Design Document - Team Chameleon (Team 46)

Moinak Bandyopadhyay, Kevin Jones, Mudit Manu Paliwal, Jessica Blair, Jacob Solomon

Architecture and Rationale

Static Architecture

Architecture Diagram: https://docs.google.com/viewer?a=v&pid=explorer&chrome=true&srcid=1CcUrLeLQT5JttuW1I8WNzuU8By5DD50HioZFv9OmgoIdgInXft7O64sNTryS&hl=en_US

Coding File Structure:
https://docs.google.com/document/d/1tRtA2cboIoX4IpZAq44iURC6zAh7eG2Dj_TOdR4C8aE/edit

Dynamic Architecture

UML Sequence Diagram
http://www.lucidchart.com/publicSegments/view/4eb4b003-0d18-4483-a837-18ab0a7cfb84

The blocks in red are pages that we were not implement in this time frame.
Site Map: http://www.lucidchart.com/documents/view/4e9edbcf-ff3c-4030-b08f-4b030a7cd65c

Rationale Discussion

Initially, when we were brainstorming possible implementations for our system, we considered 3 platforms: a smart phone application, a stand-alone application and a web application. We decided not to pursue a smart phone application because a smart phone does not afford the level of interactivity our design requires. We concluded that smart phones do not allow us enough real estate to work with, provide less powerful means of uploading media, and are not commonly used by children with autism. According to Bernard-Opitz, Sriram, and Nakhoda-Sapuan (2001), children with autism are most well-versed with the use of computers and laptops. Making a smart phone application would therefore necessitate an unacceptable learning curve.

We had also considered a stand-alone Java application, but we rejected it due to the overhead required to install and update Java. We also wanted to give our solution platform-independence, so that the users could use several different devices (computers, laptops, tablets) to use the same application without any application overheads. Although Java applications can compile cross platform, it would require extra effort on our part to ensure that the user interface appeared correctly on every platform.

As a result, we have decided to develop a web application, which leverages the existing computer knowledge of children with autism and benefits from a web application’s high compatibility across operating systems and platforms.

Once we selected a platform for our system, we had to decide which languages we'd use to implement this system. We had 3 different packages in mind: Actionscript 3 with the Adobe Flex SDK, HTML5, or traditional HTML/CSS/Javascript/PHP. We ultimately selected HTML/CSS/Javascript/JQuery/PHP/MySQL for several reasons: First, we had some previous experience of using some of these technologies, leading to a smoother learning curve. Secondly, because these tools are standard for web development, there is a lot of documentation to help us tackle issues that arise during the development process.

We did not go with Adobe Flex because there is no server-side interface functionality built into ActionScript. We would need to implement our own methods for passing data between Flex and our backend (Java, .NET, PHP etc.) There are libraries out there to AMF remoting with just about any backend which makes it easy. But as far as the objects we would have to pass between the two, and we would have to maintain both or use a code generation tool to create AS3 classes from our backend classes. We did not want this excess overhead. We rejected HTML5 due to a lack of broad browser support and the lack of documentation. Compared to the time-tested HTML4 standard, HTML5 is new and experimental.
Thus, we thought keeping the skill set of our team and the requirements of our project in mind, the traditional package would be best suited for us.

Once we had the implementation and the languages/technologies selected, we had to think of how to structure our system. We decided to go with a traditional client-server model. Due to our budget limitations, we decided to utilize server space provided the Technology Service Organization (TSO) at Georgia Tech. Using the TSO server space, we're hosting a web server, a MySQL database, and a subversion server. The files on the web server are split between a folder for the caregiver site and onefor the child site. We also had to make another important data design decision - whether to store the actual media associated with the users on the file server or to be stored within the database. We decided to store media our site uses within the MySQL database in the form of 'mediumblob', to ease the implementation of user-uploaded media.

Data Design
Emotional Trainer will have a central database, utilizing a client-server model. All the data will be stored in a central database server and the different nodes will pull the data in from this server. There is an attached information flow diagram that explains how the information flows through the system. There is also an attached relational database schema chart attached that describes the schema of the central database. All the persistence of data is handled through this central database.

