Introduction

After a false start with the now-abandoned “Personal Doctor” project, our team decided to pursue a project based on one of Moinak Bandyopadhyay’s research papers composed for a course on educational technology. The system we would create, Emotional Trainer, was aimed at children between the ages of four and eight who have been diagnosed with autism. A caregiver of one such child would register for Emotional Trainer, create an account for their child, and upload personally meaningful media to use to teach the child how to identify the emotions that other people express. Unfortunately, as we continued working on this system, we discovered that implementing it fully was beyond the team’s time restrictions and limited web development expertise.

This document describes our team’s process of working on Emotional Trainer: what went right, what went wrong, lessons for our future projects, and lessons for future CS4911 students. We hope our experience can help guide a future group in the right direction.

Sincerely,
Team Chameleon Designs (Fall 2011)

Kevin Jones - Project Manager
Moinak Bandyopadhyay - Development Lead/Domain Researcher
Jessica Blair - UI Design Lead
Mudit Manu Paliwal - System Logic Programmer
Jacob Solomon - UI Programmer

Historical Overview

Following is a summary of our work on a sprint-by-sprint and week-by-week basis. This reflects the difference between what we planned and what we actually accomplished.

Sprint 1 - The focus of this sprint was determining requirements, the initial vision documentation, and setting up development environment.

- Week 1 (ending Aug 26) - In Week 1, we assembled a team, decided on a project, assigned team roles, established a meeting schedule, and created a T-Square site. At this point we began pursuing the Personal Doctor project, a mobile phone application which enables users to keep track of their own medical records. Note that our initial team structure was very different than what it eventually became: at this time, Jacob was project manager, Mudit was development lead, Kevin was user support/documentation lead, and Moinak was QA lead. While Jessica was established as the user experience leader at this team and retained that role throughout the project, the roles of the other team members shuffled about as we grew more experienced working as a group and became more familiar with our strengths and weaknesses.
- Week 2 (ending Sep 2) - We sought out a sponsor and made contact with Dr. Neil Shulman, sponsor of a handful of other senior design projects. We met with him that Friday to discuss our proposal. We had planned to create a list of user stories in Pivotal Tracker at this time, but weren’t able to do this due to our late initiation of our meeting with Dr. Shulman.
- Week 3 (Sep 9) - The meeting with Dr. Shulman didn’t go as we expected. Because of the team’s lack of knowledge in the medical field and ignorance of health insurance laws such as HIPAA, we determined that Personal Doctor was not a feasible project for us to pursue. We established the web application Emotional Trainer as our new project and reassigned team roles. At this time, Jacob became development lead, Mudit

became testing lead, Kevin became project manager, and Moinak became the “program manager”, responsible for keeping the project vision on track and acting as an external point of contact. We met with our new sponsor, Dr. Rosa Arriaga of the College of Computing, to discuss our project on Friday. However, we still weren’t able to establish our list of user stories.

- Week 4 (Sep 16) - We finally drafted a list of features and began managing them using Pivotal Tracker. We also enumerated a list of pages we'd need to develop and sketched out the UI for these pages. At this time, we delegated work to compose some of the deliverable documentation required by CS4911, and started trying to reserve server space from TSO to host our application.
- Week 5 (Sep 23) - We reviewed the user interfaces and reduced the complexity of our implementation by removing unnecessary features from our feature list. However, Dr. Arriaga was out of the country, so we had to prioritize our list of features without her feedback. Note that we have not yet completed the process of reserving server space from TSO.

Sprint 2 - The focus of this sprint was on our initial development effort. We completed caregiver account registration and enabled them to create child accounts.

