**SPMP for the *Awesome Alphabet* educational application**

Approvals: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**1. Introduction**

**1.1 Project overview**

This project has been organized to produce an educational application called "Awesome Alphabet." The application will be developed in several stages as the customer intends to specify the requirements in stages following the demonstration of each version. The application will be designed to help children learn the names and sounds of letters.

**1.2 Project deliverables**

The following are to be delivered at the times shown:

Documentation listed below: fourth week of class

Version 1: Feb 28th: First Iteration. Expected features: Alphabet page, individual letter pages, sounds and pictures for each letter.

Version 2: March 28th: Second Iteration. Expected features: Customization (add/remove words, add/remove pictures, add/remove sounds, add/remove themes)

Version 3: May 2nd: Third and final iteration. Features: bug fixes of any features included in the first two, and possibly more features: detailed reports of child's progress, multiple children supported, parental features password-protected.

The supporting documents are: SPMP (this document), SQAP, Project Proposal, Product Presentation, Source code, Compiled Java byte code.

Acronyms are defined in section 1.5 below.

**1.3 Evolution of the SPMP**

This document shall be maintained on a weekly basis by the project leader. It is subject to configuration management by means of the SCMP. It is the project leader's responsibility to submit this document as a CI, and to keep it up to date. This SPMP mainly follows the format of [IEEE 1058.1-1987](http://standards.ieee.org/findstds/standard/1058.1-1987.html).

**1.4 Reference materials**

[IEEE] The applicable IEEE standards are published in "IEEE Standards Collection," 1997 edition.

[MPACL5] This document is to conform to the company's "Master Plan for the Attainment of CMM Level Five"

[Braude] The principal source of textbook reference material is "Software Engineering: an Object-Oriented Perspective" by E. Braude (Wiley, 2000). This document is based on the template used for the *Encounter* project in that book.

**1.5 Acronyms**

IEEE = Institute of Electrical and Electronic Engineers

QA = quality assurance

CI = Configuration Item

SCMP = Software Configuration Management Plan

SPMP = Software Project Management Plan (this document)

SQAP = Software Quality Assurance Plan

SDD = Software Design Document

STP = Software Test Plan

UI = User Interface

GUI = Graphical User Interface

TBD = to be decided

**2. Project Organization**

**2.1 Process model**

The first two versions of this project will be executed using an Agile Development process with an iteration corresponding to one release. The iterations are to be grouped according to the classification used in the USDP. ([Jacobson](http://ivar/002-3107326-9286063), [Rumbaugh](http://james/002-3107326-9286063), [Booch](http://grady/002-3107326-9286063) "The Unified Software Development Process" Addison-Wesley 1999). The USDP groups iterations into Inception, Elaboration, Construction, and Transition iterations. The first iteration will be considered the only Inception iteration. The second iteration is the first of the Elaboration iterations.

**2.2 Organizational structure**

The project will be organized as a team of peers with designated roles. The roles are Project Leader, Backup Project Leader, Configuration Leader, Environment Leader, Integration Leader, Requirements Leader, Design Leader, Implementation Leader, Testing Leader, and Documentation Leader.

The project will employ inspections as described by the SQAP. As this is a small team with a small project, all people on the project will be part of every inspection.

**2.3 Organizational boundaries and interfaces**

The project team shall interface with the course instructor / customer on a weekly basis. We will go over our weekly meetings, discuss requirements, and demo the latest version of the educational application.

**2.4 Project responsibilities**

The responsibilities of the participants in the project are shown in the following table:

|  |  |  |
| --- | --- | --- |
| **Role** | **Tools** | **Name** |
| Project Leader | - | Mark Musante |
| Backup Project Leader |  | Michael Grant |
| Configuration Leader | GitHub | Michael Grant |
| Environment and Integration | Ant | Levi Paul |
| Requirements Leader | Pivotal Tracker | Vivek Goyal |
| Design Leader | UML | Jaleel Kazi |
| Implementation Leader | Java, Refactor | Jaleel Kazi |
| Testing Leader | JUnit | Levi Paul |
| Documentation | Javadoc | Mark Musante |

Being responsible for a document includes the following.

