Singularity Software
Milestone 3

October 18, 2011

By signing below, I approve the contents of the following document.

Alex Mullans _____

Ruben Rodriguez _____

Ethan Veatch _____

Kurtis Zimmerman _____

Contents

1 Executive Summary

This document is the third in a series of milestone documents that will accompany the planning of the Siftables Emulator. The Emulator project is an application that will allow developers of Sifteo applications to test the features of the Cubes in a virtual programming environment. There is currently an emulator from Sifteo, Inc. that comes as part of the Software Development Kit (SDK) for the Cubes. However, Singularity intends to come up with a more natural interface than the one currently provided in that application.

This milestone enumerates all of the non-functional requirements of the the Siftables Emulator project and presents low-fidelity user interface mockups. Discussed first are the requirements for usability, performance, reliability, and supportability. Following that are considerations related to other software and hardware systems as well as documentation and legal requirements. Finally, mockups of the Emulator's design are presented for consideration and to solicit client feedback. Future milestones will present plans for change control, coding standardization, and testing. Finally, design and usability reports will make up the core of milestones near the end of the quarter as the software stabilizes.

2 Introduction

Developers of applications for the Sifteo Cubes currently must test programs they create for the platform within the emulator provided by Sifteo. While this emulator does cover all of the functionality of the Sifteo Cubes, it presents a user interface that Singularity Software believes could be more naturally implemented. As such, Singularity Software will provide, in the form of the Siftables Emulator, a new software-based emulator for the Sifteo Cubes that will allow developers to more naturally interact with the platform.

Milestone 3 elaborates the non-functional requirements of the system and presents low-fidelity mockups of the user interface design. It follows Milestone 2, which laid the foundation of the Siftables Emulator specification based on the high-level design created in Milestone 1. Milestone 4 will rely on these early milestones as it defines a change control plan and test cases, and Milestone 5 will elaborate the usability guidelines and interface design that implement the features and use cases described in Milestone 2 to the non-functional requirements described herein.

3 Project Background

The Siftables Emulator is being developed by Singularity Software as part of the Junior Project sequence of classes at Rose-Hulman Institute of Technology. When projects were solicited for the sequence, clients Tim Ekl and Eric Stokes (both Rose-Hulman alumni) submitted a request for an emulator for Sifteo Cubes, a new platform intended for "intelligent play." After Singularity was chosen for the project, we met with Mr. Ekl to determine the three primary features of the Emulator: a Workspace where 1-6 Cubes could mimic the manipulations possible with physical Cubes, an Application Programming Interface (API) to program those virtual Cubes, and a set of example games designed to show off the first two features. Singularity's Emulator will be the first program of its kind on the market for Sifteo Cubes.

4 Usability Requirements

The client would like the emulator to be accessible to new developers, but also efficient for experienced developers to use. The emulator may have a familiarization time in order for users to be able to use it productively. It will include a help menu containing instructions for the various features. In addition, the client would like keyboard shortcuts and accelerators available for users who prefer keyboard-based controls. As the user must be able to understand how to interact with the Cubes relatively quickly, the Cubes' control motions must be intuitive. The user must also be able to switch between motions quickly.

5 Performance Requirements

The performance requirements for the system are fairly basic. When the user interacts with the system there must be no visible graphics delay. Additionally, games simulated on the Cubes must function with performance similar to that of the physical Cubes. Beyond these two key requirements, actions must be performed in a timely manner for all scenarios that may arise during use of the emulator.

6 Reliability Requirements

As a system that emulates another system and runs user-created code, it is important that the Siftables Emulator does not contribute defects of its own to the application testing process. As such, Singularity Software will aim for a defect rate of 5 bugs/KLOC. As a predictor of this result, we will evaluate the cyclomatic complexity of our code at each development milestone, aiming for a score of 15 or less.[1] When evaluating the reliability of the emulator prior to release, bugs will be analyzed for priority and severity. For example, misspelled words or other aesthetic issues will not be treated with the same severity as crash-inducing bugs.

7 Supportability Requirements

Because all support for Siftables Emulator will be done by the client after the project has ended, it is imperative that the emulator have well-documented code that is easy for developers unfamiliar with the project to follow. It will also need to follow a recognized coding standard for the language chosen. As a result, it must be in an object-oriented language like Python or C# that is fairly well-known and frequently used.

8 Hardware and Software Interfaces

The Siftables Emulator will work as a standalone software package and will not interface with any other software or hardware. Singularity Software considered the integration of the emulator in an Integrated Development Environment (IDE) but decided that the emulator was best served as an independent tool to allow easy development across the supported platforms. Similarly, the purpose of the emulator is to serve as a method for

[1] Expected cyclomatic complexity scores are based on the blog post at: http://gdub.wordpress.com/2006/07/09/cyclomatic-complexity-for-python-code/.

developing applications for the Cubes without connecting to the physical Cubes, so it will not interface with any hardware either.