- Week 6 (Sep 30) - Jacob and Mudit selected tools for our development environment: Dreamweaver as our IDE, using its integrated SVN functionality and Github to host our code. (We would later regret this decision, as Dreamweaver does *not*, in fact, have an SVN integrated with Git.) We decided to use PHP, MySQL, Javascript, and JQuery to manage the dynamic content of our site. Most of the actual work this week was dedicated to preparing our presentation and delivering it on Friday. However, we also got sidetracked trying to decide on a name for the child's interface. The process for brainstorming alternatives and comparing their merits was very time consuming and left us less time to discuss our implementation assignments. Note that we have still not yet completed the process of reserving server space from TSO.
- Week 7 (Oct 7) - Although Jacob and Mudit planned to configure Dreamweaver with our Github configuration over the weekend, further inspection revealed that the SVN functionality of Dreamweaver was not compatible with a Github repository. We reviewed our initial UI sketches with Dr. Arriaga and revised the site map to scope down the system to focus more on optimizing the experience for a caregiver with a single child and simplifying the media interface. Note that we have *still* not yet completed the process of reserving server space from TSO.
- Week 8 (Oct 14) - We implemented frameworks for the login page, registration page, create child page, and configured the database on a local server. We also started some basic Javascript validation to ensure the form works as intended. Finally, we succeeded in setting up an SVN and web server with TSO for use during development. The fact that establishing this development environment took four weeks was a real headache. We should have made it a higher priority so that we could begin sharing our code as early as possible. Even at this stage, we weren't able to configure access to these resources until the following week.
- Week 9 (Oct 21) - We configured SVN, FTP, and Database administration access for all team members, using the web space as a test server to verify the functionality of PHP code, rather than requiring all members to host a PHP server on a local machine. We completed coding tasks including a structurally complete login page, placeholder code for registration, form validation scripts, and our first CSS templates.
- Week 10 (Oct 28) - We completed the caregiver login page. The registration page was mostly completed; it's missing only some back-end functionality. We met on Sunday for an extended coding session, during which we finished up registration, profile image upload, and child profile creation.

Sprint 3 - In this sprint our goal was to complete the caregiver interface. We fell short of this, but have a mostly-functional child profile and media management page.

- Week 11 (Nov 4) - We organized a plan of work for Sprint 3: we aimed to refine our system’s requirements by contacting an occupational therapist, and send out a survey to collect information about our potential users and assemble a reasonable set of stock images for users to browse.

- Week 12 (Nov 11) - We created the stock image survey and emailed it to our sponsor for distribution. We also implemented session variables to keep track of the user who's currently logged in.
- Week 13 (Nov 18) - We contacted an occupational therapist and planned to schedule a meeting with her once we got a response to her email. Jessica created a set of navigation icons, which we integrated into the site. We also discovered how to create PHP header files, which helped reduce the redundancy of our code.
- Week 14 (Nov 25) - Because this was the week of Thanksgiving break, we did not plan much work. We sent an email to the OT to schedule an interview to discuss the efficacy of our system. Because of the low response rate to our stock images survey, we've decided to assemble an arbitrary set of stock photos to use in the Sprint 3 demo. We would later discover that our sponsor had forgotten to send out the email with the link to the survey--we added more stock images based on the survey results after it was sent out. We also had trouble completing a lot of our coding because the TSO web server had its error messages disabled.
- Week 15 (Dec 2) - Most of the development effort occurred in an extended coding session on Saturday. We managed to set up a local WAMP server to debug errors which were causing some of our PHP scripts to halt completely, which turned out to be relatively minor syntax errors. We also updated the database structure to simplify the process of uploading media. However, progress was slow, and we were unable to implement most of the features we had planned for this session.

Project Data

Organizational Structure

Following is a list of team members and a summary of their responsibilities. In our team contract, we envisioned a shared leadership structure in which each member assumed leadership in their primary area of responsibility. (Vision, Development, Testing, Documentation, and User Experience) Ultimately, however, we collaborated in a progressively less formal fashion in which every team member simply did what needed to be done. The only distinctions which were still held to were project manager and UI artist.

