Kotodama Fruit Juice

Speak With Purpose Project Extended Post-Mortem (Development Story)

Project website: http://www.etc.cmu.edu/projects/kotodama-fruit-juice/

Authors: Heather Martin, Dilara Semerci, Arseniy Klishin, Chenxin Momo Jiao, Allen Tingley

Project Overview

Description:

“Speak with Purpose” is an applied research project created by team Kotodama Fruit Juice in association with the Entertainment Technology Center Professors Brenda Bakkar-Harger and Shirley Yee. Tasked with exploring “stories that listen”, the five technologists that make up Kotodama Fruit Juice are using advanced speech recognition and narrative trickery to discover what possibilities exist for the future of entertainment and natural modes of interaction.

The ability to establish meaningful relationships and exchanges through one’s speech is a core tenant of the communicative experience, and current attempts to mimic the process synthetically are limited to practical, industrial spaces instead of our living rooms.

Kotodama Fruit Juice is thinking outside the (juice) box to provide realistic, meaningful interaction and story, and all you’ll have to do is speak with purpose.

Goals:

- Development of an emotive interactive experience, where player’s speech serves as the main interaction and is meaningful to the gameplay.
- Research of current technology possibilities and limitations in the field of speech recognition.

Deliverables:

- A speech driven Unity application that uses Microsoft's Speech Recognition API to determine the semantic value of voice input
- Grammar document with acceptable words and phrases
- Post-mortem/research paper for future developers

Clients:

Speak With Purpose is an ETC Faculty Pitch project.
Advisors for team Kotodama Fruit Juice are **Brenda Harger** and **Shirley Yee**.

Team:

Chenxin Momo Jiao - 3d Artist, Animator

Arseniy Klishin - Producer, Sound Designer

Heather Martin - Programmer

Allen Tingley - Writer, Co-Producer

Dilara Semerci - Designer, Programmer

What went well

- Kotodama Fruit Juice is a qualified team with rather balanced skills and a lot of passion towards the project, which allowed us to work hard towards our final goal and successfully avoid strong technology limitations with different design decisions.

- We started from a completely open field which was a very interesting creative challenge - produce a game having almost no limitations from the design point of view. The team had to spend a lot of time on iterating on design decisions, and after trying to apply these decisions to current technology, sometimes it turned out that the idea had to be strongly improved, if not completely utilized. Nevertheless, the team was always open to changing the direction and solved these problems with the best possible attitude.

- Being a part of Carnegie Mellon University, very well-known for its Computer Science School, we had a lot of specialists from the field of speech recognition very close to us, so we could get the opinion of experts from the first hands. We got a lot of help from the field professionals such as Mike Christel, Scott Stevens, Alex

Rudnicky, Arnold Blinn. It’s hard to overestimate the amount of help and useful feedback we received from them.

- Not only the technology specialists were accessible for us, we also had a chance to consult with a lot of people from the design point of view. We received a great amount of useful ideas, a lot of which we implemented in the final version of the product from Jesse Schell, Anthony Daniels, … and all the other people. Considering the setting we chose (a stand-up comedy club), it was extremely important for us to be in touch with people who perform and do improv, and luckily, we found a great contact for Woody Drennan who gave us a lot of help with playtesting a lot of great tips from the content point of view.
- Though each of us had different expectations from the project, we all agreed to create something that will make people have the “Wooooow!” moment. We overcame the initial communication problems, put our own desires aside and work on something that everyone feels enthusiastic about.

What could have been better

- Although the project being completely open is a very interesting creative challenge, it had it down sides. It turned out to be a very difficult task for us as a team of people with completely different interests and tastes to come up with ideas that both satisfy our passions and fit the limitations set to us by current technology. If we had some boundaries and didn’t start in a complete blue sky we would have come up with a valid direction and start the development process earlier.
- Our clients were our advisors at the same time. Team was not sure if they should be treated like our clients and share the information that way, or treated like our advisors and share the process differently.

Development Process

Choosing Technology

There are many speech recognition technologies available which initially made the task of choosing one daunting. Instead of tackling this problem alone, we decided to enlist the help of experts who had already worked in the field of speech recognition. Among the people we consulted were Alex Rudnicky (the director of CMU Speech Consortium) and Arnold Blinn, who had previously worked with Microsoft on developing the Kinect with speech recognition. We also talked to ETC Faculty who had experience developing speech recognition applications including Mike Christel, Scott Stevens and Bryan Maher. What we learned was that although there are many speech recognition technologies out there, there really is no “best” one. We were encouraged to not worry about which one was the best (they were all more or less good) but instead to pick one we were comfortable working with. The point being to start developing quickly to research what does and what doesn’t work when creating a speech recognition applications. We decided to use Microsoft’s Speech Recognition API for the PC because of its extensive documentation and because we were able to get it working with Unity quickly. However, there were a few other technologies we looked into that are worth noting.