9 Documentation, Installation, Legal and Licensing Requirements

The clients have stated that the emulator project is to be licensed under the BSD 2-clause license[2]; therefore, many of the standard legal and licensing issues are eliminated. Mr. Ekl has requested that Singularity Software keep documentation of development actions according to project milestone standards as well as using error tracking and version tracking systems during the design process. He has also requested that the API developed for the project is documented in a well-accepted form like that produced by Javadoc.

The emulator will be installable via an executable (on Windows) or disk image (on Mac) that will place the program's files in a user-specified directory and create shortcuts as appropriate.

10 Design Constraints

The emulator must be able to be function on computers using the Windows, Mac OS, and Linux operating systems that satisfy the minimum system requirements of the previous OS version. In other words, the emulator should run smoothly on a computer that meets the requirements for Windows Vista, Mac OS X 10.6 (Snow Leopard), or Ubuntu 11.04[3].

11 User Interface Mockups

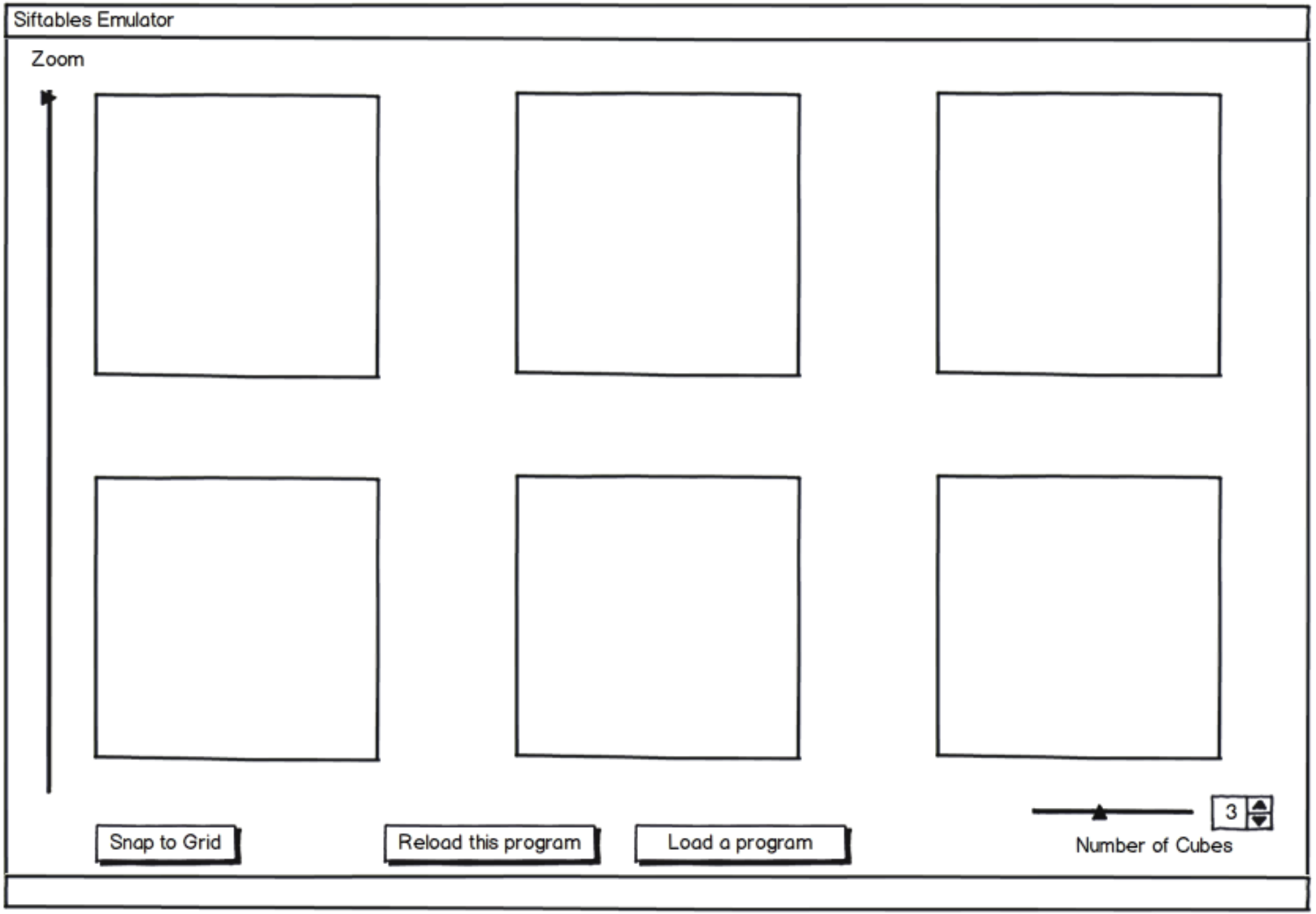
Preliminary mockups of the Siftables Emulator user interfaces are presented below. The mockups are rough sketches not intended to convey the product's final look. Rather, they present a simplified view of what Singularity thinks the emulator will look like when created in code.

