EECS 481 Software Engineering
Scope and Requirements Document

Team SET: **Anthony Chiang, Bryant Ku, Steven Louie, Mark Mevorah, Deng Ke Teo, Jason Terranova, Steven Uy**

To: *Dr. David Chesney, Professor, EECS 481*

CC: *Mr. Chris McMeeking, Instructor, EECS 481*
Mr. David Sabourin, Instructor, EECS 481

From: *Team SET*
Anthony Chiang, Student
Bryant Ku, Student
Steven Louie, Student
Mark Mevorah, Student
Deng Ke Teo, Student
Jason Terranova, Student
Steven Uy, Student

Subject: *EECS 481 Software requirements document for eye tracking communication application for people with cerebral palsy*

Date: *8 October 2013*

Version	***Date***	***Revision***	***Author***
1	2 October 2013	First Draft	Steven Uy
2	5 October 2013	Second Draft	Deng Ke Teo
3	6 October 2013	Third Draft	Anthony Chiang

System Description

System Purpose

People with cerebral palsy are often unable to fully enjoy the benefits of technology. For example, because of their lack of motor skills, they struggle with operating devices such as tablets, personal computers or even household appliances such as microwave ovens. Moreover, due to their conditions, they are sometimes unable to properly interact with the people around them. Being able to interact with others is important for everyone, even for people with disabilities. To improve the quality of life of people with cerebral palsy, we propose a communication tool using eye-tracking technology that will allow cerebral palsy patients to potentially be more expressive in communicating their thoughts. The tool we are proposing leverages the eye-tracking capabilities of the Creative Intel Gesture Camera to enable people with cerebral palsy to be better at communicating.

Project Scope

The eye-tracking communication tool we will be developing will have features that will allow Grace and people with cerebral palsy to communicate expressively using eye motions. Our tool will feature multiple modes that allow for communication with the most generic mode being a keyboard

layout that will allow users to create text sentences. Other modes available include a customizable layout of expressions or commonly used phrases that will allow users to communicate their thoughts quickly. We also plan to integrate day-to-day online features into our tool such as e-mail functionality and have the tool be fully customizable.

The software development kit for the Intel Creative Camera is for the C++ programming language. That will be our main programming language for configuring the Creative Camera to capture users' eye motion and obtaining input from the users. For the GUI, we have chosen to work with the Windows Presentation Foundation (WPF) Framework which is in C# as it is highly configurable.

Stakeholders

This section introduces the stakeholders for our project and explains how each stakeholder is relevant to our product.

Grace and her parents

This project revolves around creating a tool that will help Grace, a teenage girl with cerebral palsy, in four key areas of her life, namely, play, communication, school-life and home-life. Since Grace's parents have a part to play in each one of these areas, their feedback and evaluation on the direction and progress of our tool is necessary for us to meet Grace's needs.

Cerebral palsy patients and their caregivers

Doctors who work with cerebral palsy patients can use our tool to perform empirical analyses on their patients and find out how much our tool will aid them in their everyday life and to find out more about their patients' needs. For cerebral palsy patients, this tool will help them like they will help Grace- it will allow them to be more expressive in their communication.

Dr. David Chesney

The instructor for EECS 481, Software Engineering, and overseer of all the projects in the class is Dr. David Chesney. He provides us with the necessary tools and guidance needed to come up with a final project. All forms of documentation regarding the project, as well as the development and testing of the different stages of the project have to undergo his evaluation.

Objectives

Our objective for this project is to have a working and properly tested eye-tracking communication tool that will allow people with cerebral palsy to express their thoughts beyond what they normally can. In Grace's case, the tool will allow her to form proper sentences to communicate with; thus enhancing her expressiveness and allowing her to communicate beyond binary gestures. We have until the end of this semester to complete the project, with our final demos and presentations being held on 13 December 2013. That translates to about two and a half months of work and we want our tool to at least have basic communicative functions such as a keyboard layout to create sentences with.

Requirements Enumeration

This section introduces the variety of requirements for our project, describing purpose and means of verification.

*Acronyms for numbering of requirements:

FC = *Functional*
NF = *Non-Functional*
IO = *Input Output*
DOC= Documentation

Name: Intel creative gesture camera
Number: IO-1
Description: USB High definition webcam capable of tracking depth, gestures, landmarks.
Verification: Landmarks on the face are located at the corners of each eye. This simple detection can verify if the camera is working properly.
Notes: May not need specifically the Intel creative gesture camera if using OpenCV, where any high-resolution camera can be used.