One of the more promising devices we looked at was the Kinect for the Xbox One. The new Kinect (among other useful features) offers the ability to recognize voice input using Microsoft’s Speech Recognition Engine along with the ability to track facial expressions. These expressions include but are not limited to whether or not the subject is engaged, if they are looking away, talking, winking, or if they’re wearing glasses. From a game development perspective, the ability to combine facial expressions with voice input looked promising. However, since the Xbox One had not been released prior to our development cycle, we were not able to get out hands on a development kit.

Google has also delved into the realm of speech recognition. They have an API available for their android developers and are also working on a tool that allows speech to text in web browsers. However it was difficult to find documentation for their web tool and since the team saw no advantage in developing for mobile, we didn’t choose to use their Android speech API.

Also worth noting is Apple’s Siri. This is a widely known speech recognition application however there is no development kit available for it. Worth mentioning is that the success of Siri is primarily based on the data Apple has collected from people who have used Siri. Apple has been able to data mine useful information from people who use Siri which, in turn, makes Siri smarter. Unfortunately, this data is not made available to the public. On the other hand, there are speech recognition kits available to develop on the iOS. As previously stated though, we saw no advantage in developing for mobile since we were approaching the project from a research oriented perspective instead of a commercial one.

First prototype (Symphony Conductor)

Our first prototype was a very basic symphony conductor simulator. There were three cubes on ‘stage’ and each cube represented a different instrument in the symphony (voice, piano, and xylophone). The player was able to control each instrument through separate voice commands. These commands included the following: telling an instrument to start playing, stop playing, become quieter, and become louder. The purpose of this prototype was mainly to see if we could get Microsoft's Speech Recognition engine to work with Unity. The reason this was an issue was because at this point in time, Unity only supports .NET 2.0 and Microsoft's SRE (Speech Recognition Engine) runs on .NET 4.0. We were able to solve this problem by creating a background Visual Studio application that handled the speech processing side and wrote that data to a file that the unity application was able to read.

Choosing the character

Before we started production phase of our project we spent a lot of time on trying to find the design that suits our initial task (create an emotive experience with speech recognition as main gameplay element) and narrative that can support it. It was a very hard question on what to start with - creating the design and building the narrative around it, or in the opposite way. Finally we came to the conclusion that it might be easier for us to iterate on design after we get inspiration from the character.
During our research phase, one of the conclusions we came to was that more successful projects with speech recognition at the moment, tend to have the characters that you don't expect to understand everything you say (for instance, dogs from Nintendogs, Pikachu from Hey you, Pikachu).
Based on this fact and our collaborative taste we chose a humanoid alien-like character, who we soon called Fred.

Time Management

But before Fred became Fred, we iterated on other design decisions. We started from Time Management genre where minions looking like our character assemble things on the factory and followed player's speech commands.
Soon we realized that there are 2 major problems with that coming from the fact that gameplay was depending on time a lot:
1) Misunderstandings with speech commands will happen occasionally (probably often) and this is very frustrating when you can't lose time to win the game
2) Speech recognition has a delay, which for the same reason as the first point will cause frustration

Once Upon a Time interruption mechanics

Another idea for design we had was based on an improv game Once Upon a Time where a person tells the story and at any point viewers can interrupt the viewer saying "No, you didn't!" and the storyteller has to change what he is saying on the fly.

In our case we could write different branches for the story and allow players to interrupt whenever they want.
This design didn't have the problems of the Time Management one, but was a little too simple.

Choosing of Once Upon a Time interruption mechanics

After we received feedback from the faculty, and heard all their useful advices about the direction we should choose, we decided to go with Once Upon a Time. We started our production phase right after the choice was made, created the interruption mechanics rather quickly and were able to build up on it and think which features will create the feeling of excitement in players.

Trying to use free speech processing

One challenge we tried to tackle was free speech processing. Microsoft's API offered a speech to text function which enabled users to say anything into a recording device and the engine would do its best to match what it heard to legible text. One of the problems with this was when the user intentionally said something that was not speech (random sounds or gibberish) because the system would still try to match it to something intelligible. Although the results were often comical, it wasn't what we were looking for. Since we were in the game development space, we were always looking for specific phrases that our game could recognize and then act upon. However with this method of processing (as opposed to using a grammar that would only look for specific phrases) the amount of possible things the user could say was not limited to a finite set of words but instead only limited to what was in the English grammar as a whole. Because of this, the accuracy of the engine matching what they said to text dropped significantly.

Building up on interruptions: Adding Questions

During our playtests, we discovered that having heckling and praising as the only interactions is not enough for the experience. Most of the playtesters didn't feel like they want to heckle him as they didn't want to be mean or it's not something that they would expect in a comedy club. For praising, we discovered that people just clap their hands or simply laugh and expecting them to say "Good job" during a performance is not very realistic. We needed to have another thing that will encourage players to interact with Fred. After discussing a bunch of ideas, we decided to include questions which seem open but still have a limited amount of answers. We discovered that the questions increased the interactions a lot. Therefore, we decided to make the questions our focus on interaction while keeping the heckling and praising as our secondary interactions.

