C-lyrics - A Word Cloud for Lyrics

Software Project Management Plan

Justine Cocchi	Kelsey Fargas	Mark Krant
jcocchi@usc.edu	kfargas@usc.edu	mkrant@usc.edu
Milad Gueramian	Jeff Kang	Séb Arnold
gueramia@usc.edu	kangjr@usc.edu	arnolds@usc.edu

9 February 2015

Contents

Executive Summary

C-lyrics is a public website that will generate a word cloud for any given artist based on the 250 most frequently used words that appear across all of the artist's published songs. This product will interface with the EchoNest API which will serve as the database from which we find and analyze the songs. By clicking on a specific word in the word cloud the user can see a list of all of the songs that word appears in and how frequently it occurs in each song. Furthermore, the user can click on any listed song title to see the complete lyrics for that song with the original word selected from the word cloud highlighted every time it appears.

C-lyrics is intended for use by the general public. There will be no login required and there is no stored history of previous searches. Because of this we will have very low memory requirements and can run the product off of one server. The user can access C-lyrics using any device running any OS. After typing in the artist name and selecting the submit button, the word cloud will be generated in approximately one second and will be able to be shared via Facebook.

1. Introduction

1.1 Project Overview

The primary objective of this project is to better understand the software development lifecycle. For this specific project the objective is to create a software system that allows a user to search for an artist of their choice and generate a word cloud containing the most popular used words from their lyrics database. This program will be called C-lyrics, and is tentatively set to be delivered and readily available for public use by March 11, 2015.

There are three major milestones that will be completed and have subsections under each. These milestones consist of the completion of all requirements set for each interface page, such as the home page, songs page, and lyrics page. More details on these milestones can be read in section 4.2. The master schedule will include five phases of product development. These phases include designing the user interface, implementing each function within the interfaces, testing these interfaces, adding finishing touches, checking with the client, and submitting. A more detailed Gantt chart will be located in section 4.1 and a more detailed description of costs can be found in section 3.

By completion, contributing members of this project should be able to fully understand a product lifecycle as well as carrying out a complete deliverable to all stakeholders.

1.2 Project Deliverables

The contributing authors to this documentation are the members of the development team. They will create a fully functioning software program that will allow anyone to specify any artist and generate a word cloud comprised of that artist's most frequently used words. This software program will also have the functionality to list the specified artist's songs containing any word from the generated word cloud, and will allow the user to select one of these listed songs to display its lyrics. This program will be delivered as a product called C-lyrics. The client shall receive all documentation associated with the product as well as complete access and control of the product itself on March 11, 2015 at THH 208 and on Blackboard. Only a single copy of each deliverable shall be provided. These deliverables shall include software deliverables and document deliverables. The software deliverables will include a working software that will satisfy all set requirements and specifications made at the beginning the project, and delivered online through Blackboard. Document deliverables will include this current document as well as past and previous documents to completely satisfy the clients' request.

1.3 Evolution of the SPMP

The SPMP for C-lyrics will be available under a version control tool, GIT, monitored by all contributing members, so any changes will be made to the plan itself. The clients will be updated when significant changes have been made to the prototype. The client and development team will be able to access the updated document.

1.4 Reference Materials

[1] IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

[2] "word cloud". Oxforddictionaries.com (January 31, 2015)

[3] EchoNest API documentation. (January 29, 2015)

[4] Van Vliet, Hans. Software Engineering: Principles and Practice. 3rd ed. Indianapolis, Indiana: Wiley, 2008. 170-71.

1.5 Definitions and Acronyms

Term	Definition
AJAX	Asynchronous JavaScript And XML. Technology allowing the transfer of data from between the front- and back-end without reloading the web page.
API (EchoNest)	API will refer to the EchoNest API. EchoNest is a free API that allows developers to retrieve lyrics and artist information in web pages and other programs.
Autocomplete	Autocomplete refers to the functionality addition to the Search Bar, allowing users to enter minimal characters and choose artists that are most similar to the string and display a picture of those artists next to their name.

Autocomplete Delay	A feature designed for the search bar when a user is typing. The delay refers to the suspending action while the user is typing, making the request to the server for autocomplete.
Backend	References the PHP backend page
Back to home button	A button redirecting the user to the homepage.
Back to songs button	A button redirecting the user to the songs list page.
Commonly Used Web Browser	Browsers such as Firefox, Safari, Chrome, Explorer, and Quora which come on mobile phones, tablets and personal computers.
Customer/Client	Dr. William G. Halfond and Sonal Mahajan
GitHub	A web service that provides software version control tools. www.github.com
Stakeholders	The client and the development team
LOC	acronym: for Lines of Code
KSLOC	a metric that stands for: 1,000(K) Source Lines of Code
Desktop Platform	A screen whose width exceeds 560px
Development Team	All of the individuals whose names appear on the cover of this document. These persons have collectively put this document together and will collectively implement the software product described in subsequent sections.
Facebook	Online social network service where the generated word cloud image may be shared amongst users.

Google Doc	An online service provided by Google Inc. where an editable document can be accessed and change simultaneously by the members who have been given access to the document. In the case of the development team, google doc is the shared resource which contains the source of this SRS document.
Home Page	The first page of the website visited by the user. It contains the Word Cloud as well as the Search Bar.
Lyrics Page	The third page of the website, it contains the lyrics for one song, which is chosen by the user on the Songs Page. It will have two Navigation Buttons that can take the user to either the Home Page or back to the Songs Page.
Mobile Platform	A screen whose width is less than or equal to 560px
Navigation Buttons	Refers to any button that takes the user to previously visited pages of the website.
Software Project Management Plan (SPMP)	Refers to this document.
Prototype	A small prototype of the software including the barebones of the graphical display. Used during the second meeting with the client, screenshots available in the appendices.
Search Bar	The initial search bar on the first page of the website. Here, users can type in artist or band names to generate a word cloud.
Share Button	The standard, embeddable Facebook share button.
Software or Product	The application software delivered from the supplier to the customer.
Song List	This will be the culmination of all songs found that contain the search word indicated by the user.

