CODENAME: CHILL OUT

Components and Interfaces

Adam Gray

December 8, 2015

Objective

The primary objective of this mobile application, Codename: Chill Out, is to help users reduce and or eliminate factors that lead to stressful situations in their day-to-day lives. In order to accomplish this central task, the application must achieve four sub-goals, namely detecting a user's stress levels, recognizing the user's contextual environment, determining which factors of the events in the user's calendar are most likely to lead the user to be stressed, and providing suggestions to reduce or remove those factors from the situation. The user will be able to provide feedback on the quality of the suggestion, which the application will use to improved future suggestions.

Functional Components

- Noise level monitor — Using the microphone in the user's mobile device, the system tracks a change in noise level over the course of an event and between events. The noise level is relative, which means the monitor is not measuring an absolute value of noise but rather determining whether or not there has been a noticeable increase or decrease in the noise level of the user's surrounding environment. The following algorithm is performed on only one reading at a time, and readings are never revisited. This makes the algorithm quite efficient on both clock time and memory. Uses EJML API.

```
weights = double[5];
previousReadings = int[5];
maxAvg = int[5];
minAvg = int[5];

for each noise level reading r
       r = KalmanFilter.predict(r);

       if r > max(previousReadings)
              for i from 1 to 5
                     maxAvg[i] = ((1 − weights[i]) ∗ maxAvg[i]) + (weights[i] ∗ r)
              end for
       else if r < max(previousReadings)
              for i from 1 to 5
                     minAvg[i] = ((1 − weights[i]) ∗ minAvg[i]) + (weights[i] ∗ r)
              end for
       end if

       previousReadings[positionOfOldestReading] = r;

       for i from 1 to 4
              if(maxAvg[i] > maxAvg[i+1] && minAvg[i] > minAvg[i+1])
                     loudNoiseChangeDetected(i);
              else if(maxAvg[i] < maxAvg[i+1] && minAvg[i] < minAvg[i+1])
                     quietNoiseChangeDetected(i);
              end if
       end for
end for
```

- Physical activity level monitor — Tracks relative physical activity and overall movement of the device both during and between events. The physical activity level is more of an absolute value than the noise level. Uses EJML API.

```
//use multidimensional implementation of noise detection algorithm
//takes steps and accelerometer movement readings as parameters
```

- Stress monitor — Using biometric sensors in the Microsoft Band, e.g. heart rate sensor and GSR sensor, the monitor determines whether or not the biometric readings are elevated. Factoring in physical activity, which could also be the cause of elevated heart rate and perspiration levels, the monitor determines if the user is stressed or not stressed. Uses EJML API and Microsoft Band API.

```
//use multidimensional implementation of noise detection algorithm
//takes heart rate, GSR, and hear rate variability as parameters
```

- Contextual environment builder —Taking in the event data from the user's calendar and combining it with the user's stress level, physical activity level, and the surrounding noise level, this component builds a model of the user's surrounding environment and contextual situation.
- User data model — The model contains information specific to a given user. It stores past events, different particularly stressful situations and factors that may have caused them, and suggestion feedback in addition to all of the users personal information.

```
     GROUPS   : made up of PERSONS
     PERSON   : name, belongs to GROUP
     EVENT    : title, location, time, list of PERSONS as attendees
     NOISE    : current level, average within past hour, 5 hours, 2 days, month
     PHYSICAL ACTIVITY :  avg per hour, avg per day, avg per week, avg per month
     STRESS   : current level, avg per day, avg per week, avg per month
```

- Stress/Event correlation detector — Given a user model, this component which events and factors are particularly stressful and updates the user model accordingly

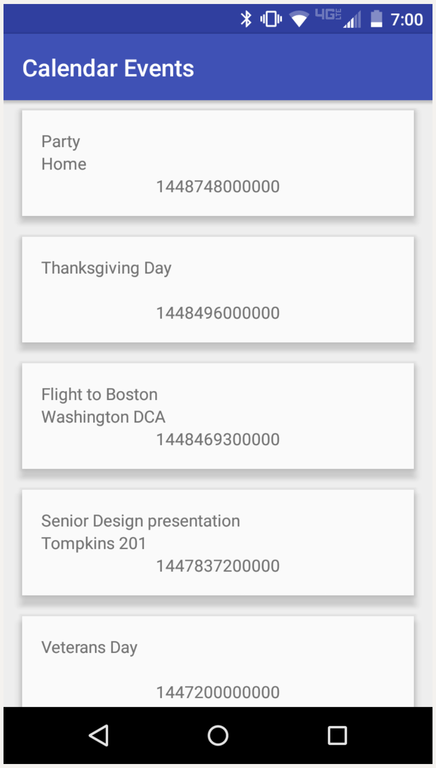
```
//custom neural net / deep learning algorithm
//yet to be designed and primary objective for next semester

//will take all components of user model as input except for stress level,
//which is used as the binary classification of arguments to train the network.
```

- User Interface — Allows the user to interact with data contained in his or her user model, primarily to view past event and stress history

```
//user is presented with list of event activity and can tap a list item for
//additional event details will be able to indicate if an event was stressful or not

//another aspect to be improved next semester but here are screenshots in their current
//implementation
```



Calendar Events

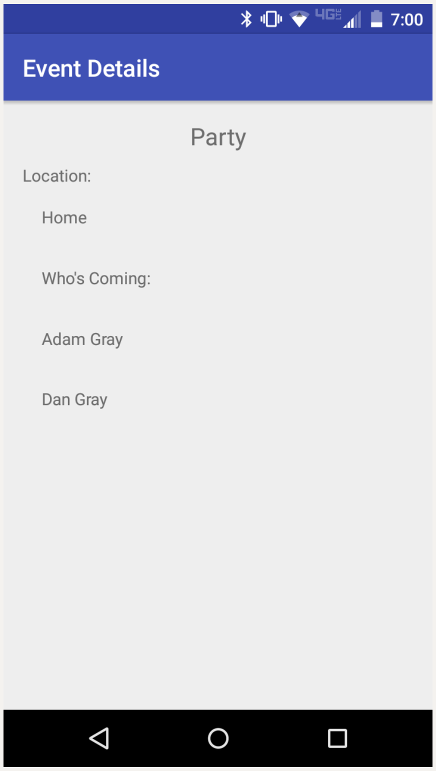
Party
Home
1448748000000

Thanksgiving Day
1448496000000

Flight to Boston
Washington DCA
1448469300000

Senior Design presentation
Tompkins 201
1447837200000

Veterans Day
1447200000000



Event Details

Party

Location:

Home

Who's Coming:

Adam Gray

Dan Gray

- Stress reduction suggestion system — Given a user model, which contains the users stress factors and past suggestion feedback, and a future stressful event, this component makes a suggestion personalized for the user to help reduce his or her stress level during the given event. Will possibly use AYLIEN API.

```
//custom neural net / deep learning algorithm
//yet to be designed and primary objective for next semester

//goes hand in hand with stress/event correlation detection
//will potentially require parsing event titles and natural language processing to
//provide more accurate and consumer friendly suggestions
```