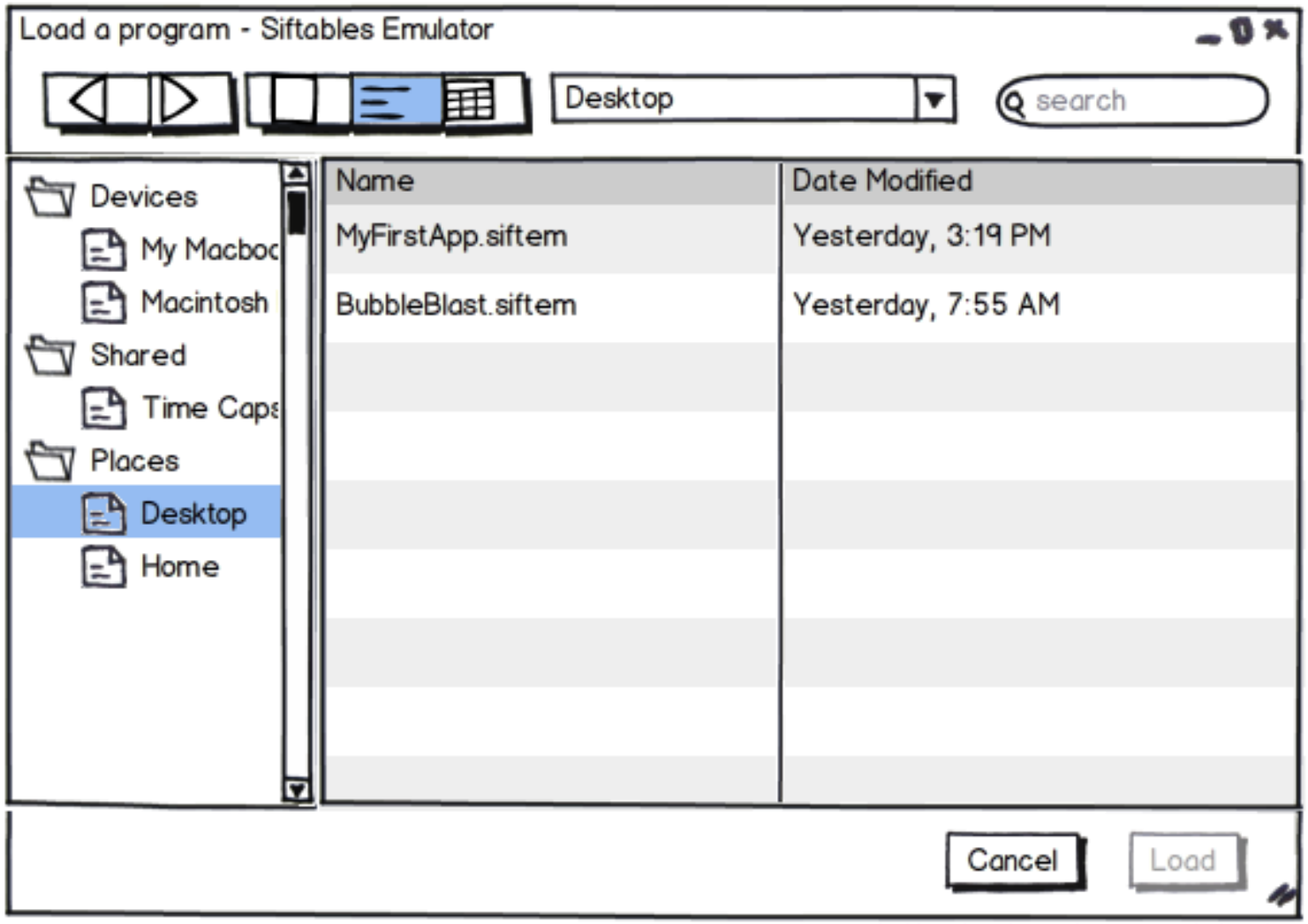
In order, the mockups display the main screen of the application, the "Load a program" dialog, and the various warning and alert dialogs that the program can display.