Moinak Bandyopadhyay : Development Lead/Domain Researcher

- Development Lead
- Databse/back-end development
- Point of Contact - External
- Domain Research
- Specifying Requirements

Jessica Blair : User Experience Lead

- Presentation Coordinator
- UI Design
- UI Art/Look and Feel
- Usability Studies

Kevin Jones : Project Manager

- Meeting Coordinator

- Meeting Notes Scribe
- Point of Contact - Internal
- Status Report Scribe
- Deliverable Courier
- T-Square Administrator
- Google Docs Administrator
- Pivotal Tracker Administrator
- Development Environment Administrator (SVN, FTP, MySQL)
- Team Contribution Verifier

Mudit Manu Paliwal : System Logic Developer

- Code Documentation & Coding Standards Compliance
- Unit/Integration Testing Lead
- User Documentation

Jacob Solomon : UI Developer

- UI Development/Integration
- Usabillity Studies
- Release Notes

Dr. Rosa Arriaga : Project Sponsor

- Guided the team's development efforts along a domain-relevant path
- Provided consultation on UI design, system methodology, etc

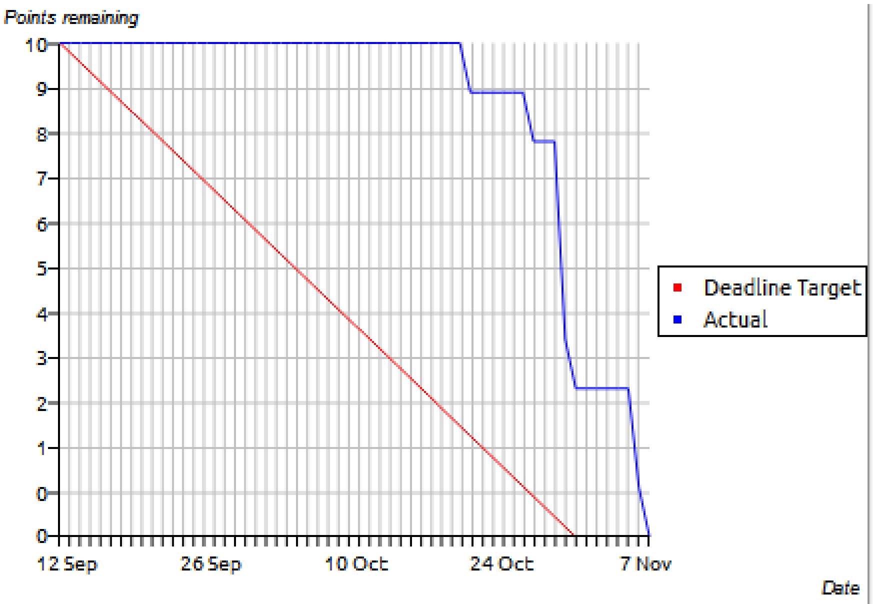
Prof. Bob Waters : Project Advisor

- Ensured the team's software development process satisfied the requirements of CS4911
- Evaluated the quality of the system's documentation, functionality, etc, and assigned grades accordingly

Schedule Data

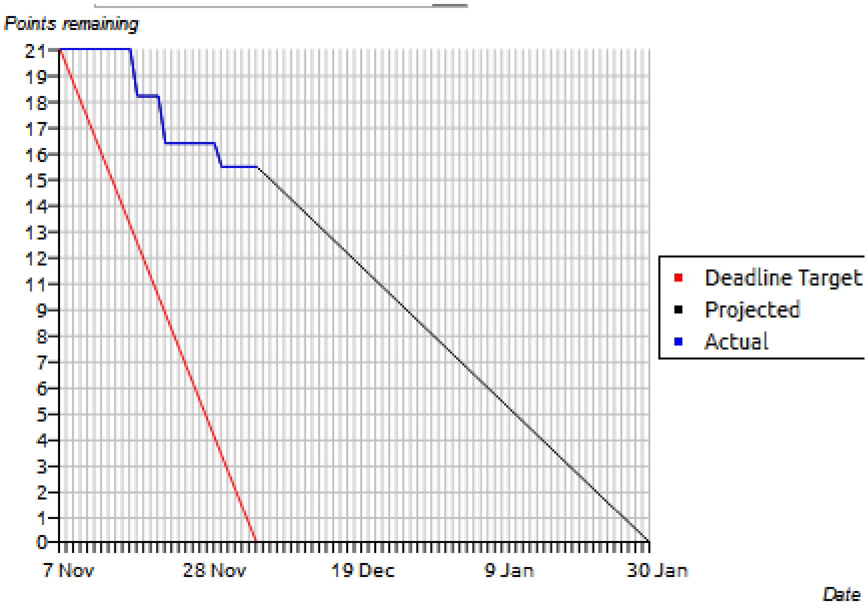
For Sprint 1, we concerned ourselves mostly with chores and planning documentation, rather than implementing features with value to the user. Therefore, we have no estimates of our planned schedule vs actual in terms of story points. However, we completed 10 chores on pivotal tracker related to documentation.