Script Iteration and Woody Drennan’s help

The script on in our project has changed many times. Only for the final design iteration, we had to rewrite the script 6 times in order to achieve the closest approximation of what we wanted in the end. We chose comedy as our primary genre, and it turned out to be an extremely difficult setting. Every culture has its very own understanding of what is funny and what is not, what is acceptable as a joke and what is insulting. There is a very small amount of topics that are funny universally, so it took a while to understand what kind of subjects we want to include in our comedian's monologue. Of course, even after figuring these questions out, every person even in the same culture has a unique sense of humor, so it was extremely hard to satisfy everybody.
Famous performers who everybody gives references to as examples of good comedy are professionals, and it takes them years to learn how to write funny things and be funny, and for a project that is 15 weeks length, this might not have been the best genre for practical reasons. But still, it was an extremely interesting field to explore.

Our comedian started with an extremely pathetic personality. His jokes were basically all about self-pity and were rather long, which didn't leave too much space for interaction. We tried different confidence levels through the semester for Fred and by the sixth iteration we found the sweet spot, somewhere between confident and overconfident.

Our character had different mood conditions and though his questions and answers to the player remain the same in each play through, the bits of the monologue between them are different depending on which mood Fred is currently in. In this way we had to have to write and record a very large amount of text, and to record it as well, since this was the only way to playtest how people are taking the new personality of our hero.

Our advisor Brenda Harger is an improv professor, so with her help we met people from Pittsburgh Improv community who helped us a lot. Woody Drennan from Steel City Improv Theatre gave us a lot of inspiration and a lot of valuable feedback that helped us to come to our final draft of the script. With his help we understood much better what kind of vocal interactions people are comfortable with and what do they expect from the performer to be entertained.

Difficulties with Technology

One of the unexpected difficulties we ran into during development was constructing the grammar for our application. The grammar was an xml document that specified which words and phrases the speech recognition engine should look for. If the SRE heard speech it would cross check the speech with the grammar and if it was able to match the speech with something in the grammar it would send back the word that was recognized via a speech recognized event. This data was then sent by our background application to our Unity game by writing to a file.

One problem we ran into was when two words in our grammar sounded similar. The SRE would often confuse these words together and the accuracy of our application’s recognition of these words would drop. An example of this problem would be the words “beta” and “delta”. Our solution was to replace these words with different ones to eliminate the confusion.

A similar problem we ran into was when speech got recognized that should not have gotten recognized. This happened when a user that said something that was not in the grammar but that sounded like a word in the grammar. Since the SRE was only checking to find the best possible match within the grammar, if it found something that sounded enough like what the user said it would register a hit. An example of this problem would be “um” and “mom”. “Mom” was a possible answer in our grammar for the question “Where are you from?”. However, through testing we realized that some of our users used the word “um” as a filler word before answering a question. So when they were answering a question and they said “um” (even if it wasn’t the “where are you from?” question) the SRE would send back the hit on “mom” which usually resulted in our Unity application defaulting to the “speech not recognized” case. Like the previous problem, this was solved by user testing to define and eliminate potential conflicts.

Playtesting and Its Importance

Playtests played a huge role in our implementation phase. During the first half of the semester we conducted 3 playtests to decide on which gameplay works better, which interactions can be achieved in a comedy club environment.
After we settled down on our idea, we had weekly playtests with certain goals. We iterated on our interaction design, environment, script and animations according to the feedback we got from our playtesters. We updated our grammar according to the playtest results. We added different answer alternatives as well as removed confusing ones.

Throughout the semester, we conducted 7 playtests with 58 playtesters from 12 different countries. It was exciting and fascinating to observe that Microsoft Speech Recognition Engine worked for all these 12 accents.

- Playtest #1 - Week 5: Once Upon a Time vs. Time Management
  We tested 2 of early ideas: Once Upon a Time and Time Management. The goal of the playtest was to decide which gameplay works better. We decided to move forward with Once Upon a Time idea.
- Playtest #2 - Week 7: Allen Performing
  After deciding on Once Upon a Time idea, we settled down on the setting “Comedy Club”. During this playtest, Allen performed a monologue of a nervous newbie comedian. We tested the audience’s reactions. Most of our playtesters felt uncomfortable, they felt pity to the comedian and did not want to interact.
- Playtest #3 - Week 7: Woody Performing
  We have asked an improv actor, Woody Drennan, to perform an interactive show. The main purpose of this playtest was getting facial and body animation reference for our artist, Momo. We had 2 sessions: during the first session, Woody was physically present in the same room with the audience whereas during the second session, we established a video conference with him where the audience members were seeing him and interacting with him through a TV screen. We wanted to see how the audience behavior differ in these 2 scenarios considering the fact that our virtual comedian will be behind the screen all the time.
- Playtest #4 - Week 10: Questions
  After our halves presentation, we decided that the heckling & praising is not enough, we need to include some questions to increase the interaction. We tested 3 types of questions: Yes/No, Either/Or and Open Ended. The theme of the questions was “Family” and we discovered that there are lots of questions that people feel uncomfortable answering about their families. We discovered that Yes and No has lots of variations such as Sure, Yeah, Nope, etc., either/or questions worked relatively better and open questions requiring calculations (such as calculating ages) caused lots of trouble. After this playtest, we decided to have questions which look open ended but which we can limit the number of answers.
- Playtest #5 - Week 11: Interruption and Questions version 1
  During this playtest, we tested our first version including both the heckling/praising interaction along with the answering questions interaction. In this version, we had a black screen when the introduction were playing, which