Name: Laptop screen (15” preferred) with speakers
Number: IO-2
Description: A laptop with screen size exceeding 15” is preferred as that will allow users to use the application more easily.
Verification: No verification needed.

Name: On-Screen Keyboard
Number: FC-1
Description: Implement an on-screen keyboard that gives users the option to use either a QWERTY or T9 layout.
Verification: Select every button and test for its expected behavior.

Name: Eye Tracking
Number: FC-2
Description: Cursor movement is controlled by tracking the pupil's focus on the screen using the Intel gesture camera.
Verification: Focus and pan across all points on the screen at varying speeds.

Name: Console
Number: FC-3
Description: The console is responsible for displaying words and sentences.
Verification: Ensure that all words or sentences typed are displayed properly.

Name: Text To Speech
Number: FC-4
Description: Outputs words or sentences displayed on the console in speech form.
Verification: Ensure that speech output is equivalent to text displayed on screen.

Name: Adjustable selection time
Number: FC-5
Description: Users will focus on buttons that they wish to select. After focusing for a period of time, the button will be confirmed and the associated action will be carried out.
Verification: Test by focusing on all buttons and observe if expected behavior occurs.

Name: Word completion
Number: FC-6
Description: As users type out words, the application will suggest words that may correspond to the one that they are typing. If so, the user can accept the recommendation and the completed word will be displayed on the console.
Verification: The word completion feature can be tested by measuring the number of words that

are completed out of a random sample of 1000 words drawn from the dictionary.

Name: Save user's settings
Number: FC-8
Description: After the user exits, settings such as the timeout for selection retains its value.
Verification: A simple test includes adjusting each setting to a value different than default, force quitting the program, restarting the computer, and re-opening the application.

Name: Tab organization
Number: FC-9
Description: Multiple input methods, T-9, Qwerty, common words, are organized in different views. To access other views, tabs on the top of the UI can be selected in the same fashion as letter selection.
Verification: Input a segment of a sentence using each of the methods, switching between each mid sentence.

Name: Calibration/User Test
Number: FC-10
Description: Have different points on the screen for the user to look at to verify pupil tracking is working successfully.
Verification: Verify the user test in several scenarios, involving lighting conditions, background movement, and different distances.

Name: Adjustable Idle timeout
Number: FC-11
Description: A timeout is set if the user pulls focus away the screen. This duration can be adjusted by the user for their needs.
Verification: Test several durations for idle timeout, including the max and minimum.

Name: Easy interface navigation
Number: NF-1
Description: The keyboards must be neatly designed with buttons large enough to select with decent accuracy. Additionally selecting word completion options or changing input method must be trivial to a new user.
Verification: Multiple test subjects of different backgrounds and ages, some with and without CP will test traversing the interface.
Notes: Refer to F-9 for organization details.

Name: Accommodation for head movement
Number: NF-2
Description: Many with CP have involuntary head movement. Pulling focus away from the screen may interfere with input.
Verification: Pull away from the camera with varying speeds with multiple users.

Name: Ease of use
Number: NF-3
Description: Using SET should be a smooth, natural, and simple experience. The tracking should be smoothed as well as cursor movement. Clear text and colors should direct user selection.
Verification: If possible testing with Grace, but testing with users with different backgrounds and experience using technology should help adjust the application.

Name: User manual
Number: DOC-1
Description: A more in depth document accessible from the application describing troubleshooting, setup, and basic directions for use.
Verification: Ask testers what additional information they would like to see in the documentation.

Name: Calibration/ User Test
Number: DOC-1
Description: At the start of a session, the user will be prompted for a calibration test to look at different points on the screen.
Verification: Use the calibration test in multiple settings with different users and verify the accuracy adjusted by the test.

Risk Management

This section details the risks expected and encountered throughout the project.

Risk Description: Grace's intermittent involuntary head movements may hinder the focus of her pupils on the screen.
Risk Detection: The group will observe and record how long it takes for her to return her head to the correct position.
Risk Avoidance: The risk will be lowered by adjusting the selection and focus of the pupils, with users deliberately turning their head away at varying speed to allow for adjustments.
Risk Mitigation Plan: To prevent accidental selection, as the camera detects any turn of the head outside a rotation range, the program will remain idle for a time period determined by the user until the head returns within the range and pupil's are focused on the screen.

