Product/Report Title Here

Senior Design Final Documentation

DanceSoft

Marcus Berger   Dicheng Wu

November 8, 2015

Contents

List of Figures

List of Tables

List of Algorithms

Overview Statements

0.1 Mission Statement

The mission of the DanceSoft team is to create a efficient and effective system for the Academy of Dance Arts. So data can be managed clearly and easily.

0.2 Elevator Pitch

The DanceSoft project is a data management software for the Academy of Dance Arts in Rapid City. The project aims to produce a simple and more effective data management tool then what is currently in use at the academy. This will be accomplished through the use of a database and a simple user friendly interface, which will allow faculty and students to easily accomplish their needs, whether that's registering for a class, getting a role sheet or just general people management. DanceSoft will provide effective management tools so people spend less time at their computer, and more time dancing.

Document Preparation and Updates

Current Version [X.X.X]

Prepared By:
Marcus Berger #1
Dicheng Wu #2

Revision History

Date	Author	Version	Comments
2/2/12	Team Member #1	1.0.0	Initial version
3/4/12	Team Member #3	1.1.0	Edited version

1

Overview and concept of operations

The overview should take the form of an executive summary. Give the reader a feel for the purpose of the document, what is contained in the document, and an idea of the purpose for the system or product.

1.1 Scope

What scope does this document cover?

1.2 Purpose

What is the purpose of the system or product?

1.3 Deliverables

1.3.1 Major System Component: Database

The first major component of the system is the MySQL relational database. The database will contain the Academy's data and be the core of the back-end side of the software, and be the conduit for most of the systems interactions with the data. The database will live within a local computer provided by the user.

1.3.2 Major System Component: User Interface

The second major component of the system is the front-end user interface. This will be the only part of the system most users ever see, and will provide an effective means to complete the desired user task. This is accomplished through the use of pages created in PyQt with interfaces to give users effective ways to interact with the database and the necessary data for their requested operation.

1.3.3 Major System Component #3

Describe briefly the role this major component plays in this system.

1.4 Systems Goals

The system needs to provide a solution which can run the dance studio data and some day to day activities in an effective and secure manner. This includes allowing teachers to print role sheets, look at schedules, and manage their information. Students need to have the ability to see information pertinent to them such as registration and class requirements. Owners need to be able to use the system to manage their employees, the academy's students and it classes, and other administrative duties such as billing and payroll Lastly this system as a whole also needs to be an improvement on the current system in use by the customer and provide an easier and more efficient way to run the clients business.

Overall the system goal is to provide a environment where academy owners, teachers, and students can effectively manage their personal needs and requirements for academy participation and continued operations.

1.5   System Overview and Diagram

Provide a more detailed description of the major system components without getting too detailed. This section should contain a high-level block and/or flow diagram of the system highlighting the major components. See Figure 1.1. This is a floating figure environment. LATEX will try to put it close to where it was typeset but will not allow the figure to be split if moving it can not happen. Figures, tables, algorithms and many other floating environments are automatically numbered and placed in the appropriate type of table of contents. You can move these and the numbers will update correctly.
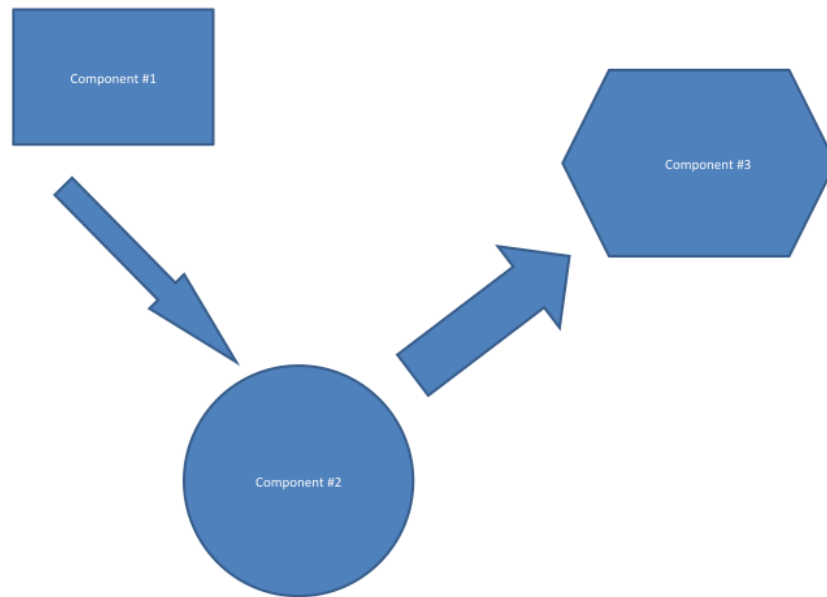
Figure 1.1: A sample figure .... System Diagram

1.6   Technologies Overview

The primary Technologies for this projects are as follows:

1. Xcode and Visual Studio - Xcode and MSVS are the primary development IDEs for this project. Of the two the one the group used the most was visual studio so we could develop on the machines provided by the school.