made people not listen to the instructions carefully, there were other audience members but they were not moving which made them feel awkward, the heckling and praising didn't work very well, the questions definitely increased the interactions, the pauses for waiting for answers felt too long, headset reference and the nickname helped players to understand that the character is talking to them. Some playtesters wanted to hear more funny jokes, some jokes that was not at his own or the player's own expense. The animation flow was not smooth. Using Oculus Rift definitely increased the engagement to the environment. After this playtest, we decided to add "Who" and "What" words to the grammar with special treatments. We decided to get the animations, both for the character and the other audience members done for the next playtest and make the environment more attractive.

- Playtest #6 - Week 12: Prettier Environment and Funnier Script
  For this playtest, we beautified the environment, added idle animation for the virtual audience members, added a back and forth conversation between an audience member and Fred to serve as a cue for the player. We shortened the delay for waiting for the response. We added fancy spotlight to the beginning and also spotlight movement for the asking questions. These helped us a lot to engage the players to the experience. We added shout-outs from the audience to encourage the player to heckle or praise Fred during the experience. Playtesters loved the environment and the light served very well for answering the questions. Our script was better than the previous one but still needed some tweaking. We enlarged our grammar according to the feedback results. We decided to add more variety to the animations and make the transition smoother.

- Playtest #7 - Week 13: More Animations and Newer Script
  During this playtest, we used the stand alone microphone and the speakers. We wanted to test whether the mic would get the game output as input or not. We discovered that other than a slight decrease in the accuracy, the system worked fine. It was a good sign that we could present our experience like this during our Soft Opening presentation. Our script is found more entertaining this time and one of the playtesters who has experienced last 3 scripts told us that he has witnessed the character's journey from being a real loser to a total jerk. After this playtest, we decided that the end section is too long, and we need to shrink that part a little bit. We added more words to our grammar. We also realized that some people kept trying to heckle him even when they have been asked questions, this made us change our introduction audio file.

For playtester data, see Appendix B.

Final Design

Background Story

Fred is a stand up comedian from another universe. Tonight, he will be performing in the interactive comedy club called Wormhole Comedy Club.

Characters

Fred:

He is a nervous comedian performing on the stage. He is responding to the verbal reactions and answers via his facial expressions, skin color, gestures and posture.

Player:

Player, our guest, is playing as one of the audience members in the comedy club. (S)he can heckle or praise Fred during his performance and (s)he is being asked questions by Fred.

Audience Members:

They are non interactable, non-player characters. They are heckling and praising Fred during his performance. One audience member has direct interaction with Fred in the beginning of the experience to serve as cue for the player. They are backing up the player whenever (s)he heckles/praised Fred.

Bartender:

He is a non interactable, non-player character. He is located in the bar section, filling beer glasses. He is added to the scene for theming purposes.

Environment

The experience takes place in a comedy club called “Wormhole Comedy Club”.

Gameplay

There are 2 main interactions during the experience. Heckling & Praising and answering questions.
During his performance, player can heckle or praise Fred by saying the words listed below. Fred’s mood increases or decreases accordingly.

Heckling (Negative Effect on Fred)	**Praising (Positive Effect on Fred)**
Shut up!	Keep it up!
(I’m) Bored! / (You’re) Boring!	You tell them!
Boo!	You can do it!

Tell some jokes!	(Great/Good) Job!
(It’s) Not funny!	

In order to increase the interactions, we included direct questions to the player. There are 3 types of questions: Yes/No, Either/Or, Open ended. We collected commonly used answers during our playtests to build our answer database. In addition to handling the answers in the database, we are handling ‘No answer’ and ‘Other answer’ as well. We are treating ‘What/Repeat that’ and ‘Who/Me/Who me’ different than ‘Other answer’:

- What/Repeat that: Causes repetition of the question starting with “I said…”
- Who/Me/Who me: Causes response of “Yeah you! Jello!”

We tried to pick the questions which seems open but still have limited amount of answers. Here is the list of questions and answer categories that we are using in our final version.

TYPE	**QUESTION**	**ANSWER CATEGORIES**
Y/N	Can you help me out for a few minutes?	Yes, No, Garbage, Silence
Y/N	Can I call you Jello?	Yes, No, Garbage, Silence
Open	Where are you from?	Pittsburgh, China, Mars, Texas, India, Mom, Garbage, Silence
Open	How do you like the weather here?	Good, Bad, Cold, Snow, Sun, Garbage, Silence
Open	What did you want to be when you grow up?	Fireman, Ballerina, Scientist, Artist, Astronaut, Player, Princess, Garbage, Silence
Open	What was your favourite subject in school?	Math, Science, Art, Reading, Language, Music, Gym, Studies, History, Recess, Garbage, Silence
E/O	Are you an introvert or an extrovert?	Introvert, Extrovert, Garbage, Silence
Open	What is your favourite color?	Orange, Blue, Red, Purple, Green, Pink, Black, Yellow, White, Grey, Garbage, Silence
Open	What is your favourite holiday?	Christmas, Thanksgiving, Halloween, New Year, Hanukkah, Spring Festival, Sunday, Garbage, Silence
Open	Where do you see yourself in 5 years?	Garbage, Silence

We built our system such a way that it is really easy to add/remove words. All we need to do is add the word in the XML file with the semantic value and reference the corresponding audio file. This system helped us iterate and test easily. For the grammar file, see Appendix A.