* Making sure that the document is created on time
* Having the team leader identify the writers of the document.
* Keeping the document up-to-date throughout the project life cycle

**3. Managerial Process**

**3.1 Management objectives and priorities**

The highest management priority shall to satisfy the mandatory requirements. The second priority is that the product be on schedule. The third priority shall be satisfying as many requirements as possible. A desirable educational application is expected only for version 3.

**3.2 Assumptions, dependencies and constraints**

The largest constraint is our schedule. We must have everything done by the third iteration on 2nd of May.

We are assuming that we will not be getting many (if any) changes in requirements requested by the customer.

**3.3 Risk management**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Risk Title | Likelihood to occur (1-10) L | Impact (1-10) I | Retirement Cost  (1-10) R | Priority  (11-L)\*  (11-I)\*R | Retirement or mitigation Plan | Responsible Engineer | Target Completion Date |
| 1 | Database | 3 | 8 | 2 | (11-3)\*  (11-8)\*  2 = 48 | Learn Java DB API | MM | Feb. 21 |

Risk 1: Database. We have a couple of people on the team who are familiar with Java databases, but no recent experience on them. It will take time to come back up to speed with this capability.

Additional risks:

* Inexperience with Java programming language and/or tools – This will be mitigated by providing learning time and, when possible, designating specific group members to manage and provide training for these tools.
* Falling behind schedule – This will be mitigated by reducing the number of mandatory features and adding additional features on a prioritized incremental basis.
* Communication Breakdown – This would likely be due to the limited number of hours of face-to-face meeting time and can be mitigated through the use of Email, Skype, and Google Drive.

**3.4 Monitoring and controlling mechanism**

The entire team will meet on a weekly basis immediately before class, from 5:15pm to 6pm. We will also meet during/after class from 8:20pm-9:30pm.

If needed, we will meet once on another day of the week over Skype, the time to be prearranged during the after-class meeting.

**3.5 Staffing plan**

See chart in section 2.4.

**4. Technical process**

**4.1 Methods, tools and techniques**

This project will use the following tools:

* Eclipse - For developing the Java source code.
* PivotalTracker (<https://www.pivotaltracker.com/>) – A tool for creating user stories, managing workloads, and tracking completion during development iterations. Our public project is located at <https://www.pivotaltracker.com/projects/737427/stories>.
* Git – A distributed version control system for managing source code.
* GitHub (<http://github.com>) - A website that provides free public hosting of source code and documents. It supports Git and contains issue tracking / bug tracking tools. Our project repositories are located at <http://github.com/cs673>.
* EGit – A plugin that will allow us to use Git and GitHub directly from the Eclipse IDE.
* Google Drive (https://drive.google.com/a/bu.edu/) – A website that will host several project documents.
* Ant – Build automation tools.
* CodePro AnalytiX – Analyzes the Java source code to find bugs, unused items, etc.
* UML – A model language that describes objects in a software system and their relationships.
* JUnit – A unit testing framework for Java.
* Log4J – A logging library for Java.
* MyBatis – A data mapping framework for Java that couples objects to database stored procedures and SQL statements.
* JavaDoc – A utility for generated API documentation based on tagged comments in the source code.
* PDFDoclet - A tool for converting the generated javadoc into a single PDF.
* Skype – For conference calls / group meetings.

**4.2 Software documentation**

See the SQAP section 4.

**4.3 Project support functions**

No technical support specialist will be used for this project.

**5. Work packages, schedule and budget**

**5.1 Work packages**

The work breakdown structure is shown the following chart:

(This will be added once we start moving into the design phase of the project).

**5.2 Dependencies**

Iteration three is dependent on the completion of iteration two which is in turn dependent on the completion of iteration one.

We are dependent on the lectures of the course: each phase of the project is dependent on the topics covered in those lectures.

**5.3 Resource requirements**

The project will require five engineers. The hardware resources are going to be supplied by the individual contributors. All documents will be stored in Google Drive and GitHub.

**5.4 Budget and resource allocations**

There is no budget.

**5.5 Schedule**

Feb 7th: Initial Documentation

Feb 28th: Iteration 1

Mar 28th: Iteration 2

May 2nd: Iteration 3

**6. Additional components**

**6.1 Index**

TBD

**6.2 Appendices**

TBD