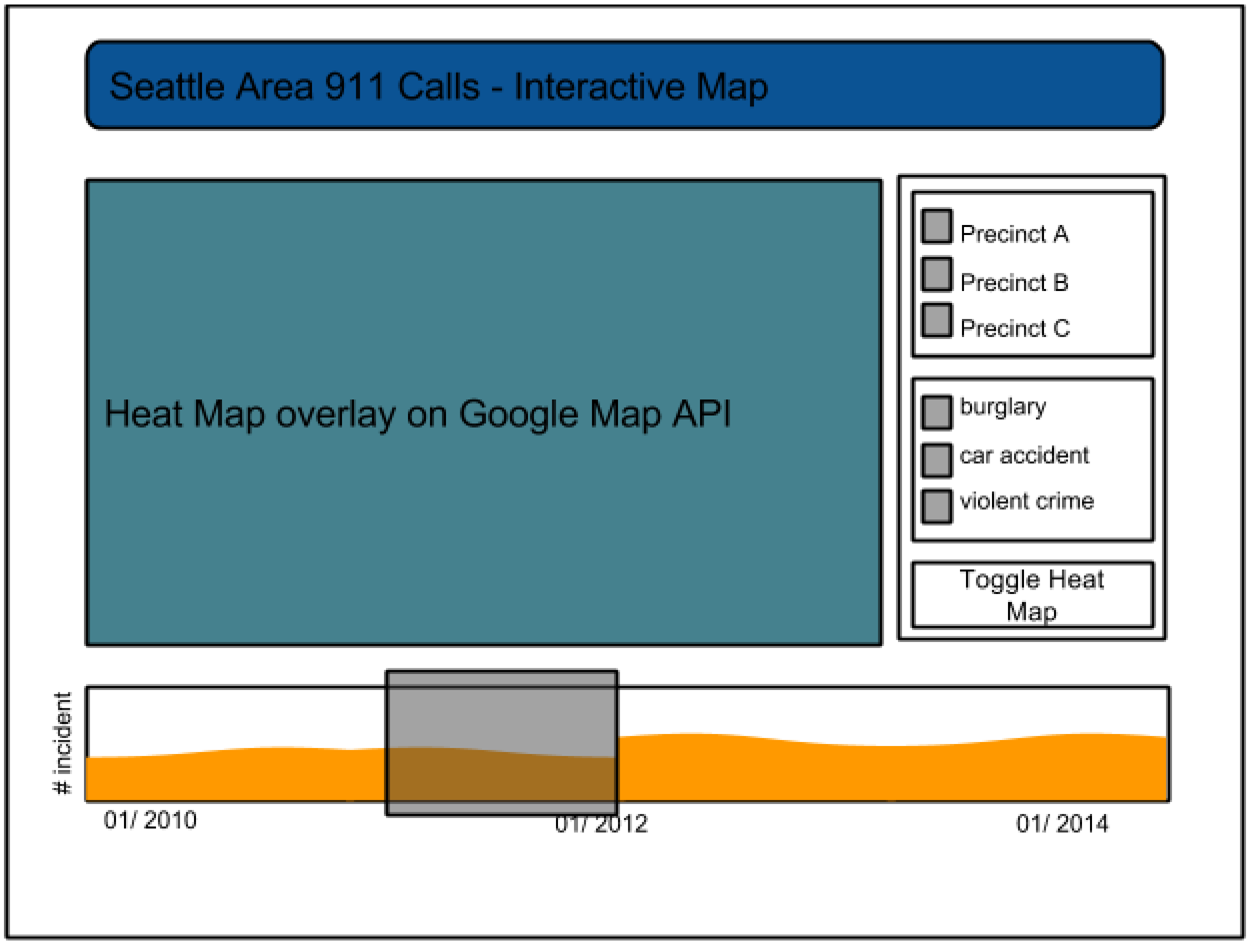
<u>Dataset:</u>

We provide analysis for the Seattle Police Department dataset on incident response. You can find an up-to-date dataset here, though our dataset is slightly smaller (May ‘13-Feb ‘14)
https://data.seattle.gov/Public-Safety/Seattle-Police-Department-911-Incident-Response/3k2p-39jp

The dataset includes a (partial) dataset for each reported incident, including various incident *identification numbers*, *categories for reported incident*, (e.g. home burglary alarm) and the *actual categorization* after the incident (e.g. false alarm). In addition, you get location by *precinct* and *beat*, *latitude*, *longitude* and *address*, as well as dates and times for *clearance* and *on-scene* information. This last data point is not present for all incidents.

<u>Visualization Techniques:</u>

The Seattle police department does have a current visualization for the dataset provided. However, its interface is less than desirable. We want to improve the interface and provide a more attractive and useable visualization for the data. We envision an interface that we roughed out looks something like this:



<u>Features:</u>

1. **Heat map of number of incidents for some time interval** (overlay on Google Map)
   A Map overlay is the most intuitive for data like this, with geographical information and importance. For instance, a quick sampling of this technique in Tableau showed that there were concentrations of 911 activity in certain areas of downtown, Seattle Center, and the Udistrict which are both informative and well-represented on a map encoding.
2. **Exploration of time intervals via brushing on the time graph** (lower section)
   One of the most frustrating aspects of the existing online map interface is the inability to filter by time or see how the incidents change over time. That is, to answer the question "has crime increased in my neighborhood over the last year?". We added a time slider to answer this exact question. Brushing selects a time window to show on the heat map.
3. **Filter options for precinct/beat and incident type** (right sidebar)
   This option allows you to view whether some precincts are hit harder than others, or whether they might be slower to respond. It also provides a nice geographical filter, much like filtering by state or county (for national data) does.
4. **Option to switch heat map to response time** (clearance date - onsite time)
   One of the more interesting values in the dataset (unrepresented in their online map) was the implicit value of "response time": the amount of time between an on-the-scene response and the incident being cleared from the books. We want to also provide a heat map that reflects this, with long response times being reflected in red and quick response times in cooler colors. This would provide a visual of where (on the map) responses can be expected quickly.

<u>Implementation:</u>

In the end, development proved a little too tricky for us to complete our full prototype as storyboarded above. We got the heat map working with the time slider, as well as a few more options for the heat map, but were unable to complete the full prototype we had envisioned. Our heat map and time slider are very similar to the storyboard. However, we ended up allowing people to change the scale on the slider, both for more detail on the map and the graph of total incidents. The filters were bypassed due to lack of time.

We both met to discuss the preliminary design and the dataset mockup, as well as to explore the data in Tableau. Following that we (independently) created mockups, both working with the same idea. We explored both options for adjusting the time exploration and decided to go with one of the prototypes, which we then improved upon together.

Getting the brushing slider to work correctly was by far the longest process. Google maps and the heat map were relatively straightforward, but getting time selection and translating that into pointers in array (and in particular, doing it smoothly and efficiently) were much more difficult. A general lack of familiarity with the languages involved was also a major hindrance to our implementation: the flow of data was not always clear, and how to do error correction was also difficult.