2. Python - the primary programming language for the project

3. PHP - primary language for student web interface

4. PyQt and Qt Designer - Gui package and development environment

5. MySQL - MySQL provides the database and relational quires to manage the data and organize it within the system.

These technologies were selected after a first research sprint where research into programming language, GUI, and database options were selected. A brief description of the research can be found in the sprint 1 report or the first prototype sections of this document.

2

Project Overview

This section provides some housekeeping type of information with regard to the team, project, etc.

2.1 Team Members and Roles

The DanceSoft team consist of two members, Marcus Berger and Dicheng Wu.

Marcus Berger(Scrum Master/Development Team) - As a member of a two person team the roles for this project blend together significantly. Both team members mostly do equal shares of all work types. As the primary manager of the DanceSoft trello board, Marcus has mostly taken on the role of scrum master within the group. However his primary role is still development.

Dicheng Wu(Development Team) - Dicheng's primary role as with both members of the team, is development of the software. However like the other members since the team is so small each member of the two person team must be able to fill all needed roles within the team.

Dr. Jeff Mcgough(Product Owner) - While not a working member of the team Dr. Mcgough is a secondary scrum master and product owner to the group due the small size of the team. Main duties in this role include talking to the client about what is needed, and making sure the team stays on task and is going in the correct direction based on the clients needs

2.2 Project Management Approach

This project is managed using several service and an Agile sprint based methodology. The sprints which range in purpose from research to development are three weeks long with a week after each sprint for review and possible presentations to the client. The project is broken into several users stories, which are then broken down into a project backlog of task for development. The stories and backlog are discussed in a later section of this document. Project management tools used by the team are Github, for source control and code management, and Trello for sprint and backlog management. Trello provide a list of task to be accomplished in each sprint and the ability to see modified tasks and keep track of team members current assignments.

This section will provide an explanation of the basic approach to managing the project. Typically, this would detail how the project will be managed through a given Agile methodology. The sprint length (i.e. 2 weeks) and product backlog ownership and location (ex. Trello) are examples of what will be discussed. An overview of the system used to track sprint tasks, bug or trouble tickets, and user stories would be warranted.

2.3 Phase Overview

If the system will be implemented in phases, describe those phases/sub-phases (design, implementation, testing, delivery) and the various milestones in this section. This section should also contain a correlation between the phases of development and the associated versioning of the system, i.e. major version, minor version, revision.

2.4   Terminology and Acronyms

Provide a list of terms used in the document that warrant definition. Consider industry or domain specific terms and acronyms as well as system specific.

3

User Stories, Backlog and Requirements

3.1 Overview

The overview should take the form of an executive summary. Give the reader a feel for the purpose of the document, what is contained in the document, and an idea of the purpose for the system or product.

The userstories are provided by the stakeholders. You will create the backlogs and the requirements, and document here. This chapter should contain details about each of the requirements and how the requirements are or will be satisfied in the design and implementation of the system.

Below: list, describe, and define the requirements in this chapter. There could be any number of subsections to help provide the necessary level of detail.

3.1.1 Scope

This section covers the purpose of systems, the client's information and restrictions on the system, and the user stories for requirements.

3.1.2 Purpose of the System

The purpose of this system is to provide a system for the Academy of Dance Arts to manage their day to day operations, their employees, and their s friendly students. These day to day operations range from assigning teachers to classes, looking up information, printing roles sheets for classes, etc. Also the system must accomplish these task using a user friendly interface, that is as intuitive as possible. Students will also have a simple web interface to look up their information and register for new classes.

3.2 Stakeholder Information

The stakeholder and sponsoring customer for this project is Dr. Jeff McGough a computer science professor at the South Dakota School of Mines and Technology and the vice president of the Academy of Dance Arts in Rapid City, South Dakota. Well not the sponsor of the project Dr. McGough's wife, Julie McFarland is also a key part of the customer base and a stakeholder as the owner of the Academy.