Art

Character Art

We explored variety of character designs at the beginning of the semester. We decided to use our character Fred since he has a more humanoid look and it is easier to show his emotion. Fred has big eyes and stylish skinny body, which contributes a lot to the expression of his performance.

Fred has not only body rigging, but also facial rigging, which could express the emotions better.

Since our character has 3 different moods, we made 3 different talking animations for him, which match his positive, neutral and negative statuses.

Fred also has immediate reactions to any interruption, the reaction animations are different based on his mood and whether the interruption is a heckling or praising.

Our story changes according to different inputs, Fred has 6 different responses animations for this situation.

Fred’s mouth doesn’t match perfectly with what he is saying, that’s because our script changed a lot, and the lip sync tool we find doesn’t support live animation in Unity, so we made basic talking animation for his mouth.

Environment Art

Here is the list of assets we used for building our environment.

Room: Brick wall, red interior wallpaper, wall lamps, club sign
Stage: Stage, curtain, microphone, stool, spot lights
Tables: Table, chair, candle, pizza, unopened beer bottle, opened beer bottle, glass of beer, glass of wine
Bar: Bar furniture, beer taps, beer glasses, bottles

All the assets except Fred, stage, microphone, club sign and wall lamps are open source. All animation is created by our artist.

Sound Design

In terms of sounds design there were 2 primary tasks:

1) Make the atmosphere of the comedy club natural and by that help the player to adjust to the surroundings and let him engage in gameplay without feeling weird
2) Record the voiceovers for the monologue, questions, responses and reactions, and afterwards edit it and process with alien-like effect.

Background Sounds

In the beginning of the semester, more as a placeholder, we created a sound for the club-restaurant like background noise. It did serve its purpose, but after playtesting we received feedback about the surroundings being unnaturally quiet.
To solve that problem we didn't just add more noise, but had some quiet laughter and hardly hearable conversations, which in combination with animated audience members created the effect we wanted to have.
One of the most important factors that made it work was the fact that we used 3D sounds - the noise doesn't come from one audio place, but instead, it is heard from many audio sources on different distances.
Another thing that helped to create an illusion of sitting in a room with other people was recording short shout-outs from different voice actors. These shout-outs served as interaction cues for the player for heckling or supporting the comedian by shouting specific phrases like “Shut up!” or “Keep it up!” at practically any time of the performance.

Voiceovers

Our main character, comedian Fred, is a humanoid, which makes him able to speak in advanced English, but he is not a human. Although we partially left it to the players to decide what is Fred's exact origin, he is designed as an alien. It would sound odd if we gave Fred a human voice considering his appearance, so we had to find a suitable sound effect to process his speech with.
It was a challenge to use the distortion correctly to achieve the perfect voice for Fred as we wanted the speech to be clear so that the guests can fully understand his performance and his questions. The solution to that was to flanger it with “Hell's chorus” effect which met our requirements. Fred's words are comprehensible and this effect also added mystery to his voice.
The biggest challenge in sound design was that we had a too many audio files to record. We had 6 iterations for the script and our scripts were not linear. In order to test and iterate quickly, we tried to use a speech synthesizer. After the first playtest, we realized that it wouldn't help us to get feedback about our script as it is highly unnatural and not easily understandable. That's why, for each script, we recorded the statements, questions and reactions. As we had 3 different mood states for Fred, this multiplied the amount of audio files by 3.

After we settled down on our script, it became easier for the sound design process as we only had to put additional lines and/or re-record the ones that didn't work well enough considering playtesting results.

Design Decisions

Oculus Rift

We decided to include Oculus Rift in our experience. We observed that it's breaking the wall between the player and the virtual environment. It felt so much better with Oculus Rift than trying to talk to a TV screen. Oculus Rift helped us to communicate the setting, it helped players to be more engaged in the atmosphere, the 3D club environment with other audience members around. Fred is also directly looking to the camera which made players feel like he is talking to them. We are also using Oculus with a headset, which helps us to engage players with 3D sound.
Our experience works without Oculus Rift and the headset. Though we strongly recommend to experience with these tools which increases the engagement, we use speakers and a stand alone microphone during our presentations, while presenting a number of people.

Lessons Learned

- Speech recognition is not a solved problem yet

Although technology certainly improved in recent years, it is still not enough to do projects that have natural language processing. Speech recognition for games at the moment is very limited, but as our project demonstrates, it is enough in order to make design decisions which allow to create an illusion of the system understanding what the player says.

- Data mining is crucial

In order to create the feeling that a virtual character understands the player, one of the best ways we found to help with this is playtesting, a lot of playtesting. When you limit your domain of interaction and research what people tend to say, the chances of you succeeding in creating the magic feeling of understanding and reacting to your input increase significantly.