Risk Description: Pupil tracking is impossible with the Creative Intel gesture camera.
Risk Detection: If the existing SDK for Creative Intel camera fails pupil tracking nearing a release date, a hardware change will be discussed.
Risk Avoidance: The risk will be decreased with early testing and research with the existing SDK for the Creative Intel camera.
Risk Mitigation Plan: Many hardware alternatives exist, such as the Kinect with an external lens or a high definition webcam.

Risk Description: Grace's eye movements may not be stable enough for our implementation.
Risk Detection: The group will work with Grace, observing her pupil focus while reading text on a screen.

Risk Avoidance: This risk can be avoided through extensive testing of the eye tracking feature to ensure that Grace's eye movements match up with the cursor on the screen. Initial testing without Grace will need to focus on simulating movements similar to Grace's.
Risk Mitigation Plan: If Grace's movements do not match our expectations, we will have to adjust the eye tracking to accommodate them.

Risk Description: Multiple objects or people in the background being detected may interfere with the eye tracking and selection.
Risk Detection: This risk can be detected through tests with the Intel camera such as multiple people or moving objects in the background.
Risk Avoidance: This risk could be avoided by designing application only to track the pupils of Grace and ignore background images.
Risk Mitigation Plan: If this risk does occur, the environment can be controlled without additional objects or eye tracking can be designed specifically for Grace.

Risk Description: Student who is recently fired decides to join our team
Risk Detection: Student requests to join the team.
Risk Avoidance: Refuse to allow student to join the team.
Risk Mitigation Plan: New team member will be assigned simple tasks until he or she is fully familiar with the project and the team's source code.

Risk Description: Loss of a team member
Risk Detection: As per the group contract, members may not leave the team one week prior release (alpha/beta/final). Moreover, they must inform all team members before leaving.
Risk Avoidance: A group meeting will be held and members will be deterred from leaving.
Risk Mitigation Plan: Using the Pivotal Tracker tool, tasks can be easily reassigned to the remaining members of the team.

Project management

Task breakdown

Our project has two main aspects: eye-tracking and user interface. For the majority of our application's development, these two aspects are decoupled, meaning that they can be developed independent of each other. Every few weeks, work done on these two aspects will be integrated to ensure coherency.

Because our team is relatively large, we have decided to divide our team into two sub-teams. Sub-teams model the Agile project management approach and work in parallel. Each sub-team can be working on one of the two aspects concurrently. This will reduce the inefficiency caused by a large team and allow us to capitalize on our team's size and develop rapidly.

Figures 1 and 2 are Gantt Charts showing the start and end dates per task split between the interface team and the eye tracking team. To manage these tasks in a structured manner, we are using Pivotal Tracker, an agile project management tool to keep track of our tasks.