3.2.1 Customer or End User (Product Owner)

The primary end user for this product is Julie McFarland and her employees to manage the Academy of Dance Arts. Julie well not playing a direct role in product development is able to convey the academy's needs through Dr. McGough.

Dr. Jeff McGough is the primary point of contact in the project. He assumes the role of scrum master at times and drives the product backlog and provides more details on product backlog materials during the weekly meeting with the development team.

3.2.2   Management or Instructor (Scrum Master)

Dr. Jeff McGough is the primary point of contact in the project. He assumes the role of scrum master at times and drives the product backlog and provides more details on product backlog materials during the weekly meeting with the development team.

3.2.3   Developers –Testers

The DanceSoft team consist of two members, Marcus Berger and Dicheng Wu, who are both primarily developers and testers. due to the fact that the team is so small. All development roles are shared between the two team members.

3.3   Business Need

The customer needs us to develop a software solution which can run the dance studio in an effective manner. The product also needs to handle changing classes from year to year without needing to be updated. This means that the software needs to sync with multiple users, and handle new information such as class rosters, prices, clothing requirements for classes, changes in the employment roster, and many other changes that can occur in the running of a dance school.
This project as a whole needs to be an improvement on the current system in use by the customer and provide an easy and efficient way to run the clients business. This project will the data manipulation task through the back-end MySQL database, and the ease of use will be handled with a simple Qt Gui

3.4   Requirements and Design Constraints

This section discusses what requirements exist that deal with meeting the business needs the customer has. For the DanceSoft these include system needs to run the academy, network connection issues, and some environment requirements laid out by both the client and the senior design requirements.

3.4.1   System Requirements

The system requirements laid out by the clients are just the necessary features laid out in the user stories below. There was no preference on language or GUI environment on the part of the client. Due to some of the user stories and information handled within the system a level of security becomes a system requirement.

3.4.2   Network Requirements

The network inside the Academy has connection issues and therefore a cloud or online based data storage option is not a possibility. The network issues within the school mean the system will be contained in a local system to provide more stability.

3.4.3   Development Environment Requirements

The academy runs on a Mac so the system must work on the mac OSX operating system. While not required the project is developed in Python, so the end product will work cross platform should the academy ever switch operating systems.

3.4.4   Project Management Methodology

The client requires a weekly meeting every Wednesday at 1:00 p.m. to check on the progress of the system. These meeting vary on topic and length depending on the needs of the project and the status of task. The senior design class requires that this project be completed in six sprint that are all roughly three weeks long, with a week long results period after each one. Another project requirement is the presentations that are required by the senior design class. These presentation occurs twice every semester usually after the first and last sprint each semester. Each presentation cover the content of the project up to that point, and updates

on topics such as risk mitigation, budget, and current prototypes. Lastly it was requested by Dr. Mcgough as part of senior design and as the client that we provide him with access to the Github repository for the project and the Trello board.

3.5 User Stories

After the requirement for the project were laid out the team created the user stories based on those requirement. The user stories the team came up with are as fallows:

1. As a user i want to adjust students payment models

2. As the owner I would like to see automatic database backups.

3. As a student I would like to be able to register online (with special app). Classes must be approved before added.

4. As a student I would like to be able to search clothing requirements.

5. As a student I would like to know my billing.

6. As the owner I would like to indicate clothing requirements per class.

7. As a studio person, I would like to be able to add students to classes.

8. As a student, teacher etc, I would like to be able to look up a students class list.

9. As the teacher I would like to get a class role for each class.

10. Given a class list, I would like to get an invoice of the tuition due.

11. Studio would like to track payments and estimate remainder due. I would like to generate an invoice for this amount.

12. As a student I would like to be able to register online (with special app). Classes must be approved before added.

13. As a student I would like to know my billing.

14. As the owner I would like to track teacher hours and compute payroll.

15. As the owner I would like to indicate clothing requirements per class.

16. As a student I would like to be able to search clothing requirements.

17. As the owner I would like to see automatic database backups.

3.5.1 User Story #1

User story #1 discussed.

3.5.1.a User Story #1 Breakdown

Does the first user story need some division into smaller, consumable parts by the reader? This does not need to go to the level of actual task definition and may not be required.

3.5.2 User Story #2

3.5.2.a User Story #2 Breakdown

User story #2 ....

3.5.3   User Story #3

3.5.3.a   User Story #3 Breakdown

User story #3 ....