- Comedy is hard

As our narrative direction we chose comedy, because we were all interested in exploring it. Although it was very exciting, and our research in comedy was an extremely interesting part of the project, it turned out to be an extremely difficult topic.

Not only humor is a very subjective thing from person to person, but working with different nationalities and understanding how different the concept of humor is in various countries, specifically made us realized that we made a very inconvenient choice. If we continued the project and wanted to explore the depth of the character and his believability, we would most certainly have chosen drama, or at least a more serious narrative.

Conclusion

In the beginning of the semester we settled three direct tasks for our project:
1) Voice input is recognized by the system
2) Players feel like the character understands them
3) Write a thorough documentation for future developers
We believe that we succeeded in all of these three points and we hope that this documentation of our project, which we tried to make as complete as possible, will be helpful for the future generation of developers.

Field of speech recognition is definitely not at it’s highest point yet, but we believe we proved that it is possible to create interesting projects in it already, and it is certainly worth exploring and innovating.

Appendix

A. Grammar

```
<?xml version="1.0" encoding="utf-8"?>
<grammar xml:lang="en-US" root="rootNode"
tag-format="semantics/1.0" version="1.0"
xmlns="http://www.w3.org/2001/06/grammar">

  <rule id="rootNode">
      <one-of>
            <item><ruleref uri="#sucksAnswers" /></item>
            <item><ruleref uri="#extrovertAnswers" /></item>
            <item><ruleref uri="#introvertAnswers" /></item>
            <item><ruleref uri="#hanukkahAnswers" /></item>
            <item><ruleref uri="#musicAnswers" /></item>
            <item><ruleref uri="#artAnswers" /></item>
            <item><ruleref uri="#historyAnswers" /></item>
```

```
<item><ruleref uri="#socialStudiesAnswers" /></item>
<item><ruleref uri="#recessAnswers" /></item>
<item><ruleref uri="#gymAnswers" /></item>
<item><ruleref uri="#scienceAnswers" /></item>
<item><ruleref uri="#mathAnswers" /></item>
<item><ruleref uri="#englishAnswers" /></item>
<item><ruleref uri="#readingAnswers" /></item>
<item><ruleref uri="#princessAnswers" /></item>
<item><ruleref uri="#greenAnswers" /></item>
<item><ruleref uri="#pinkAnswers" /></item>
<item><ruleref uri="#purpleAnswers" /></item>
<item><ruleref uri="#greyAnswers" /></item>
<item><ruleref uri="#whiteAnswers" /></item>
<item><ruleref uri="#blackAnswers" /></item>
<item><ruleref uri="#orangeAnswers" /></item>
<item><ruleref uri="#texasAnswers" /></item>
<item><ruleref uri="#pittsburghAnswers" /></item>
<item><ruleref uri="#yellowAnswers" /></item>
<item><ruleref uri="#blueAnswers" /></item>
<item><ruleref uri="#redAnswers" /></item>
<item><ruleref uri="#playerAnswers" /></item>
<item><ruleref uri="#artistAnswers" /></item>
<item><ruleref uri="#astronautAnswers" /></item>
<item><ruleref uri="#scientistAnswers" /></item>
<item><ruleref uri="#ballerinaAnswers" /></item>
<item><ruleref uri="#firemanAnswers" /></item>
<item><ruleref uri="#halloweenAnswers" /></item>
<item><ruleref uri="#springFestivalAnswers" /></item>
<item><ruleref uri="#newYearAnswers" /></item>
<item><ruleref uri="#thanksgivingAnswers" /></item>
<item><ruleref uri="#christmasAnswers" /></item>
<item><ruleref uri="#sunAnswers" /></item>
<item><ruleref uri="#snowAnswers" /></item>
<item><ruleref uri="#coldAnswers" /></item>
<item><ruleref uri="#badAnswers" /></item>
<item><ruleref uri="#goodAnswers" /></item>
<item><ruleref uri="#momAnswers" /></item>
<item><ruleref uri="#indiaAnswers" /></item>
<item><ruleref uri="#repeatQuestion" /></item>
<item><ruleref uri="#russiaAnswers" /></item>
<item><ruleref uri="#chinaAnswers" /></item>
<item><ruleref uri="#marsAnswers" /></item>
<item><ruleref uri="#yesAnswers" /></item>
<item><ruleref uri="#noAnswers" /></item>
```

```
			<item><ruleref uri="#positivePhrases" /></item>
			<item><ruleref uri="#negativePhrases" /></item>
			<item><ruleref uri="#sundayAnswers" /></item>
		</one-of>
	</rule>

	<rule id="sundayAnswers">
		<item><tag> out = "sunday"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>sunday</item>
			</one-of>
		</item>
	</rule>

	<rule id="sucksAnswers">
		<item><tag> out = "sucks"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>sucks</item>
			</one-of>
		</item>
	</rule>

	<rule id="extrovertAnswers">
		<item><tag> out = "extrovert"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>extrovert</item>
			</one-of>
		</item>
	</rule>

	<rule id="introvertAnswers">
		<item><tag> out = "introvert"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>introvert</item>
			</one-of>
		</item>
	</rule>

	<rule id="hanukkahAnswers">
		<item><tag> out = "hanukkah"; </tag>
```

```
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>hanukkah</item>
			</one-of>
		</item>
	</rule>

	<rule id="musicAnswers">
		<item><tag> out = "music"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>music</item>
				<item>band</item>
			</one-of>
		</item>
	</rule>

	<rule id="artAnswers">
		<item><tag> out = "art"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>art</item>
				<item>modeling</item>
			</one-of>
		</item>
	</rule>

	<rule id="historyAnswers">
		<item><tag> out = "history"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>history</item>
			</one-of>
		</item>
	</rule>

	<rule id="socialStudiesAnswers">
		<item><tag> out = "social studies"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>social studies</item>
			</one-of>
		</item>
	</rule>
```

```
<rule id="recessAnswers">
        <item><tag> out = "recess"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>recess</item>
                </one-of>
        </item>
</rule>

<rule id="gymAnswers">
        <item><tag> out = "gym"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>gym</item>
                        <item>physical education</item>
                        <item>p e</item>
                </one-of>
        </item>
</rule>

<rule id="scienceAnswers">
        <item><tag> out = "science"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>science</item>
                        <item>chemistry</item>
                        <item>biology</item>
                        <item>physics</item>
                </one-of>
        </item>
</rule>

<rule id="mathAnswers">
        <item><tag> out = "math"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>math</item>
                        <item>calculus</item>
                        <item>algebra</item>
                        <item>geometry</item>
                </one-of>
        </item>
</rule>
```

```
<rule id="englishAnswers">
        <item><tag> out = "english"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>english</item>
                        <item>latin</item>
                        <item>french</item>
                        <item>spanish</item>
                        <item>german</item>
                </one-of>
        </item>
</rule>

<rule id="readingAnswers">
        <item><tag> out = "reading"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>reading</item>
                </one-of>
        </item>
</rule>

<rule id="princessAnswers">
        <item><tag> out = "princess"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>princess</item>
                </one-of>
