Rayner Kristanto - 903031966
CS4590

Device Context

This is how I imagined the user would use the system. The user would be wearing a wearable device around the ear and that device would use the sound system that I created. The purpose of the device is to give the user the information needed so the user could decide whether or not to check his or her phone when the phone receives an event. A secondary purpose of the device is to provide the user the information he or she would get by checking his or her phone so the user doesn't have to get his or her phone out. For example, instead of the user having to check the phone to see someone texted the user "ok", the user could be notified through the system instead.

Sound Choices

Each event has it's own sound. I chose the tweet sound because the sound reminds me of a bird while also not sounding how a real bird would sound. This would allow the user to quickly connect it with twitter notifications while not confusing the user by sounding too realistic. I chose the text sound because it is the default iPhone text sound so many people have associated the sound with a text message. I chose a doorbell sound as the email sound because I thought it would be easy to associate the sound with a packaging arriving which is associated to mail. I tried to make the sound less realistic by editing the sound so it would not confuse the user. I synthesized a phone ringing sound for the missed call sound because people can easily associate that sound with a missed call. This sound was created by alternating the frequencies of C_5 and D_5 quickly. I also synthesized the voice mail sound to make it similar to the missed call sound. I played the frequency of C_5 two quick times so the user would be sure that the sound played. I chose my heartbeat sound because it was a repetitive sound that had a duration of low frequency noises followed by a duration of high frequency noises. This way, even if the gain of the sound was low, the user could hear the sound due to the constant change in frequency.

Each of the original event sounds I chose to play have high frequencies. The original heartbeat sound also had high frequencies but a lot of those were filtered out using a lowpass filter. This is because humans are good at hearing high frequency sounds and bad at hearing low frequency ones. I wanted the users to detect the event sounds with no effort, but in order to hear the heartbeat noise, I wanted the users to have to try to hear it.

Sound Design based on Context

This is how I interpreted the priorities once I read over all the sample event JSON data:

1. The user may get happiness by knowing the event, but the event can be disregarded
2. The user may want to know about the event and disregarding the event may affect the user negatively
3. The user will want to know about the event but the event does not require immediate action
4. The user will want to know about the event and the event may require immediate action

This interpretation contributed to how I decided when sounds should and should not be played.

While presenting, one reason I would want to know if my phone receives an event is if that event may lead to immediate action. For example, if the user's mother was hit by a car, the user may want to immediately go see her. To me, the only event that would make me stop my presentation is a phone call. What makes a phone call unique from the other events is that the sender wants me to respond immediately. Thus, if the event is not a missed phone call, it can wait. When the user receives a missed call with a priority over 4, the missed call sound is played. I use TTS to inform the user the missed call's sender's information in order to help the user decide if the phone call was urgent enough to stop the presentation.

Another reason I would want to know if my phone receives an event is if that event was important to me. Thus, for all other events with a priority over 4, the respective event sound is played. Even though I will continue to present, these sounds will let me know the amount of important information I received and what applications to check and use to obtain that information. I imagine that because this system is able to assign priorities to the incoming notifications, the phone applications would be able to sort notifications based on priority. This would allow the user to quickly see the important notifications that were missed by just knowing where to check (ex: Twitter, Phone, Mail).

While hanging out with friends, it's important to show that you value your friends' time. However, your friends would understand you quickly responding to things. Thus, I decided to not notify the user if priority 1 events came. For all other events, I play the respective event sound so the user knows someone is trying to reach the user and decide how to act based on how the conversation is going. For example, if I hear that I received a text message and my two friends I am hanging out with are talking to each other, I would decide to check my phone to read the text message and do something based on that event's information. If I hear that I received a text message and a friend is talking to me, I will know that I got a text message which I can check later once my friend is no longer talking to me. For priority 4 events, I used TTS to inform the user of the sender of each event. One reason for this was to let the user know that the event had a high priority. Another reason is that by informing the user of the sender, the user may be able to predict the contents of the event. I do not think that the TTS would interfere with a conversation with friends because priority 4 events should be rare and performing TTS on the sender's name, tag, or number should take at most 4 seconds (this number is from timing how long it took to perform TTS on a 9 digit phone number) .

While walking, I had to balance the sound so that it provided as much information to the user as possible without distracting the user from getting to his or her destination. I played an event's respective sound no matter the priority because I hypothesize that the information received would not distract the user so much that it would endanger or inconvenience the user. However, to be sure, I would ideally test this with people so I could have data to support my hypothesis. In this context, I added a layer of complexity by adjusting the rate of the events sound based on

the content summary if those events had a context summary field in the JSON. The sound would play faster if the content was positive, the normal rate if the content was neutral, and slower if the content was negative. This is because faster sounds are associated to be positive while slower sounds are associated to be negative. This layer of complexity was added because I believe the user would have enough cognitive resources to devote to associate the sound with the correct event, even if the sound's frequency was changed. I also believe the user would be able to tell whether the sound had a higher frequency, the same frequency, or a lower frequency to determine the content summary. I do not think that users would be able to do this during a presentation or conversation because in those contexts, the user has to focus on what the user is saying or what other people are saying.

While working out, I had to focus on how to convey as much useful information as possible without making the sound output cluttered. I also had to consider that most users are forcing themselves to workout and that they are not working out because they enjoy to. Thus, I do not want to give them an excuse to stop working out. This reason steered me away from adding TTS to inform the user of the sender of priority 3 events. This should not hurt the user because priority 3 events do not require immediate action meaning the user can attend to them once the user has finished working out. However, most people do have their phones near them when they are working out. Thus, once they hear they have a notification, they will most likely check their phones and see the notification anyways. Even if they do not hear the notification, some users will still check their phones, usually to check the time or change their music. Because of this, I decided to use TTS to inform the user of the sender of priority 3 events. I also conveyed more information per tweet by using TTS to inform the user of the number of favorites and retweets a tweet received. In addition to that, the event sounds have the same complexity as the walking context because the user has more cognitive resources to devote to recognizing and differentiating between the sounds.

Ideally, this sound system would have a sound queue which would prevent the sounds from overlapping. However, due to time constraints this was not implemented. I do think this is a minor thing because most users will not be bombarded with several different notifications at the same time. However, this is something that should be added to benefit the user.

Review Sound Functionality

The review sound functionality was designed to store the exact same events that the user heard before and play the respective event sounds as well as the timestamp of when those events played. Ideally, the JSON object would store the absolute time (like 9:42) instead of a time relative to when the file was loaded so the system could tell the user numbers that the user would understand.