Songs Page	The second page of the website. It contains the Song List as well as a Navigation Button back to the Home Page. The user navigates to the Songs page by clicking on a word in the Word Cloud on the Home page.
Submit Button	The button adjacent to the Search Bar. When the user enters an artist name into the Search Bar and is ready to generate the Word Cloud, he or she must click on the Submit Button to begin the process.
Supplier	The team developing the product for the customer.
System	The set of machines running the software making it accessible to the user.
User	A person who interacts with C-lyrics software
Word Cloud (WC)	A word cloud (otherwise known as a tag cloud) is, according to the Oxford Dictionary, an image composed of words used in a particular text or subject, in which the size of each word indicates its frequency or importance [2].

2. Process and Organization

2.1 Software Lifecycle Process

The software lifecycle process for this project will be based on the Waterfall Model. In this model, each phase of software development needs to be completed before attempting to work on the next phase. Testing is the last phase of the process which takes place after the software has been implemented.

2.2 Team Organization

The organizational structure of our team is flat with no specified leader or hierarchy. Our team is structured by collaboration across all members resulting in group decisions on who will work on each part of the overall project. All internal deadlines and deliverables are set as a team and communicated to

everyone during team meetings and emails. Each member of the team is equally responsible for every finished deliverable as there is no team lead for any specific component.

2.2.1 Current Group Process

The current group process in place for the development team is to meet as an entire group whenever it is deemed necessary by a general consensus. The consensus is gathered by means of group text messaging. At each meeting, the development team decides the tasks to be completed, progress made as of the time of the meeting and delegates future tasks to each member for completion. This process has proved itself highly effective as of the date of this publication and it is closely estimated to continue through the entire life cycle of the software development process for C-lyrics.

2.2.2 Staff and Personnel Plan

The C-Lyrics team will consist of the members of the development team. Each member is proficient in object oriented design and concepts, and has some familiarity with web development. Knowledge of git and other version control methods is a must. The duration of need will be until the last milestone and deliverable is reached. Each member will be assigned tasks based on his or her prior knowledge. Since there will be no training, members will have to research on the job if they are insufficiently prepared. Personnel will not be phased out and retention will not be a factor.

2.2.3 Staff Allocation

See *Figure 1* for a detailed view of the staff allocation.

2.2.4 Team Member Description

This section provides a brief description of the members of the development team and their relative experiences. Each member is a student at the University of Southern California and is affiliated with the Viterbi School of Engineering Computer Science Department. For additional information on each member, refer to appendix for resumes.

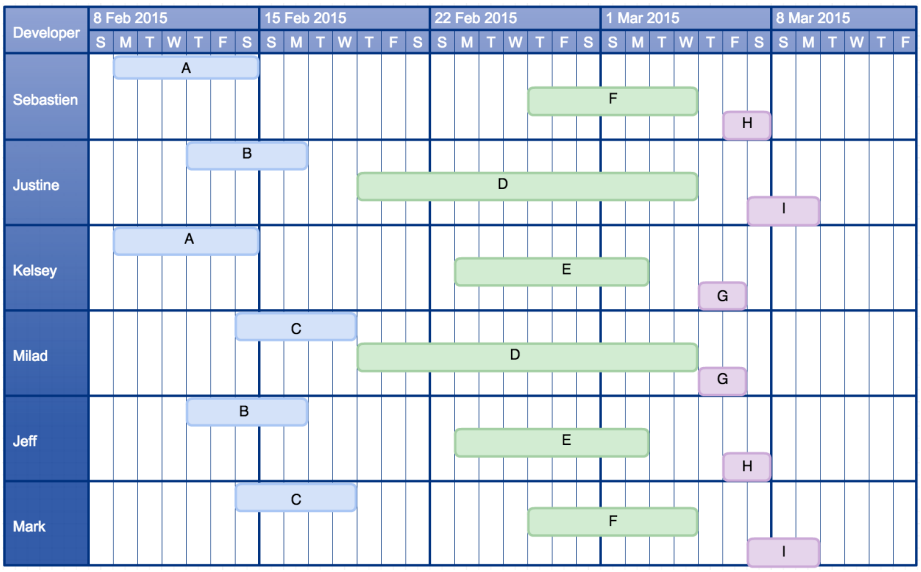

Figure 1: Staff Allocation

3. Cost Estimation

3.1 Cost Influence

The cost influencing factors that have been considered follow the standard COCOMO II model for estimating project effort. These include the four main categories of product factors, platform factors, personnel factors, and project factors. The detailed list of cost drivers and the associated estimations can be found in section 3.3 of this document.

3.2 Cost Schedule Relationship

Cost and schedule are related based on the fact that this particular project contains no defined wages in terms of fiscal costs. Therefore, the only real "cost" is the effort and time that will be put into creating the C-lyrics project. In other words, the schedule is perfectly correlated with the cost of the project.

3.3 Development Costs

Our development costs have been estimated using the COCOMO II Model. The following results are estimates of the cost drivers that are applied to the model [4].

Product Factors

Criteria	Value
Reliability required	very low, 0.75
Database size	low, 0.93
Product complexity	low, 0.88
Required reusability	nominal, 1.00
Documentation needs	very high, 1.13

The project requires very low reliability, since it would only pose a slight inconvenience if the C-lyrics platform encountered problems and was not available. We will most likely not need a database since the lyrics generation will be based on an online API, therefore the database size is listed as low. The low product complexity was selected based on an average of the nominal control operations level and the very low computational operations level. The reuse of components in the product is nominal, since components only need to be reused within the project. The documentation relative to the complexity of our project is very high.

Platform Factors

Criteria	Value
Execution-time constraints	nominal, 1.0
Main storage constraints	nominal, 1.0
Platform volatility	low, 0.87

The execution-time constraints are nominal since we will require less than 50% of available execution time. The main storage constraints are nominal since we will require less than 50% of available system storage. Platform volatility is low since the hardware and software required to run our program will not need frequent updates.