        </item>
</rule>

<rule id="greenAnswers">
        <item><tag> out = "green"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>green</item>
                </one-of>
        </item>
</rule>

<rule id="pinkAnswers">
        <item><tag> out = "pink"; </tag>
                <ruleref special="GARBAGE"/>
```

```
			<one-of>
				<item>pink</item>
			</one-of>
		</item>
	</rule>

	<rule id="purpleAnswers">
		<item><tag> out = "purple"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>purple</item>
			</one-of>
		</item>
	</rule>

	<rule id="greyAnswers">
		<item><tag> out = "grey"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>grey</item>
			</one-of>
		</item>
	</rule>

	<rule id="whiteAnswers">
		<item><tag> out = "white"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>white</item>
			</one-of>
		</item>
	</rule>

	<rule id="blackAnswers">
		<item><tag> out = "black"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>black</item>
			</one-of>
		</item>
	</rule>

	<rule id="orangeAnswers">
		<item><tag> out = "orange"; </tag>
```

```
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>orange</item>
			</one-of>
		</item>
	</rule>

	<rule id="texasAnswers">
		<item><tag> out = "texas"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>texas</item>
			</one-of>
		</item>
	</rule>

	<rule id="pittsburghAnswers">
		<item><tag> out = "pittsburgh"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>pittsburgh</item>
				<item>here</item>
			</one-of>
		</item>
	</rule>

	<rule id="yellowAnswers">
		<item><tag> out = "yellow"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>yellow</item>
			</one-of>
		</item>
	</rule>

	<rule id="blueAnswers">
		<item><tag> out = "blue"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>blue</item>
				<item>turqouise</item>
			</one-of>
		</item>
	</rule>
```

```
<rule id="redAnswers">
	<item><tag> out = "red"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>red</item>
		</one-of>
	</item>
</rule>

<rule id="playerAnswers">
	<item><tag> out = "player"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>player</item>
		</one-of>
	</item>
</rule>

<rule id="artistAnswers">
	<item><tag> out = "artist"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>artist</item>
		</one-of>
	</item>
</rule>

<rule id="astronautAnswers">
	<item><tag> out = "astronaut"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>astronaut</item>
		</one-of>
	</item>
</rule>

<rule id="scientistAnswers">
	<item><tag> out = "scientist"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>scientist</item>
		</one-of>
	</item>
```

```
</rule>

<rule id="ballerinaAnswers">
        <item><tag> out = "ballerina"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>ballerina</item>
                </one-of>
        </item>
</rule>

<rule id="firemanAnswers">
        <item><tag> out = "fireman"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>fireman</item>
                </one-of>
        </item>
</rule>

<rule id="halloweenAnswers">
        <item><tag> out = "halloween"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>halloween</item>
                </one-of>
        </item>
</rule>

<rule id="springFestivalAnswers">
        <item><tag> out = "spring festival"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>spring festival</item>
                </one-of>
        </item>
</rule>

<rule id="newYearAnswers">
        <item><tag> out = "new year"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>new year</item>
                        <item>new years</item>
                        <item>new years eve</item>
                </one-of>
        </item>
</rule>

<rule id="thanksgivingAnswers">
        <item><tag> out = "thanksgiving"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>thanksgiving</item>
                </one-of>
        </item>
</rule>

<rule id="christmasAnswers">
        <item><tag> out = "christmas"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>christmas</item>
                </one-of>
        </item>
</rule>

<rule id="sunAnswers">
        <item><tag> out = "sun"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>sun</item>
                </one-of>
        </item>
</rule>

<rule id="snowAnswers">
        <item><tag> out = "snow"; </tag>
                <ruleref special="GARBAGE"/>
                <one-of>
                        <item>snow</item>
                </one-of>
        </item>
</rule>

<rule id="coldAnswers">
        <item><tag> out = "cold"; </tag>
                <ruleref special="GARBAGE"/>
```

```
			<one-of>
				<item>cold</item>
			</one-of>
		</item>
	</rule>

	<rule id="badAnswers">
		<item><tag> out = "bad"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>bad</item>
			</one-of>
		</item>
	</rule>

	<rule id="goodAnswers">
		<item><tag> out = "good"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>good</item>
			</one-of>
		</item>
	</rule>

	<rule id="momAnswers">
		<item><tag> out = "mom"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>mom</item>
			</one-of>
		</item>
	</rule>

	<rule id="indiaAnswers">
		<item><tag> out = "india"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>india</item>
			</one-of>
		</item>
	</rule>

	<rule id="repeatQuestion">
		<item><tag> out = "REPEATQUESTION"; </tag>
```

```
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>repeat that</item>
				<item>what</item>
			</one-of>
		</item>
	</rule>

	<rule id="whoMe">
		<item><tag> out = "ASKINGME"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>me</item>
				<item>who</item>
				<item>who me</item>
			</one-of>
		</item>
	</rule>

	<rule id="russiaAnswers">
		<item><tag> out = "russia"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>russia</item>
			</one-of>
		</item>
	</rule>

	<rule id="chinaAnswers">
		<item><tag> out = "china"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>china</item>
			</one-of>
		</item>
	</rule>

	<rule id="marsAnswers">
		<item><tag> out = "mars"; </tag>
			<ruleref special="GARBAGE"/>
			<one-of>
				<item>mars</item>
			</one-of>
		</item>
```

```
</rule>

<rule id="yesAnswers">
	<item><tag> out = "yes"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>yes</item>
			<item>yea</item>
			<item>yup</item>
			<item>for sure</item>
			<item>okay</item>
			<item>sure</item>
		</one-of>
	</item>
</rule>

<rule id="noAnswers">
	<item><tag> out = "no"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>no</item>
			<item>nope</item>
			<item>nah</item>
			<item>never</item>
		</one-of>
	</item>
</rule>

<rule id="positivePhrases">
	<item><tag> out = "positive"; </tag>
		<ruleref special="GARBAGE"/>
		<one-of>
			<item>keep it up</item>
			<item>good job</item>
			<item>great job</item>
			<item>you tell them</item>
			<item>you can do it</item>
		</one-of>
	</item>
</rule>

<rule id="negativePhrases">
	<item><tag> out = "negative"; </tag>
		<ruleref special="GARBAGE"/>
```

```
                        <one-of>
                                    <item>shut up</item>
                                    <item>boring</item>
                                    <item>bored</item>
                                    <item>not funny</item>
                                    <item>tell some jokes</item>
                                    <item>boo</item>
                                    <item>get off the stage</item>
                                    <item>you're terrible</item>
                        </one-of>
            </item>
    </rule>

</grammar>
```