3.6   Research or Proof of Concept Results

Before production could begin research had to be conducted into which programming language, GUI framework, and database type would be used to complete the project. A explanation of the research conducted can be found in the sprint one wrapper or in the prototype sections of this document. After this research was conducted the team selected Python, PyQt, and MySQL as the language, GUI, and database respectfully. After the research no explicit proof of concept was required.

4

Design and Implementation

This section is used to describe the design details for each of the major components in the system. Note that this chapter is critical for all tracks. Research tracks would do experimental design here where other tracks would include the engineering design aspects. This section is not brief and requires the necessary detail that can be used by the reader to truly understand the architecture and implementation details without having to dig into the code. Sample algorithm: Algorithm 1. This algorithm environment is automatically placed - meaning it floats. You don't have to worry about placement or numbering.

Algorithm 1 Calculate $y = x^n$

Require: $n \geq 0 \vee x \neq 0$
Ensure: $y = x^n$
  $y \Leftarrow 1$
  if $n < 0$ **then**
    $X \Leftarrow 1/x$
    $N \Leftarrow -n$
  else
    $X \Leftarrow x$
    $N \Leftarrow n$
  end if
  while $N \neq 0$ **do**
    if N is even **then**
      $X \Leftarrow X \times X$
      $N \Leftarrow N/2$
    else $\{N$ is odd$\}$
      $y \Leftarrow y \times X$
      $N \Leftarrow N - 1$
    end if
  end while

Citations look like [2, 1, 3] and [6, 4, 5]. These are done automatically. Just fill in the database `designrefs.bib` using the same field structure as the other entries. Then pdflatex the document, bibtex the document and pdflatex twice again. The first pdflatex creates requests for bibliography entries. The bibtex extracts and formats the requested entries. The next pdflatex puts them in order and assigns labels. The final pdflatex replaces references in the text with the assigned labels. The bibliography is automatically constructed.

4.1 Major Component #1

4.1.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.1.2 Component Overview

This section can take the form of a list of features.

4.1.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.1.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.1.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.1.6 Design Details

This is where the details are presented and may contain subsections. Here is an example code listing:

```c
#include <stdio.h>
#define N 10
/* Block
 * comment */

int main()
{
    int i;

    // Line comment.
    puts("Hello world!");

    for (i = 0; i < N; i++)
    {
        puts("LaTeX is also great for programmers!");
    }

    return 0;
}
```

This code listing is not floating or automatically numbered. If you want auto-numbering, but it in the algorithm environment (not algorithmic however) shown above.

4.2 Major Component #2

4.2.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.2.2   Component Overview

This section can take the form of a list of features.

4.2.3   Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.2.4   Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.2.5   Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.2.6   Design Details

This is where the details are presented and may contain subsections.

4.3   Major Component #3

4.3.1   Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

4.3.2   Component Overview

This section can take the form of a list of features.

4.3.3   Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

4.3.4   Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

4.3.5   Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

4.3.6   Design Details

This is where the details are presented and may contain subsections.

5

System and Unit Testing

This section describes the approach taken with regard to system and unit testing.

5.1 Overview

Provides a brief overview of the testing approach, testing frameworks, and general how testing is/will be done to provide a measure of success for the system.

5.2 Dependencies

Describe the basic dependencies which should include unit testing frameworks and reference material.

5.3 Test Setup and Execution

Describe how test cases were developed, setup, and executed. This section can be extremely involved if a complete list of test cases was warranted for the system.

6

Development Environment

The purpose of this section is to give a developer the necessary information to setup their development environment to run, test, develop and/or update the DanceSoft software.

6.1 Development IDE and Tools

Two main IDEs were used in the this project were Microsoft Visual Studio 2015, and Xcode 6. Python IDLE would also be used on occasion for minor quick fixes so the main IDEs would not need to be loaded completely. Of these the bulk of development was conducted with Visual Studio due to the fact that the laptops provided by South Dakota School of Mine and Technology run windows as their primary operating system. Visual Studio also provided a suite of debugging and testing features that allowed the team to manage and manipulate the code effectively.

The second IDE used was Xcode which is the primary development environment for the Mac operating system. This IDE was used when ever we want to directly test Mac compatability with our code. Since Mac is the required working operating system for this project. Though due to accessibility Xcode was not the Main IDE used by the team. Should the project be futher developed in the future IDE selection should not matter due to the cross-platform development of the project.