Personnel Factors

Criteria	Value
Analyst capability	very low, 1.50
Programmer capability	low, 1.16
Application experience	very low, 1.22
Platform experience	low, 1.10
Language and tool experience	low, 1.12
Personnel continuity	very high, 0.84

Analyst capability is very low since the group's ability to create detailed, high-level design for software projects is minimal. Programmer capability is low because of the lack of significant industry programming experience of the team. Language and tool experience is low since most of the team's programming

experience centers around one or two functional languages and little experience with software tools. Personnel continuity is very high since all members of the team will continue to be working together over the course of the project.

Project Factors

Criteria	Value
Use of software tools	low, 1.12
Multi-site development	very high, 0.84
Required development schedule	very high, 1.0

The use of software tools is designated as low because we will not be using highly developed tools that are structured for the purposes of our project. The multi-site development rating is very high, since the group has consistent communication both in person and through electronic channels. The required development schedule is listed as very high since the implementation phase of our project exceeds 160% of the expected time to complete the project.

The effort in terms of person-months is calculated as the KSLOC multiplied by the product of all cost drivers according to COCOMO II. With an estimated KSLOC of 1, the number of person-months that this calculation returns is 1.247.

3.4 External Costs

Some of the external costs that are not directly factored into the project costs include: purchasing the class textbook for research, programming resources, and transportation costs for group incured by individual members for meetings.

4. Schedule, Milestones, and Deliverables

4.1 Schedule

See *Figure 2* for a detailed view of the staff allocation.

4.1.1 Activity Network

See *Figure 3* for a detailed view of the staff allocation, and section 4.2 for the specific task encompassed in each milestone.

Task Name	Duration	Start	Finish
T1: Design of User Interface/Start	10	02/09/15	02/18/15
A: Create designs for Interface I: Home Page	6	02/09/15	02/14/15
B: Create designs for interface II: Songs Page	5	02/12/15	02/16/15
C: Create designs for interface III: Lyrics Page	5	02/14/15	02/18/15
T2: Implementation of Product	15	02/18/15	03/04/15
D: Implementation of Interface I: Home Page	15	02/18/15	03/04/15
E: Implementation of Interface II: Songs Page	8	02/23/15	03/02/15
F: Implementation of Interface III: Lyrics Page	7	02/26/15	03/04/15
T3: Testing Functionality	6	03/04/15	03/09/15
G: Testing of Interface I: Home Page	2	03/04/15	03/05/15
H: Testing of Interface II: Songs Page	2	03/05/15	03/06/15
I: Testing of Interface III: Lyrics Page	2	03/06/15	03/07/15
J: Testing Integration of all Interfaces	3	03/07/15	03/09/15
T4: Approval from Client	2	03/09/15	03/10/15
T5: Finishing Touches	2	03/10/15	03/11/15
T6: Submitting/End	1	03/11/15	03/11/15

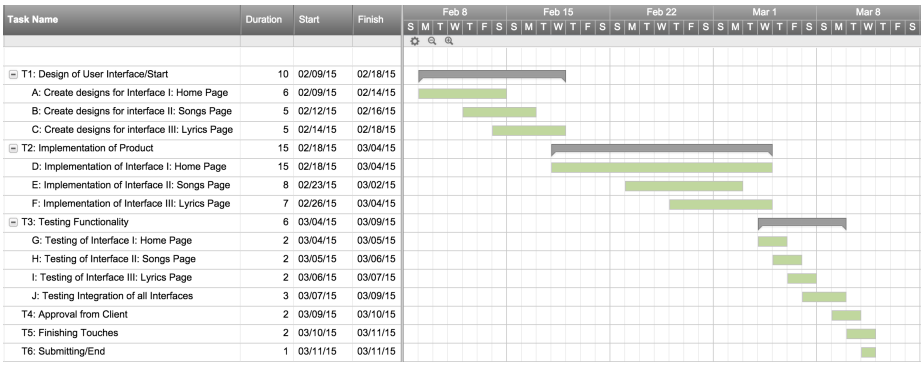

Figure 2: Schedule

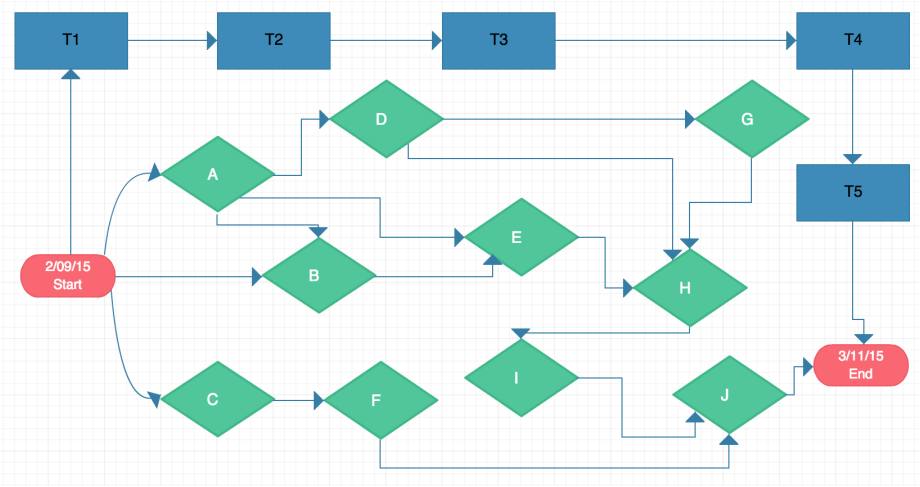

Figure 3: Activity Network

4.2 Project Milestones and Deliverables

The process model for our project is the Waterfall Model, meaning we will finish each big software development phase before we start the next. We will finish requirements engineering before starting the project design, which will be finished before we start implementation, which will be finished before we start testing, which will be finished before we arrive at the final phase of project, maintenance.