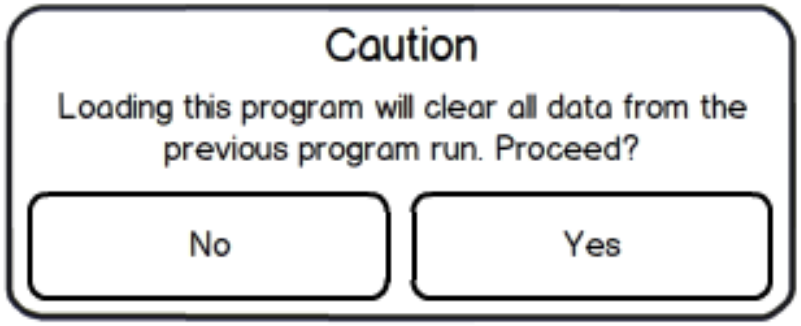
The Manipulate Cube use case (UC6) is not easily presented in mockup format. Essentially, it applies to any of the 6 Cubes in the first mockup; any of the actions described in the use case can be applied to each virtual Cube.

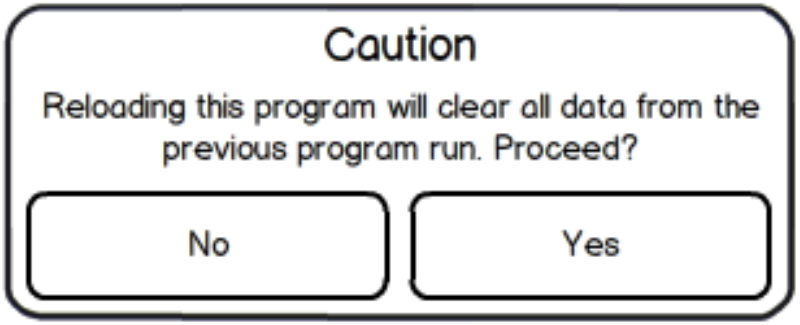
[2]http://www.opensource.org/licenses/bsd-license.php

[3]The Ubuntu machine should support graphics acceleration (i.e. it should be capable of running the GNOME 3 or Unity window managers).









A Features

Feature	Description	Status	Priority	Risk	Reason
Individual, virtual Sifteo Cube	A virtual representation of a single Sifteo cube	Approved	Critical	Low	Replicates physical Sifteo Cube
Buttons to manipulate each virtual Cube	Buttons on the virtual Cube will allow the user to flip and tilt it	Approved	Critical	Medium	Replaces physical actions where said actions would be impractical with a mouse
Workspace where multiple cubes can be emulated	Multiple cubes will be displayed on a workspace that replicates the free-form nature of physical Sifteo Cubes	Approved	Critical	Low	Replicates multiple Sifteo Cubes in a natural, free-form environment
Interactions between Cubes	The Cubes present on the workspace will communicate when they are neighbored	Approved	Critical	Low	Cubes can simulate the interactions possible with physical Cubes
Load programs into the Cubes	The user will load his own and example programs into the emulators Cubes	Approved	Critical	Medium	The ability to program programs for the emulator is dependent on a common interface
Snap Cubes to invisible grid	The Cubes will snap into an invisible grid when a button is clicked	Proposed	Useful	Medium	Increases productivity by allowing a quick reset if the Cubes are in disarray
Zoom Workspace	The Workspace will zoom to the level of an individual Cube or the whole space	Proposed	Useful	Low	Inspecting individual Cubes allows for precise checks of program Graphical User Interfaces (GUIs)

B Use cases and functional requirements

ID	Name	Description	Features
U1	Load program	The User selects the program file to be loaded and run by the emulator.	F5
U2	Reload program	The User reloads the program currently running in the emulator.	F5
U3	Zoom screen	The User zooms the Workspace to the desired level.	F7
U4	Add/remove Cubes	The User adjusts the number of Cubes present in the emulator.	F3, F4
U5	Snap Cubes to grid	The User pulls the Cubes into a grid orientation.	F6
U6	Manipulate Cube	The User manipulates a Cube by clicking the buttons or the Cube itself.	F1, F2, F3
OR1	API	An API is included to define a set of rules and specifications via which Cube programs can be created.	F2
OR2	Example games	Example games are included to demonstrate how the Cubes interact with each other.	F3

Glossary

Graphical User Interface is a visual way of allowing the user to interace with a computer program. 9

Linux is a Unix-based operating system based on free and open source software. 5

Mac is a series of lines of personal computers developed by Apple. 5

Sifteo Cubes are small machines capable of loading programs and interacting with one another as well as responding to predefined movements. 3

Software Development Kit is a collection of tools designed to help build software for a particular platform. It may include an and an emulator of the target platform among other things.. 3

Windows is a series of operating systems developed by Microsoft. 5

References

1. Sifteo Inc. Online: http://www.sifteo.com
2. Tim Ekl. Client Meeting. 4 10 October 2011 4:15 p.m.
3. Milestone 1. Singularity Software.
4. Milestone 2. Singularity Software.

Index