1. Visual Studio install:

2. Visual Studio Reference:

3. Xcode install:

4. Xcode Reference:

5. IDLE install:

6. IDLE Reference:

6.2 Source Control

The source control for this product is conducted through Github. The github repo is broken down into a final code folder, a documentation folder, and six sprint folders which contain the materials worked on in each sprint for senior design grading purposes. Connects and manage were done through the provided Github GUI, which is installed when Github is downloaded. If the GUI was insuffient the group would sometimes use the shell also provided with the install. The repo is public and can be seen by anyone by going to `https://github.com/SDSMT-CSC464-F15/dancesoft/`

6.3   Dependencies

Currently the project contains two known dependencies. The first is in the PyQt4 Python library which if change could cause issues within the system GUI. However this seem unlikely since the main focus of PyQt updates is now focused on PyQt5. Second is the project dependence on MySQL relational database. This dependency should also be negligible since any update to MySQL are normally done with continuing compatibly with existing software in mind.

6.4   Build Environment

NOT KNOWN AT THIS TIME, WILL ADD LATER

How are the packages built? Are there build scripts?

6.5   Development Machine Setup

Setup requirements to develop on this project:

1. Download Python 3:

2. Download necessary IDE from the links provided above or another source

3. Download MySQL:

4. Download the PyQt4 Python library:

5. Connect to the MySQL database using credentials received from the client

6. Get the python scripts and any makefiles from the DanceSoft Github repo

7. Run and test script functionality

8. Begin development updates

7

Release – Setup – Deployment

This section should contain any specific subsection regarding specifics in releasing, setup, and/or deployment of the system.

7.1 Deployment Information and Dependencies

Are there dependencies that are not embedded into the system install?

7.2 Setup Information

How is a setup/install built?

7.3 System Versioning Information

How is the system versioned?

8

User Documentation

This section should contain the basis for any end user documentation for the system. End user documentation would cover the basic steps for setup and use of the system. It is likely that the majority of this section would be present in its own document to be delivered to the end user. However, it is recommended the original is contained and maintained in this document.

8.1 User Guide

The source for the user guide can go here. You have some options for how to handle the user docs. If you have some `newpage` commands around the guide then you can just print out those pages. If a different formatting is required, then have the source in a separate file `userguide.tex` and include that file here. That file can also be included into a driver (like the senior design template) which has the client specified formatting. Again, this is a single source approach.

8.2 Installation Guide

8.3 Programmer Manual

9

Class Index

9.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

10

Class Documentation

10.1 Poly Class Reference

Public Member Functions

- Poly ()
- ∼Poly ()
- int myfunction (int)

10.1.1 Constructor & Destructor Documentation

10.1.1.a Poly::Poly ( )

My constructor

10.1.1.b Poly::∼Poly ( )

My destructor

10.1.2 Member Function Documentation

10.1.2.a int Poly::myfunction (int *a*)

my own example function fancy new function
    new variable
    The documentation for this class was generated from the following file:

- hello.cpp

11

Business Plan

11.1 Business Model

11.2 Market and Competition

11.3 Regulatory environment

11.4 Intellectual Property and Freedom to Operate

11.5 Management Team and Advisors

11.6 Sources and Uses of Capital

11.7 Financial Statements

11.8 Metrics and Milestones

11.9 Exit Plan

12

Experimental Log

For research projects one needs to keep a log of all research/lab activities.

10/15/15 Ran modified filter on data sets 1 - 6. Results were ...

10/17/15 Changed tolerance on sensor and collected data. These ...

13

Research Results

This chapter describes the results and conclusions of your research. This would be the final report for a research project.

13.1 Result 1

13.2 Result 2

13.3 Conclusions

13.4 Further work

Bibliography

[1] R. Arkin. *Governing Lethal Behavior in Autonomous Robots*. Taylor & Francis, 2009.

[2] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, June 2005.

[3] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[4] V. Lumelsky and A. Stepanov. Path planning strategies for point mobile automation moving amidst unknown obstacles of arbirary shape. *Algorithmica*, pages 403–430, 1987.

[5] S.A. NOLFI and D.A. FLOREANO. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. A Bradford book. A BRADFORD BOOK/THE MIT PRESS, 2000.

[6] Wikipedia. Asimo — Wikipedia, the free encyclopedia. `http://upload.wikimedia.org/wikipedia/commons/thumb/0/05/HONDA_ASIMO.jpg/450px-HONDA_ASIMO.jpg`, 2013. [Online; accessed June 23, 2013].