Our project initiation was being assigned this project by Dr. Halfond. The first step in this project was beginning to work on the Software Requirements Specification. The subsequent milestones and deliverables are listed below.

Deliverables and Milestones:

- Project Management Plan 2/9/15
- There are no specific milestones for this deliverable as it will be worked on by all team members and completed in full by February 9th.
- Design. 2/18/15*
- Milestones A-C
  - A: Design of the Home Page
    - A.1: Search bar
    - A.2: WC generation
    - A.3: Share button
    - A.4: Add to Cloud
  - B: Design of the Songs Page
    - B.1: List of songs
    - B.2: Back to Home button
  - C: Design of the Lyrics Page
    - C.1: Lyrics displayed on page
    - C.2: Back to Songs button
    - C.3: Back to Home button
- Implementation. 3/4/15*
- Milestones D-F
  - D: Implementation of the Home Page
    - D.1: Search bar with autocomplete functionality when typing in an artist's name
    - D.2: WC generation with words that can be selected to take the user to the Songs Page
    - D.3: Share button to upload the WC to Facebook
    - D.4: Add to Cloud button to create a new WC based off of words commonly used by both of the specified artists

  - E: Implementation of the Songs Page
    - E.1: List of songs sorted by how frequently the selected word is used in each song
    - E.2: Song titles in list able to be selected, taking the user to the Lyrics page
    - E.3: Back to Home button takes the user back to the Home Page with the WC still displayed and the artist's name still in the Search Bar
  - F: Implementation of the Lyrics Page
    - F.1: Lyrics displayed on page with the selected word highlighted every time it appears in the song
    - F.2: Back to Songs button takes the user back to the Songs Page with the same list of songs still displayed in the same order
    - F.3: Back to Home button takes the user back to the Home Page with the WC still displayed and the artist's name still in the Search Bar

- Testing and Final Delivery. 3/11/15*
- Milestones G-J
  - G: Testing of the Home Page
    - G.1: Search bar with autocomplete functionality when typing in an artist's name
    - G.2: WC generation with words that can be selected to take the user to the Songs Page
    - G.3: Share button to upload the WC to Facebook
    - G.4: Add to Cloud button to create a new WC based off of words commonly used by both of the specified artists
  - H: Testing of the Songs Page
    - H.1: List of songs sorted by how frequently the selected word is used in each song
    - H.2: Song titles in list able to be selected, taking the user to the Lyrics page
    - H.3: Back to Home button takes the user back to the Home Page with the WC still displayed and the artist's name still in the Search Bar
  - I: Testing of the Lyrics Page
    - I.1: Lyrics displayed on page with the selected word highlighted every time it appears in the song
    - I.2: Back to Songs button takes the user back to the Songs Page with the same list of songs still displayed in the same order
    - I.3: Back to Home button takes the user back to the Home Page with the WC still displayed and the artist's name still in the Search Bar

- J: Testing of the entire product to ensure all pages work together as specified in the SRS

Deliverables with an asterisk (*) by the due date indicate that there are risk factors that may alter the completion date. These risk factors include, but are not limited to, changing requirements for each deliverable and the deliverables taking either more or less time than expected. Due dates for each milestone are listed in the schedule in section 4.1. Also note that these dates are subject to change by the client and are only tentative (dates are from project schedule given by client, where it is noted that they are subject to change)

The termination activity of our project is presenting the completed product to both Dr. Halfond and Ms. Mahajan and ensuring that both of them are satisfied with our work. However, if this is not accomplished by the final project deadline, 3/11/15, then our project will be terminated regardless of the customer's satisfaction.

5. Configuration Management Plans

5.1 Product Documentation Management

Formal report formats between the client and development team will be structured according to IEEE standards and use LaTeX as a document preparation system for the client. The code documentation will be generated by Sphinx, which provides an easy way to create intelligent and beautiful documentation. In particular, it allows the documentation to be generated in several formats (eg, HTML, LaTeX, PDF) directly from the comments in the source code. Therefore, the documentation will partly be integrated in the code base and will be versioned by git as well. These processes will be used for the Software Requirements Specifications (SRS), Software Project Management Plan (SPMP), and a Design Document (DD).

5.2 Code Base Management

The whole code of the project will hosted and managed by the git program. Specifically, the project is going to be divided in three main repositories; the frontend, the backend and a third repository for general purpose. All three of the repositories will be hosted by GitHub.com and they will belong to the C-Lyrics organisation. This will allow the development team to better coordinate their effort while creating the software, and takes care of most of the merges in the code base. As stated in Section 5.3, the GitHub repositories will be coupled with Travis CI for continuous integration.

5.3 Software Quality Assurance Plan

Reviews and audits will take place after every milestone is completed. The development team will review the progress made thus far to make sure deadlines will be met, while observing and complying with the details of the project management plan. If there are noticeable shortcomings and deviations in the current progress of the project from what has been decided in the project management plan, audits to the project will have to be made.

In addition, the software will be developed according the Test Driven Development methodology. This aims to provide an additional measure for the advancement of the project as well as an assurance on its quality. By writing the basic tests first and then implementing the functions so that they pass the tests, the goal of the development process is slightly modified. Instead of aiming to implement every requirement and functionality, the objective is to have all the pre-written tests pass.To keep track of such advancements, GitHub will be coupled with Travis CI allowing continuous integration along the process. Finally some code coverage tools (ie, JSCoverage, PHP_Unit) will be used in order to keep track of the percentage of total coverage on the project.

In order to keep track of the progress, the development team will use a custom metric. Essentially, the metric mixes the objectivity of the test coverage as well as the personal insights of the team members. Each team member will give a grade from 1 to 5 to each one of the other members based on the amount of work they accomplished. Each member's score will be averaged and the average of these averages will be the final score of the team. This score is then combined with the amount of tests that pass as well as the amount of code coverage. Hopefully, this combination of both an objective metric as well as the team's self opinion on how they are doing provides an accurate estimation of the progress on the project. The mathematical details and some additional reasons for the use of this metric are included in the Appendix.