For Sprint 2, we planned to complete 10 story points before October 31st. However, we were only able to complete 8 out of these 10 before the due date. The burn-down chart follows.



This release finished on 7 Nov 2011, missing its deadline of 31 Oct 2011.

For Sprint 3, we moved features into the backlog aggressively, since we expected the velocity to ramp up in accordance with out changed methodology in writing out user stories. This did not occur. We planned to complete 21 story points and have only completed 5 of them as of the December 5th deadline. The burn-down chart follows.



This release is projected to finish on 29 Jan 2012, missing its deadline of 5 Dec 2011.
There are 16 points remaining in this release.

Lines of Code

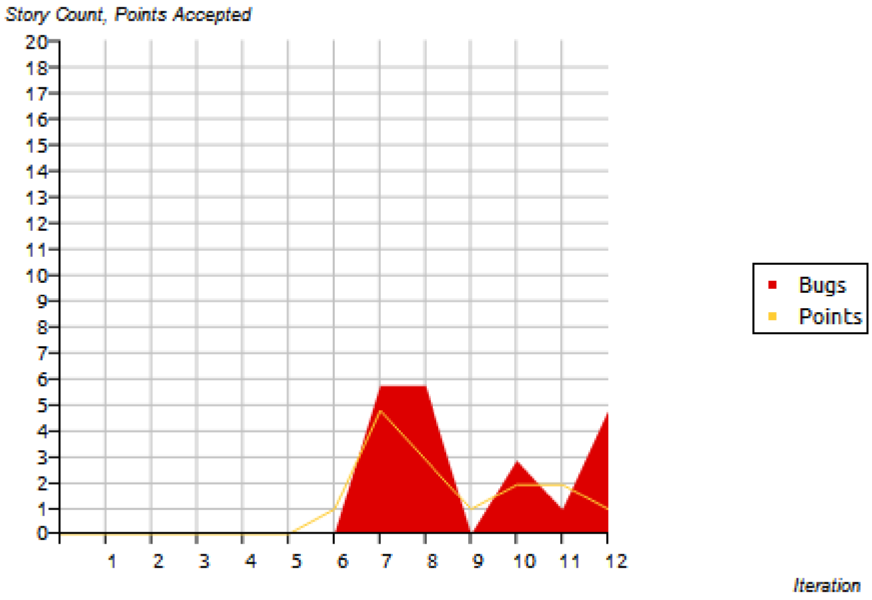
The following table includes a list of lines of code, excluding comments and whitespace, by file extension. These figures have been rounded to the nearest hundred. Please note that, due to the nature of web development, a single document may contain a mixture of HTML, PHP, Javascript, and CSS code. This also excludes the JQuery library and other libraries we used in our project.

	*.html	*.js	*.php	*.css	**Total**
Sprint 1	0	0	0	0	**0**
Sprint 2	600	200	400	2500	**3700**
Sprint 3	100	200	1200	2600	**4100**

As you can see, between Sprint 2 and Sprint 3, we moved a lot of code out of HTML documents and into PHP documents. This allowed us to integrate them with dynamic content based on the current session. We introduced PHP include headers, so there's less code repeated from page to page. The reason our CSS files are so numerous is because several online resources we utilized, such as the styles for the registration page, included their own CSS templates which we modified for our own use.