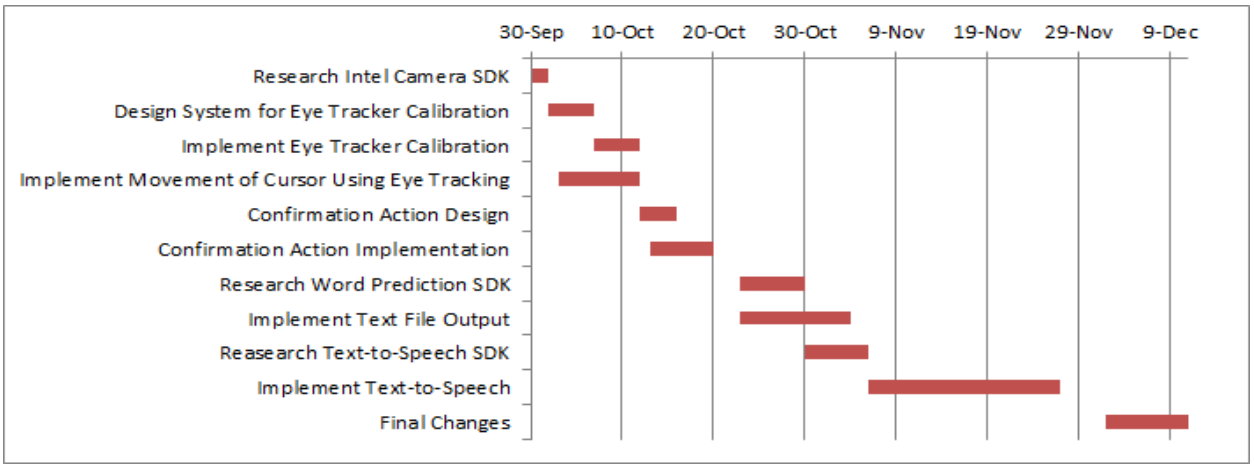

Figure 1: Schedule of eye tracking team's milestones and deliverables

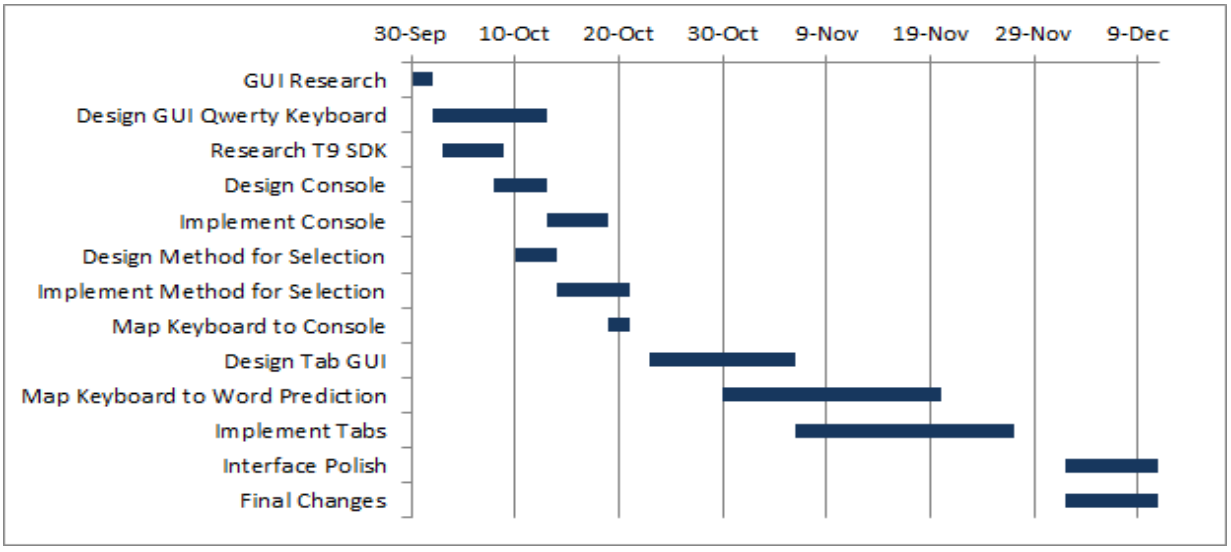

Figure 2: Schedule of UI team's milestones and deliverables

Assessment of progress to date

To date, the backend sub-team working on eye tracking has had to work with only one camera. Due to this, the backend team started off with inertia as members had to rotate the use of the camera. We have requested for another camera to work with and the course instructor has gladly agreed to provide us another one. This will lead to higher productivity.

Development standards and conventions

Functions, classes, and variables should follow the camel case standard. Classes should have their first letter capitalized and should be in singular form. Variables should have meaningful names, for example 'foo' and 'bar' are not particularly enlightening.

Containers should have plural names, for example a vector holding Person objects should be named people.

Examples:

```
void keyPress(//function arguments);
class KeyButton;
int counter;
vector< Person> people;
```

Version Control Standard

Our team will be using an open-source distributed Version Control System called "Git". With Git, every user has their own copy of the entire repository (including its entire history). This solves the problem that centralized Version Control Systems usually face which are:

1. If the central repository goes down, you can't commit code to the repository, nor can you pull code from the repository.

2. If the central repository explodes, you have lost the record of your code. Any previous versions are lost.
   Details of Git can be found in https://github.com

Environment Description

Development Environment:	Microsoft VisualStudio 2013 on Windows XP will be used.
Target Environment:	Windows 2000 or later will be the target environment. The software will not support Linux or Mac operating systems.
Software Tools:	[1] Git [2] Open CV or Intel Perceptual Computing SDK(Tentative) [3] Windows Presentation Foundation(WPF)

References

[1]Git library. Git.<http://libgit2.github.com/>