5.4 Project Monitoring Plan

Github's issue tracker and progress reporting system, along with Google Docs, will be used as the primary project monitoring mechanism among team members. Tasks will be assigned to each team member with a description and a project milestone. Each issue completion will correspond to a milestone set in the project management plan. If there is a change in the milestone contents and dates, the issue tracker will be updated in accordance and strictly follow the most up to date plan. Once the task is completed, the team member will submit the deliverables to Github and mark the task in the issue tracker as completed.

6. Risk Management Plan

6.1 Risk Identification

Risk Type	Possible Risks
Technology	The maximum number of queries allowed on EchoNest is reached during testing. The server that is hosting the domain goes down. The website traffic is overloaded, causing a crash. The real-time performance of the software is inadequate.
People	Staff members do not have the required technical skills demanded by the project and must spend time to learn them. One or more members of the staff are seriously ill or incapable of working. Staff members who lack experience are unable to learn quick enough to meet deadlines.
Organisational	One or more new staff members are added to the development team. One or more of the current staff members quits. A flat organization produces a lack of accountability and confusion, resulting in delayed deliverables. The waterfall method is ineffective due to changing customer demands and uncertainties.
Metrics	The metrics are not completely objective
Tools	The chosen CASE tools cannot be integrated into the software without serious restructuring.

Requirements	The client and stakeholder want to alter small details of the product. The client and stakeholder want to restructure the product entirely. Requirements from the client are misinterpreted during implementation. The development team implements features that looks aesthetically pleasing, but is unwanted by the client.
Estimation	The software takes longer to develop than previously anticipated. The number of people using the software is greater than what was previously decided by the client and stakeholder.

6.2 Risk Analysis

No.	Risk	Probability	Effects
1	The maximum number of queries allowed on EchoNest is reached during testing.	Very Low	Catastrophic
2	The server that is hosting the domain crashes.	Low	Serious
3	The website traffic is overloaded, causing a crash.	Very Low	Serious
4	Staff members do not have the required technical skills demanded by the project and must spend time to learn them.	Moderate	Tolerable

5	One or more members of the staff are seriously ill or incapable of working.	Low	Serious
6	Staff members who lack experience are unable to learn quick enough to meet deadlines.	Low	Serious
7	One or more new staff members are added to the development team.	Low	Insignificant
8	One or more of the current staff members quits.	Low	Tolerable
9	The chosen CASE tools cannot be integrated into the software without serious restructuring.	Moderate	Tolerable
10	The client and stakeholder request to alter small details of the product.	High	Tolerable
11	The client and stakeholder want to restructure the product entirely.	Moderate	Catastrophic
12	The software takes longer to develop than previously anticipated.	High	Tolerable

13	The number of users is greater than what was previously decided by the client and stakeholder.	Moderate	Tolerable
14	A flat organization produces a lack of accountability and confusion, resulting in delayed deliverables.	Moderate	Serious
15	The waterfall method is ineffective due to changing customer demands and uncertainties.	Moderate	Tolerable
16	The real-time performance of the software is inadequate.	Moderate	Serious
17	Requirements from the client are misinterpreted during implementation.	High	Tolerable
18	The development team implements features that looks aesthetically pleasing, but is unwanted by the client.	Moderate	Tolerable
19	Objectibility of custom metric to track progress of project is low	Moderate	Tolerable

6.3 Risk Planning

Risk No.	Strategy Type	Solution
1	Avoidance	Make smarter test cases that cover more areas of vulnerability in a smaller amount of API queries.
2	Avoidance	Host the software on a reliable server, or perhaps change the hosting location based off of previous crashing problems..
3	Avoidance	Select a more robust hosting service.
4	Contingency Plan	Allocate staff members to develop parts of the project that best suits their skills. If no one is familiar with the language or practices required, work in teams of two to promote collaboration
5	Contingency Plan	Assign the tasks of the ill staff member to various other members, while the ill staff member can update the new members on the current state of the task.
6	Contingency Plan	Delay the deliverable output, or submit the incomplete deliverable and patch the mistakes later when given more time.

7	Minimisation	Update the new staff member with the current state of the software and introduce low level tasks to help get him or her up to speed.
8	Minimisation	Allocate the work of the quitting staff member among the remaining team members.
9	Contingency Plan	Find new tools to fill the needs of the software.
10	Contingency Plan	Update previous documents with the new information and fill out a revision history of each document. Change the code to fit the demands and, depending on the magnitude of the desired changes, restructuring of the software may be necessary.
11	Contingency Plan	Depending on the amount of change desired by the client, the staff may have to create new documents entirely and essentially start a new project.
12	Contingency Plan	Change the deliverable dates to compensate for a new estimated time using COCOMO.

13	Minimisation	To compensate for more users than previously intended, the software may have to be altered to more effectively handle storage, the EchoNest API licensing may have to be purchased to allow more requests, and the server will need to be able to handle more requests.
14	Minimisation	Hold team members more accountable for his or her own tasks. Part of this can be achieved by instilling a productive atmosphere during meetings and other official work times. If there continues to be problems with a flat organization, a hierarchy may need to be introduced to inspire more efficient work.
15	Contingency Plan	The development team will have to work around the inefficiencies of the waterfall method. Switching to Agile methodology is not an option.
16	Avoidance	The development team will need to structure the code in a way that produces results in the quickest way possible. If it is too slow, restructuring may be required. Caching results is one way to speed up results.

17	Avoidance	If the requirements specified by the client is different than what is implemented, the development team will have to fix the unwanted components. This can be avoided by following the requirements document closely, assuming the client specified everything during the formation of the requirements document.
18	Avoidance	The development team should follow the requirements document regardless of the opinions of the team on the specifications of components.