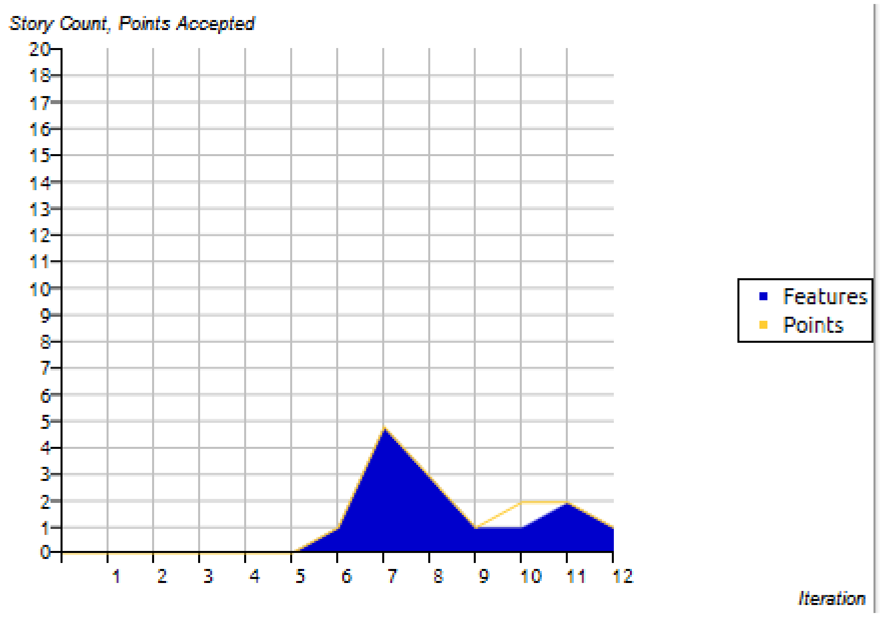
Bug Count

In the average weekly iteration, 1.2 features, 3.8 chores, and 1.8 bugs are accepted. We closed a total of 21 bugs during our work on this project, leaving behind 15 observed bugs open for the next group. We will continue working to eliminate bugs until the semester ends. Following is a chart of bugs accepted in each pivotal tracker iteration.

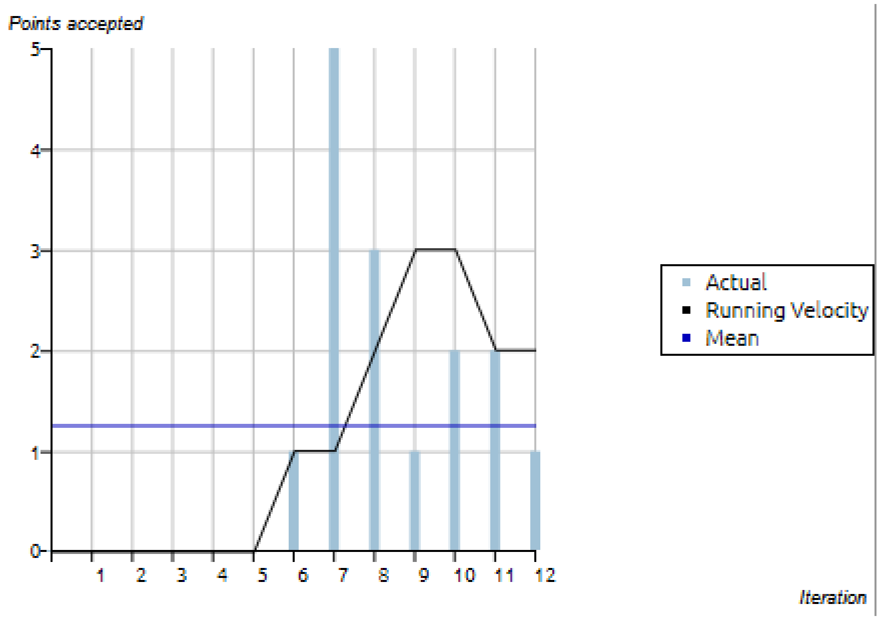


Features

We proposed a total of 57 user stories for this project. We completed 14 of them, with 11 remaining in our backlog for Sprint 3 and 32 in the icebox. Following is a chart of user stories we accepted each week by their story point values.