SDSMT SENIOR DESIGN
SOFTWARE DEVELOPMENT AGREEMENT

This Software Development Agreement (the "Agreement") is made between the SDSMT Computer Science

Senior Design Team   DanceSoft

Consisting of team members Dicheng Wu and Marcus Berger,

,

and Sponsor   Jeff McGough,

with address: 501 E St Joseph St, Rapid City, SD 57701.

1 RECITALS

1. Sponsor desires Senior Design Team to develop software for use by dance studio and dance schools in South Dakota.
2. Senior Design Teams willing to develop such Software.

NOW, THEREFORE, in consideration of the mutual covenants and promises herein contained, the Team and Sponsor agree as follows:

2 EFFECTIVE DATE

This Agreement shall be effective as of October 1, 2015

3 DEFINITIONS

1. "Software" shall mean the computer programs in machine readable object code form and any subsequent error corrections or updates supplied to Sponsor by Senior Design Team pursuant to this Agreement.
2. "Acceptance Criteria" means the written technical and operational performance and functional criteria and documentation standards set out in the DanceSoft Senior Design Documentation
3. "Acceptance Date" means the date of the South Dakota School of Mines and Technology Senior Design Fair, when all Deliverables have been accepted by Sponsor in accordance with the Acceptance Criteria and this Agreement.
4. "Deliverable" means a deliverable specified in the DanceSoft Senior Design Documentation section 1.3
5. "Delivery Date" shall mean, the end of the spring 2017 semester on which University has delivered to Sponsor all of the Deliverables in accordance with section 1.3 of the DanceSoft Senior Design Documentation and this Agreement.

6. "Documentation" means the documents, manuals and written materials (including end-user manuals) referenced, indicated or described in the DanceSoft Senior Design Documentation or otherwise developed pursuant to this Agreement.

7. "Milestone" means the completion and delivery of all of the Deliverables or other events which are included or described in section 1.3 of the Dancesoft Senior Design Documentation scheduled for delivery and/or completion on a given target date; a Milestone will not be considered completed until the Acceptance Date has occurred with respect to all of the Deliverables for that Milestone.

4 DEVELOPMENT OF SOFTWARE

1. Senior Design Team will use its best efforts to develop the Software described in [the project plan.] The Software development will be under the direction of or his/her successors as mutually agreed to by the parties ("Team Lead") and will be conducted by the Team Lead. The Team will deliver the Software to the satisfaction of the course instructor that reasonable effort has been made to address the needs of the client. The Team understands that failure to deliver the Software is grounds for failing the course.

2. Sponsor understands that the Senior Design course's mission is education and advancement of knowledge, and, consequently, the development of Software must further that mission. The Senior Design Course does not guarantee specific results or any results, and the Software will be developed only on a best efforts basis. The Software is considered PROOF OF CONCEPT only and is NOT intended for commercial, medical, mission critical or industrial applications.

3. The Senior Design instructor will act as mediator between Sponsor and Team; and resolve any conflicts that may arise.

5 COMPENSATION

No COMPENSATION will occur.

6 CONSULTATION AND REPORTS

1. Sponsor's designated representative for consultation and communications with the Team Lead shall be Jeff McGough or such other person as Sponsor may from time to time designate to the Team Lead.

2. During the Term of the Agreement, Sponsor's representatives may consult informally with course instructor regarding the project, both personally and by telephone. Access to work carried on in University facilities, if any, in the course of this Agreement shall be entirely under the control of University personnel but shall be made available on a reasonable basis.

3. The Team Lead will submit written progress reports. At the conclusion of this Agreement, the Team Lead shall submit a comprehensive final report in the form of the formal course documentation at the conclusion of the Senior Design II course.

7 CONFIDENTIAL INFORMATION

1. The parties may wish, from time to time, in connection with work contemplated under this Agreement, to disclose confidential information to each other ("Confidential Information"). Each party will use reasonable efforts to prevent the disclosure of any of the other party's Confidential Information to third parties for

   a period of three (3) years after the termination of this Agreement, provided that the recipient party's obligation shall not apply to information that:

   (a) is not disclosed in writing or reduced to writing and so marked with an appropriate confidentiality legend within thirty (30) days of disclosure;

   (b) is already in the recipient party's possession at the time of disclosure thereof;

   (c) is or later becomes part of the public domain through no fault of the recipient party;

   (d) is received from a third party having no obligations of confidentiality to the disclosing party; (e) is independently developed by the recipient party; or (f) is required by law or regulation to be disclosed.