6.4 Risk Monitoring

Throughout the development cycle, the team will have to monitor each risk regularly and prepare for it depending on the estimated likelihood that it will occur. Each member will be aware of the risk pertaining to his or her tasks and discuss the chance that they occur at each meeting. Many risks do not involve the actual implementation and cannot be anticipated, therefore these risks can be ignored during team meetings. Larger risks for a particular milestone or deliverable can be noted in the github issue tracker along with the regular team meeting procedure.

7. Appendices

A.1: Appendix 1: Meetings with Customer

Change to Requirements

- On February 5, 2015, the client indicated a change in the text color preferences for the WC. Before, the client required the text color of the

WC to be black. Now, there should be different, randomly assigned colors for each word.

Added Details

- Search bar should be user friendly, user should be able to leave out common words and still get an accurate suggestion.
- Search bar should have some sort of direction for the user as to how he/she should proceed

A.2: Appendix 2: Metric and Measurement

Our main issue in this document was to come up with a relatively objective and reliable metric. As we decided to extensively use code coverage it felt natural to use this aspect in our final metric. However, as for any metric, code coverage can be tricked and might not reflect the actual progress of the project. Therefore we decided to add a human insight and average it several times in order to obtain a relatively objective metric. Finally, we combine both of those metrics together to obtain the current state of progress on the project. We then evaluate this state against a pre-defined threshold, which should tell us whether we are on time in regard to our schedule.

There are several reasons why we include a seemingly subjective aspect in our measurement process. First of all, we consider that this measurement is made less subjective than it might seem, as we average values that we assume come from a gaussian distribution. (A grader should choose the more nominal value if hesitating, according to COCOMO and the lecture slides) Therefore, we argue that the Central Limit Theorem applies, and that the distribution of the groupe grade should also be gaussian. In addition, as we perform the grading process within the same population and with the exact same criterias, a comparison between two final scores should be meaningful. Finally, human beings seem more difficult to trick than a numbered test and the combination of both seemed optimal.

Computation

Each of the team member will frist grade each one of the other team members. This grade is the personal estimate of the progress of a team member on his tasks. We consider that each team member furnishes the same amount of work. For example, one team member could think that another is slightly ahead of time, at the current time and for the task he was assigned to. In such a case, the grade will be 4.

$$\forall i, j \in members, i \neq j :$$

$$g_{i,j} = \begin{cases} 1 & Very\ Late \\ 2 & Sligthly\ Late \\ 3 & On\ Time \\ 4 & Ahead\ of\ Time \\ 5 & Very\ Ahead\ of\ Time \end{cases}$$

We then average the grades together for each of the team members. Then, we take all of those averages for each person and average them together, which gives us the group score. Finally, we normalize this number to obtain a score percentage. Note that this percentage can be above 100%, in the case where team members are ahead of the schedule.

$$\forall i \in members : g_{i,avg} = \frac{\sum_{j \in members, i \neq j} g_{i,j}}{|members| - 1}$$

$$score_{grade} = \frac{\sum_{i \in members} g_{i,avg}}{|members|}$$

$$score_{team} = 100 \cdot \frac{score_{grade}}{3}$$

We then proceed to obtaining the same kind of score for the code coverage. This score is influenced by the percentage of current coverage and the percentage of passing tests.

$$score_{coverage} = coverage_{perc} \cdot passing_{perc}$$

Finally, we average both score together. Here some variations might apply, depending on the development team members, and we could decide to weight one aspect more than the other. And if this final scalar is above our team threshold, then we know we are on time otherwise we are not.

$$score = \frac{w_{coverage} \cdot score_{coverage} + w_{team} \cdot score_{team}}{2}$$

$$measurment = \begin{cases} On\ Time & score \geq threshold_{team} \\ Late & Otherwise \end{cases}$$

Note that this score is far from being perfect, as some strange cases might appear. Consider for example that the code is fully covered by tests and that all of them are passing, but that the team members are not happy with each other. Then the final score is supposed to be perfect, however it is not. We argue that such a situation should not appear, unless the code coverage is not ahead of the implementation, which contradicts the method we will want to use.

In the opposite case where each members are extremely happy with each other's progress, but tests are not passing, it is also possible to have a perfect score. Again, this situation should never happen but if it does, then the measurement can be improved by modifying the team threshold and weighting of both criterias.

A.3: Appendix 3: COCOMO Details

We used the COCOMO II model for estimating the work required to complete our project because it is an effective standard formula that takes initial assumptions and makes a useful projection from them. Although the cost drivers operate based on assumed levels, we feel that the calculation takes several important factors into account, making up for any subjectivity. The individual assumptions for why we chose certain levels for each driver is listed in section 3.3. We also made an assumption for the KSLOC, estimating that our project will reach about 1,000 source lines of code. We are assuming that the project will be rather small compared to industry software projects since the website does not need to be excessively robust in design and functionality.

Formula

$$E(person - months) = b \cdot KLOC^c$$

(our estimation for KLOC approx = 1)

A.4: Appendix 4: Resumes

Justine Cocchi
729 W 28th St. Los Angeles, CA 90007
jcocchi@usc.edu (562) 810-0944

Education

B.S. in Computer Science and Business Administration
University of Southern California, May '16 GPA: 3.36
Viterbi Study Abroad Program in London (June – July '13)

- Was the only Freshman student admitted to Viterbi's most competitive summer program
- Excelled in two upper division courses, writing for engineers and economics of engineering

Skills

Applications: MS Word, Excel, PowerPoint, Eclipse, VirtualBox, Rational Application Developer
Programming Languages: Java including JUnit testing, C/C++, Python
Platforms: Mac OS X, Windows, Linux

Projects

Sim City (Group Project)

- Worked as team lead on a multithreaded program in Java utilizing agent methodology to simulate a city complete with houses, apartments, banks, markets, and five distinct restaurants. Users can add citizens and pick a job for them, residents can drive a car to work, take a city bus, or walk. A* algorithm implemented to determine walking path of citizens.