Our final velocity was **2**. This is calculated as a 3-iteration average. The overall mean velocity is 1.3, with a standard deviation of 1.5. The following chart illustrates the way our velocity varied from week to week.



Lessons Learned

Although the final outcome of our labor does not live up to our vision for the project, the process of developing it has proven a valuable experience for us. Following are the main highlights of this journey.

- **Pivotal Tracker.** Pivotal Tracker has proven itself to be a valuable tool for keeping track of tasks we need to complete as a team and estimating the amount of work we'll be able to get done in a specified period of time. However, we want to emphasize a few key practices that will help you get more out of it. First of all: spread your ideal feature set into as many independent user stories as you possibly can. This will allow you to measure smaller pieces of progress effectively and let people on your team work independently without getting wrapped up in dependencies. Secondly: Plan releases. Make sure you include the documentation requirements associated with each release as chores. This will give you visible milestones to work towards and scope your project out accordingly. Third: Never bet on a spontaneous increase in velocity. It's more effective to make a realistic plan of work and use the extra time to keep your code clean, fix bugs, and refine requirements.
- **Choosing a project.** Our group was very excited about the option to choose our own project for senior design, since we had collaborated previously in CS3750 in similar circumstances. However, we lost about three weeks' worth of development effort due to our false start, and had to do a lot of extra work to refine our requirements. I would recommend any group in a similar position to only go forward with a custom project if you've already established what you want to do, who you want to do it with, and who your advisor is prior to the start of the semester. Otherwise, you simply won't have enough time to get everything you need to do completed. Speaking of which...
- **Non-technical work is still work.** A lot of our effort this semester was spent doing domain research, creating a survey, contacting experts in the domain, and revising our user interfaces. None of this manifested directly in code or completed user stories in Pivotal Tracker. However, it was still valuable: the goal is not just to make a system with a bunch of features and rigorously optimized procedures, but a product that provides real value to the people who are going to use it. When you're assessing your team's progress on your senior design project, don't despair if your code doesn't seem to be going anywhere for a week or two if your team is still putting effort into the project and learning more about it.
- **Web development environment.** We learned the hard way how important it is to have a completely functional development environment as soon as possible, to enable proper evaluation of code and collaboration. If you're doing a web development project, set up your production server and source control ASAP. Do this even before you've refined the requirements for your project. Furthermore, you should be aware that the PHP configuration on TSO's servers has debugging messages turned off. It is probably worth the effort to establish a server on a local machine for the purpose of debugging database issues and PHP scripts.
- **Team communication.** Based on our experiences this semester, we've concluded that a chatty team is a well-coordinated one. Err on the side of sending extra emails or text messages to your teammates so you can keep track of who's doing what and the progress on it. If you're working with a teammate's code, either meet up with them physically or get in touch with them through an instant messaging tool like Skype so they can quickly answer questions you have about their code. It is also wise to include a procedure in your team contract for re-assigning work if a particular member of your team stops responding to emails, misses meetings, and can't be reached by phone, especially if a lot of future features depend on that person's assignment.
- **Division of labor.** Do not attempt to divide labor on a web development project based on programming languages. Everybody in the team should be familiar with every language you are using. HTML, PHP, JavaScript, SQL, and CSS are like the five fingers of web development, and you don't want to type code with a missing finger.

- **Surveys.** If it’s possible for you to distribute your survey without action on the part of your advisor, then do so. Do not ask your advisor to do something that you can do for yourself. Make sure that, if using a survey to collect information, you distribute it early enough to make use of the data you collect.