B. Playtester Data

TOTAL PLAYTESTERS		
Nationality	**Male**	**Female**
American	16	8
Turkish	0	2
Chinese	2	10
Portuguese	3	0
Korean	0	1
Mexican	1	0
Columbian	0	1
Indian	7	1

Austrian	0	2
South African	1	0
Taiwanese	2	0
Russian	1	0
TOTAL	**33**	**25**

Week 5		
Nationality	**Male**	**Female**
Turkish	0	1
Chinese	0	1
Russian	1	0
American	1	1

10/9/2013		
Nationality	**Male**	**Female**
American	4	2
Indian	1	0

10/10/2013		
Nationality	**Male**	**Female**
American	5	2
Indian	1	1
Chinese	1	2

11/1/2013		
Nationality	**Male**	**Female**

Portuguese	1	0
Chinese	1	2
Columbian	0	1
Austrian	0	1

11/7/2013		
Nationality	**Male**	**Female**
American	1	1
South African	1	0
Chinese	0	2
Taiwanese	1	0
Indian	1	0
Mexican	1	0
Portuguese	1	0

11/15/2013		
Nationality	**Male**	**Female**
American	4	2
Austrian	0	1
Chinese	0	1
Indian	1	0

11/22/2013		
Nationality	**Male**	**Female**
American	1	0
Turkish	0	1
Chinese	0	2

Portuguese	1	0
Korean	0	1
Indian	3	0
Taiwanese	1	0