Side-Scroller Game

- Imagined and realized a side-scroller game called "Road Kill" using C++ and QT to support graphics simulating a user driving a car. Squirrels and opossums walk across the screen giving the driver chances to hit them and earn points. Driver must avoid trees that popup and pedestrians that walk across the screen at random. Implemented AI to add difficulty.

Social Network

- Designed a command line based social network using C++ with the capability to add users, create and remove friend connections between users, and assess the betweenness centrality score of any given user based on analyzing the edges of the entire social graph.

Introduction to Digital Logic

- Learned basics of digital logic: Boolean algebra, Boolean functions, binary arithmetic, combinational logic devices, sequential circuits, state machine design and implementation.

Professional Involvement

IBM Internship *May – August '14*

- Tested the IBM Security Privileged Identity Manager product and became familiar with practical applications of databases as well as testing programs in a corporate environment

VP Student Affairs, Society of Women Engineers (SWE) *August '14 – May '15*

- Organized events to foster academic, professional, and social member development

VP Operations, Alpha Gamma Delta *Nov – Dec '14*

- Tracked chapter wide attendance to ensure that girls fulfill their obligation to AGD, assigned fines to girls as needed, took minutes at all meetings at the chapter, officer, and executive levels, completed and updated chapter paperwork as needed

Awards and Achievements

Girl Scout Gold Award (Boy Scout Eagle Award equivalent)

KELSEY FARGAS

3630 Lemon Avenue, Long Beach, CA, 90807
1(562) 338 4511 kelsey.fargas@gmail.com

EDUCATION

University of Southern California, LA, California: Applied and Computational Mathematics B.S., Minor in Computer Science **(2013-2016);** GPA: 2.66
Marymount College, Palos Verdes, California: (2012-2013); GPA: 3.89 **Combined GPA: 3.16**

RELEVANT COURSEWORK

Discrete Methods in CS	Data Structures and Object-Oriented Design	Principles of Software Development
Leading Organizations	Marketing Fundamentals	Microeconomics for Business

EXPERIENCE

Undergraduate & Student Affairs, ***Admission Intern Manager, (USC Viterbi, Los Angeles, August 2013 Present)***

- Administer office intern responsibilities, including training and logistical planning for office staff
- Synchronization of tasks and emails within Microsoft Outlook, Access, and SIS inquiries based off of prospective and current student database
- Assist with admitted student events and admission directives

Trendii.com, ***Social Media Intern, (Los Angeles, July 2014-Present)***

- Maintain social media sites (Facebook, Twitter, LinkedIn, & Google+) through Hootsuite
- Collaborate worldwide for company growth and evolvement through projects and employ marketing strategies

CSavvy.org, ***Co-Founder, (Los Angeles, May 2014-Present)***

- Lead workshops throughout LA District for young girls to demonstrate computer science concepts such as programming and soft and paper circuits
- Direct, edit, and produce videos for workshop promotions
- Design website through WordPress and manage social media sites (Facebook, Instagram, and Twitter)

ACADEMIC INVOLVEMENT

Society of Women Engineers, ***Membership Chair and Webmaster, (USC, 2014-Present)***

- *Membership Chair of the Operations Committee*: tracking and sustaining membership with new and current members
- *Webmaster of the Operations Committee*: running the official USC chapter website, maintaining social media sites, and photograph events held by SWE
- *General duties:* organizing, planning, and collaborating with other committees on yearly events

Troy Philippines, ***Philippine American Culture Night: Lead 2014, Skit Director 2015, (USC, 2014- Present)***

- *PACN Leading Actress:* Prepared cast for cultural show case and starred as leading actress show
- *PACN Skit Director:* Facilitate duties amongst back stage crew members and direct production of showcase

RELEVANT PROJECTS

Social Network (C++) (individual)

- Developed social networking program that lets a single user create a network of friends.
- Implemented breadth first search with integrated programming concepts such as data structures, access methods, and functional decomposition.

Search Engine (C++ & QT)(2 person team)

- Created a search engine with a user interface using QT by applying the PageRank Algorithm and WebCrawler.

Instant Messaging (Java & MySQL)(4 persons team)

- Collaborated to design and create an interactive application that lets users communicate over wireless networks through a graphical user interface.
- Implemented multi-threaded programming and networking.

KPMG Marketing Plan (5 persons team)

- Analyzed various data to create a better marketing mix, channel and service distribution, and promotion effectiveness
- Developed a marketing plan which consisted of optimal marketing strategies to reach both targeted segments (college students and potential clients)

SKILLS & INTERESTS

Proficient	Microsoft Office Suite 2013/2014, Photoshop, Video Editing (iMovie)
Knowledgeable	Java, C/C++, Python, MySQL, JSON, XML, HTML5/CSS, ANGULAR JS
Interests	Triathlon, Swimming, Hack-a-thon, Music production, Rock Climbing

Mark Krant

Student at University of Southern California

Experience

Software Developer at Southern California Earthquake Center (SCEC)
June 2014 - Present (9 months)

Software Developer at Local Corporation
June 2013 - August 2013 (3 months)

Languages

Spanish

Skills & Expertise

Software Development
Programming
Teamwork
Communication Skills
Learning Quickly
Learning New Software
Computers
Computer Science

Education

University of Southern California
Bachelor of Science (BS), Computer Science, 2013 - 2016
Activities and Societies: Kappa Sigma, Undergraduate Research

Page1

Milad Gueramian
1844 Thayer Avenue #301, Los Angeles CA, 90025
310-717-8283
gueramia@usc.edu

Education

University of Southern California, Los Angeles, CA — May 2016
Bachelor of Science in Computer Engineering and Computer Science

Technical skills

Programming Languages: Python, C++, Java(in progress)
Operating Systems: Linux, Windows

Spoken Languages Persian (fluent), Portuguese (proficient), French (intermediate), German (beginner)

Projects

Web Scraper from Online Directory (Current) — Summer 2014-Present
- Utilized regular expressions and python http requests and cookie handling libraries to scrape student email addresses from school's online student directory.
- Analyzed system with Google Chrome Developer Tools

Modelling Twitter — Summer 2014
- Created custom C++ templated classes and built functionality to identify follower and following relationship
- Build GUI with QT graphics library to create a simple version of the Twitter application.