2. In the event that information is required to be disclosed pursuant to subsection (6), the party required to make disclosure shall notify the other to allow that party to assert whatever exclusions or exemptions may be available to it under such law or regulation.

8 INTELLECTUAL PROPERTY RIGHTS

All intellectual property created for use in the DanceSoft project is property of the South Dakota Board of Regents.

9 WARRANTIES

The Senior Design Team represents and warrants to Sponsor that:

1. the Software is the original work of the Senior Design Team in each and all aspects;
2. the Software and its use do not infringe any copyright or trade secret rights of any third party.

No agreements will be made beyond items (1) and (2).

10 INDEMNITY

1. Sponsor is responsible for claims and damages, losses or expenses held against the Sponsor.

2. Sponsor shall indemnify and hold harmless the Senior Design Team, its affiliated companies and the officers, agents, directors and employees of the same from any and all claims and damages, losses or expenses, including attorney's fees, caused by any negligent act of Sponsor or any of Sponsor's agents, employees, subcontractors, or suppliers.

3. NEITHER PARTY TO THIS AGREEMENT NOR THEIR AFFILIATED COMPANIES, NOR THE OFFICERS, AGENTS, STUDENTS AND EMPLOYEES OF ANY OF THE FOREGOING, SHALL BE LIABLE TO ANY OTHER PARTY HERETO IN ANY ACTION OR CLAIM FOR CONSEQUENTIAL OR SPECIAL DAMAGES, LOSS OF PROFITS, LOSS OF OPPORTUNITY, LOSS OF PRODUCT OR LOSS OF USE, WHETHER THE ACTION IN WHICH RECOVERY OF DAMAGES IS SOUGHT IS BASED ON CONTRACT TORT (INCLUDING SOLE, CONCURRENT OR OTHER NEGLIGENCE AND STRICT LIABILITY), STATUTE OR OTHERWISE. TO THE EXTENT PERMITTED BY LAW, ANY

11 INDEPENDENT CONTRACTOR

For the purposes of this Agreement and all services to be provided hereunder, the parties shall be, and shall be deemed to be, independent contractors and not agents or employees of the other party. Neither party shall have authority to make any statements, representations or commitments of any kind, or to take any action which shall be binding on the other party, except as may be expressly provided for herein or authorized in writing.

12 TERM AND TERMINATION

1. This Agreement shall commence on the Effective Date and extend until the end of classes of the second semester of Senior Design (CSC 464), unless sooner terminated in accordance with the provisions of this Section ("Term").

2. This Agreement may be terminated by the written agreement of both parties.

3. In the event that either party shall be in default of its materials obligations under this Agreement and shall fail to remedy such default within thirty (30) days after receipt of written notice thereof, this Agreement shall terminate upon expiration of the thirty (30) day period.

4. Any provisions of this Agreement which by their nature extend beyond termination shall survive such termination.

13 ATTACHMENTS

Attachments A and B are incorporated and made a part of this Agreement for all purposes.

14 GENERAL

1. This Agreement constitutes the entire and only agreement between the parties relating to the Senior Design Course, and all prior negotiations, representations, agreements and understandings are superseded hereby. No agreements altering or supplementing the terms hereof may be made except by means of a written document signed by the duly authorized representatives of the parties.

2. This Agreement shall be governed by, construed, and enforced in accordance with the internal laws of the State of South Dakota.

15 SIGNATURES

Marcus Berger   10/1/2015

Marcus Berger   Date

10/1/15

Dicheng Wu   Date

Jeff McGough   Date

A

Product Description

Write a description of the product to be developed. Use sectioning commands as neccessary.

NOTE: *This is part of the contract.*

B

Publications

Research Track: This chapter will include any publications generated from the research. Most likely these will be preprints and one will just include the pdf.

C

Sprint Reports

1 Sprint Report #1

The sprint reports should be inserted here. Reports focus on process. Design elements can be inserted into the design chapter with the report discussing the design element in more of an overview fashion.

2 Sprint Report #2

3 Sprint Report #3

4 Sprint Report ...

D

Industrial Experience and Resumes

1   Resumes

Your resumes are included here. See the source file (industrial.tex) and uncomment the PDF includes to see how this works. If your resume is written in LaTeX then you can just insert the LaTeX source code.