Information Flow Diagram
https://docs.google.com/leaf?id=0B4fvGrF5tDhyNzJkODEyOTctZTkwYi00YmNhLWI2YjMtMGIzNjAwMDJiNTZm&hl=en_US

Relational Database Schema
https://docs.google.com/leaf?id=0B4fvGrF5tDhyZTlhNDFiOTEtOTYxNi00NjI2LWFkNmMtYzU5MTNmZjlhZjcw&hl=en_US

Detailed Design

Website Sitemap
https://www.lucidchart.com/documents/view#4e766dea-b358-40d7-9292-585a0a7e1344?branch=f51999da-95a4-41a6-a2f3-4cc572ad1d8d

User Interface Design
https://docs.google.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BzwwzlWBN8XNZDUwODc3ZTItZWVlNS00OWUzLTk2YTctMzFmZGQ5NzBlNjdh&hl=en (UI Sketches)

Our user interface design focuses on two distinct modes of interaction: one for the caregiver, and one for the child. The caregiver's interface is designed to be easy for a neurotypical adult to use, giving them quick access to important features like managing a child's profile and making the sophisticated task of organizing media simple. The child's interface, on the other hand, is designed to be exciting to interact with, and avoids relying too much on language to convey information.

Validation Plan

The first part of our validation plan assesses the quality of the system architecture itself. We plan to evaluate the entire system to insure its compliance with the elements of SOLID object-oriented design. This process includes the following procedures:

1. **Single Responsibility Principle.** We will examine every object in the UML diagram to assure that it has a single responsibility. If a class appears to be too monolithic, we will break it up into smaller sub-classes and evaluate each of those. We'll treat every JavaScript or PHP file we create as a single class.
2. **Open-Closed Principle.** We will inspect every object to make sure it has low coupling with the classes it works with. This will allow us, and any future teams working on this project, to easily implement new functionality without modifying existing code.
3. **Liskov Substitution Principle.** In every parent-child hierarchy, we will verify that the child objects can be inserted in place of the parent wherever the parent might be used. If they do not, we will alter the offending hierarchies.
4. **Interface Segregation Principle.** We will ensure that classes do not depend on any interfaces they do not use. (For example, a Test class would not derive from a TimedObject interface, as not all Tests are timed.) Violating classes will be corrected.
5. **Dependency Inversion Principle.** We will check to make sure that high level models do not depend on low level models through proper use of abstract interfaces. Interfaces will be added to these hierarchies as appropriate.

Secondly, we will validate the quality of our user interface design through the following usability criteria. Time permitting, we hope to perform a usability study with our target audience (5-10 year olds with autism and their caregivers) to assess these criteria and potentially discover new ones.

Usability Goal	Measure By

Provide a convenient way to upload media	After logging in, user should be able to start an upload within one minute.
Users (both caregivers and chilren) should be able to distinguish important colors in the interface.	In the post-study questionnaire, participants will be asked how effectively the user interface makes use of color. Responses should average a minimum of 3.5 for us to consider the interface's use of color sufficiently effective.
Caregivers and children should enjoy interacting with the interface.	In the Post-Study Questionnaire, participants will be asked "On a scale from 1 to 5, where 1 is 'very unenjoyable' and 5 is 'very enjoyable', how engaging was your experience using this application?" Responses should average a minimum of 3.5 for us to consider the interface sufficiently enjoyable.
Children should receive comprehensible feedback on their test performance.	During the usability study, a proctor will ask the child if he or she got the last question right or wrong after the third question they answer. At least 95% of our study sample should be able to answer this correctly.
Minimize inconvenient scrolling.	At no point should the interface display more information than can be displayed without horizontally scrolling on a 1024x768 display. The child's interface should not require any scrolling whatsoever. The caregiver's interface should never include more than 3 pages' worth of vertical scrolling in any area other than documentation and media galleries.
The website should keep the user informed of the system status.	Any action that takes more than 0.5 seconds to process should be accompanied by a visual indicator of progress.