Circuit Design Software — Spring 2014
- Implemented Verilog file parser to create gate networks and connections and used A* algorithm to correctly get output bits from a file of input bits

UC Irvine Cubesat — Sep 2012-Apr 2013
Team Leader, Thermal Radiator (research)
- Secured $550 in grants by writing a proposal to the University Research Opportunities Program.
- Networked within campus to fabricate custom pieces like jack screws and 3-D printed material and to borrow equipment from other departments.

Work Experience

USC IT Services: Learning Environments |Los Angeles, CA — Oct 2013-Mar2014
Student Assistant
- Oversaw classroom operations using remote monitoring systems
- Advised students and Professors on how to use University web services and classroom/lecture hall hardware.
- Communicate with campus police to gain/give building access

USC Roski School of Fine Arts IT Services | Los Angeles, CA — Oct 2014-present
- Set up hardware such as printers and projectors for various classes and events

Self-representing Soccer Agent |Los Angeles, CA — Sep 2012-Aug 2013
Manager
- Built and maintained a large network in the German Bundesliga including the German Football Association
- Secured several try-out invitations at high profile German clubs
- Played on trial at Brazilian division 1 club Avai S.C.

Digicarpet.com: Online movie distribution platform — Jun 2013-May 2014
Chief Strategy Officer
- Completed limited market analysis to determine feasibility
- Wrote business plan and developed strategy for approaching and negotiating with investors

JEFFREY KANG

18161 Ivorycrest Lane, Huntington Beach, CA 92648 • kangjr@usc.edu • (714)600-4291

EDUCATION

University of Southern California — Expected Graduation: May 2016

- Major: Computer Science/Business Administration (Viterbi School of Engineering)
- GPA: 3.6
- SAT: 2350 (Verbal: 770, Writing: 780, Math: 800)

EXPERIENCE

Work Experience:

Force Impact Technologies — September 2014 – Present
Team Leader

- Managing and supporting a crowdfunding campaign to raise research capital
- Developing the introductory product: The FITGuard (www.fitguard.me)
- Advising in business decisions for a growing startup company
- Representing the company at startup competitions and shows such as Consumer Electronics Show

Brandboom — September 2012 – July 2013
Marketing/User Interface Intern

- Researching trade shows to find potential fashion designer clients
- Testing software for bugs, aiding in website design, and writing customer support articles
- Capturing sample pictures for client catalogs

Leadership, Volunteering:

International Business Fraternity of Delta Sigma Pi — February 2013 – Present
Pledge Class President, Professional Activities Committee Member, Chancellor

- Leading a pledge class through the pledging program
- Contacting businesses/organizations to plan networking events and case study workshops
- Presiding over regular business meetings

Learning About International Commerce: Buenos Aires, Argentina — October 2012 – May 2013

- Developing global business understanding in a classroom environment
- Traveling to Buenos Aires to tour local businesses and observe global business operations

USC Kicks for Kids — September 2014 – Present

- Working with special needs children while teaching soccer skills

Laguna Fencing Center — May 2011 – August 2012
Assistant Coach, Team Captain

- Leading a team in national fencing competitions
- Instructing fencing classes, creating exercises to develop student fencing skills

AWARDS AND ACHIEVEMENTS

- Cal State Fullerton Stock Market Competition 1st place
- International Baccalaureate Diploma
- AP National Scholar/Collegeboard Scholar of Distinction (10 AP Tests, scored 5 on all)
- Alpha Lambda Delta Honor Society Member
- USC Marshall School of Business Dean's List
- USC Viterbi School of Engineering Dean's List

SKILLS AND INTERESTS

Skills: Java and C++ programming, Microsoft Office
Languages: English, Proficient Spanish and Korean
Sabre Fencing: 7 years national competitive experience; National C09 Rating
Guitar: 8 years lesson experience; Classical, Spanish, Rock, and Jazz styles
Ballroom Dance: 2 semesters class experience; Member of Break on 2 Performance Salsa Dance Team

Sébastien Arnold

CONTACT
3760 S. Figueroa St, Apt 323
90007 Los Angeles
USA

E-mail: arnolds@usc.edu
Website: www.seba1511.com
Phone: +1 213 604 4496

EDUCATION

University Southern California — August 2014 — Present
BsC. Computer Science
Audition of graduate artificial intelligence class. Minor in Physics.

ETH Zürich — September 2011 — June 2014
BsC. Computer Science
German-based instruction, extended mathematical background, advanced class in parallel and distributed computing, and machine learning project with Yuxin Chen, PhD student of Prof. Dr. Andreas Krause.

Stanford University — June 2011 — August 2011
High School Summer College Students program
Introductory classes to computer science and decision making.

Gymnase Auguste Piccard — August 2008 — August 2011
High School/Matura degree
Certifications in French, English, German and Italian. Sport recognition for ski racing.

WORK EXPERIENCE

University of Southern California — January 2015 — Present
Research Assistant
Research in quantum computing and quantum machine learning within the team of Prof. Dr. Daniel Lidar, under the directives of Dr. Itay Hen.

Schneeberger — June 2014 — November 2014
Lead Developer
Development of a financial management tool for an international company with an ever evolving structure. Managed a small team of 4 engineers and decided of the architecture and technologies used for the application. Extensive documentation and presentation to the company.

ETH Zürich — June 2012 — Present
Research Assistant
Development of complex web applications and experiments with JavaScript (VisualScience, Meritocracy) and PHP. Project received appraisal from the European Commission and fundings for coming years.(www.inn.ac)

Tooski — January 2009 — Present
Founder
Creation and management of the most complete French-speaking skiing website, its ski-clubs mobile application and its magazine.(www.tooski.ch)