2   ABET: Industrial Experience Reports

2.1   Name1

2.2   Name2

2.3   Name3

E

Acknowledgment

F

Supporting Materials

This document will contain several appendices used as a way to separate out major component details, logic details, or tables of information. Use of this structure will help keep the document clean, readable, and organized.

LaTeX Example

LaTeX sample file: Remove from submitted materials

1 Introduction

This is a sample input file. Comparing it with the output it generates can show you how to produce a simple document of your own.

2 Ordinary Text

The ends of words and sentences are marked by spaces. It doesn't matter how many spaces you type; one is as good as 100. The end of a line counts as a space.

One or more blank lines denote the end of a paragraph.

Since any number of consecutive spaces are treated like a single one, the formatting of the input file makes no difference to TeX, but it makes a difference to you. When you use LaTeX, making your input file as easy to read as possible will be a great help as you write your document and when you change it. This sample file shows how you can add comments to your own input file.

Because printing is different from typewriting, there are a number of things that you have to do differently when preparing an input file than if you were just typing the document directly. Quotation marks like "this" have to be handled specially, as do quotes within quotes: "'this' is what I just wrote, not 'that'".

Dashes come in three sizes: an intra-word dash, a medium dash for number ranges like 1–2, and a punctuation dash—like this.

A sentence-ending space should be larger than the space between words within a sentence. You sometimes have to type special commands in conjunction with punctuation characters to get this right, as in the following sentence. Gnats, gnus, etc. all begin with G. You should check the spaces after periods when reading your output to make sure you haven't forgotten any special cases. Generating an ellipsis ... with the right spacing around the periods requires a special command.

TeX interprets some common characters as commands, so you must type special commands to generate them. These characters include the following: $ & % # { and }.

In printing, text is emphasized by using an *italic* type style.

A long segment of text can also be emphasized in this way. Text within such a segment given additional emphasis with Roman *type. Italic type loses its ability to emphasize and become simply distracting when used excessively.*

It is sometimes necessary to prevent TeX from breaking a line where it might otherwise do so. This may be at a space, as between the "Mr." and "Jones" in "Mr. Jones", or within a word—especially when the word is a symbol like *itemnum* that makes little sense when hyphenated across lines.

Footnotes[1] pose no problem.

TeX is good at typesetting mathematical formulas like $x - 3y = 7$ or $a_1 > x^{2n}/y^{2n} > x'$. Remember that a letter like x is a formula when it denotes a mathematical symbol, and should be treated as one.

---

[1]This is an example of a footnote.

3   Displayed Text

Text is displayed by indenting it from the left margin. Quotations are commonly displayed. There are short quotations

> This is a short a quotation. It consists of a single paragraph of text. There is no paragraph indentation.

and longer ones.

> This is a longer quotation. It consists of two paragraphs of text. The beginning of each paragraph is indicated by an extra indentation.
> This is the second paragraph of the quotation. It is just as dull as the first paragraph.

Another frequently-displayed structure is a list. The following is an example of an *itemized* list.

- This is the first item of an itemized list. Each item in the list is marked with a "tick". The document style determines what kind of tick mark is used.

- This is the second item of the list. It contains another list nested inside it. The inner list is an *enumerated* list.

  1. This is the first item of an enumerated list that is nested within the itemized list.
  2. This is the second item of the inner list. LATEX allows you to nest lists deeper than you really should.

  This is the rest of the second item of the outer list. It is no more interesting than any other part of the item.

- This is the third item of the list.

You can even display poetry.

> There is an environment for verse
> Whose features some poets will curse.
>
> For instead of making
> Them do *all* line breaking,
> It allows them to put too many words on a line when they'd rather be forced to be terse.

Mathematical formulas may also be displayed. A displayed formula is one-line long; multi-line formulas require special formatting instructions.

$$x' + y^2 = z_i^2$$

Don't start a paragraph with a displayed equation, nor make one a paragraph by itself.

4   Build process

To build LATEX documents you need the latex program. It is free and available on all operating systems. Download and install. Many of us use the TexLive distribution and are very happy with it. You can use a editor and command line or use an IDE. To build this document via command line:

```
alta>  pdflatex SystemTemplate
```

If you change the bib entries, then you need to update the bib files:

```
alta>  pdflatex SystemTemplate
alta>  bibtex SystemTemplate
alta>  pdflatex SystemTemplate
alta>  pdflatex SystemTemplate
```

The template files provided also contain a Makefile, which will make things much easier.

Acknowledgment

Thanks to Leslie Lamport.