**Cupcake Express**

**System Design**

Adele Francois, Jawan Higgins, Lance Toornman

June 18, 2016

## **Section 1. Introduction**

The Software Design document is utilized to provide aid in the software development process by providing details on how to approach building specific software. The following deliverable contains narrative and graphical documentation through the use of hardware/software mapping, software architecture, and other supporting requirement information.

*1.1 Purpose of System*

The purpose of the Cupcake Express’ proposed project is to develop a fully functional cupcake vending machine application that meets the client’s need of distributing cupcakes easily and generating revenue effectively.

*1.2 Design Goals*

The following design goals represent the desired qualities of Cupcake Express and provide a consistent set of criteria that must be considered when making design decisions:

*Usability*

● Application should be accessible on all versions of vendor hardware.

● Application only supports one customer and Administrator at a time.

● Application should provide easy navigation with clear navigable naming conventions.

*Reliability*

● The system should support hundreds of users consecutively with continuous functionality without any freezes or delays.

● The system should withstand weather conditions such as rain, hail, snow, etc. by using thick layers of metal for the body and shatter proof glass for the screen.

● The system will always store a large quantity of cupcakes and almost never out of stock to ensure customers that our machines are very reliable.

● In the event of system failure, the system can only afford to fail a maximum of two times in a year.

● Shutdown occurrences or freeze ups should not occur for more than ten minutes.

*Performance*

● Application should be able to transition from one stage to another in no more than ten seconds.

● The system should be able to select and purchase cupcakes within a maximum time of 3 minutes for the average user.

● The system should process the input of cash and credit cards in less than 5 seconds.

● The system will transition to each screen in less than 10 seconds.

*Supportability*

● The system supports storage space for up to 18 different cupcake combinations.

*Implementation*

● The system will implement a database to store a variety of cupcakes, keeping track of all transactions, which will add and delete cupcakes in real time.

● The system will implement Administrative access to the back-end of the system for adding, deleting and updating the type of cupcakes.

*Interface*

● The system should have an option for the user to print out their receipt from their purchase.

*Packaging*

● The system should have a maximum of 10 cupcakes per selection.

● The system should deliver a package sealed cupcake in perfect condition as it was freshly made.

● The system should provide a shelf life for the cupcakes of 15 days.

*1.3 Definitions, Acronyms, and Abbreviations*

**Administrator**

Users that can adjust stock and product line.

**Cupcake**

A system object which represents half of the final product being purchased. Defined by a list of attributes and is selectable by customers, and manipulated by administrators.

**Cupcake Express**

The name of the company for which the product is being developed.

**Customer**

The users interacting with the application to make purchases.

**Icing**

A system object which represents half of the final product being purchased. Defined by a list of attributes and is selectable by customers, and manipulated by administrators.

*1.4 References*

The following deliverables/resources are referred upon in the development of the System Design deliverable:

* Problem Statement Deliverable
* Requirement Elicitation Deliverable
* Requirement Analysis Deliverable
* <https://wwwbruegge.in.tum.de/lehrstuhl_1/component/content/article/43-books/266-oose-bumpers-systemdesigndocument>
* <https://www.cise.ufl.edu/class/cen3031fa11/documents/examples/SDD_Example_1_2011.pdf>

*1.5 Overview*

The development of the Cupcake Express project is very extensive, but is projected to be an overall success with only a few more developmental steps left. The system design template, designed by the development team, will aid in delivering a complete application to the client on time. The initial idea of creating a cupcake vending machine derives from recognizing that there was no company in the bakery market that had a dessert vending machine and it seemed like an opportune time to create one. Cupcake Express plans to become a leader in the area of dessert vending by setting the standard of efficiency, creativity, reliability and affordability.

## **Section 2. Current Software Architecture**

There is no currently existing system that is being replaced. Similar systems, however, allow for users to make their selections but often only one at a time. Some do allow for multiple selections per transaction, both in terms of quantity and differing products. The concerns and issues that are typically addressed by such systems are the necessity of displaying a wide and variable selection of products and customizing orders to the customer’s taste.

## 

## **Section 3. Proposed Software Architecture**

*3.1 Overview*

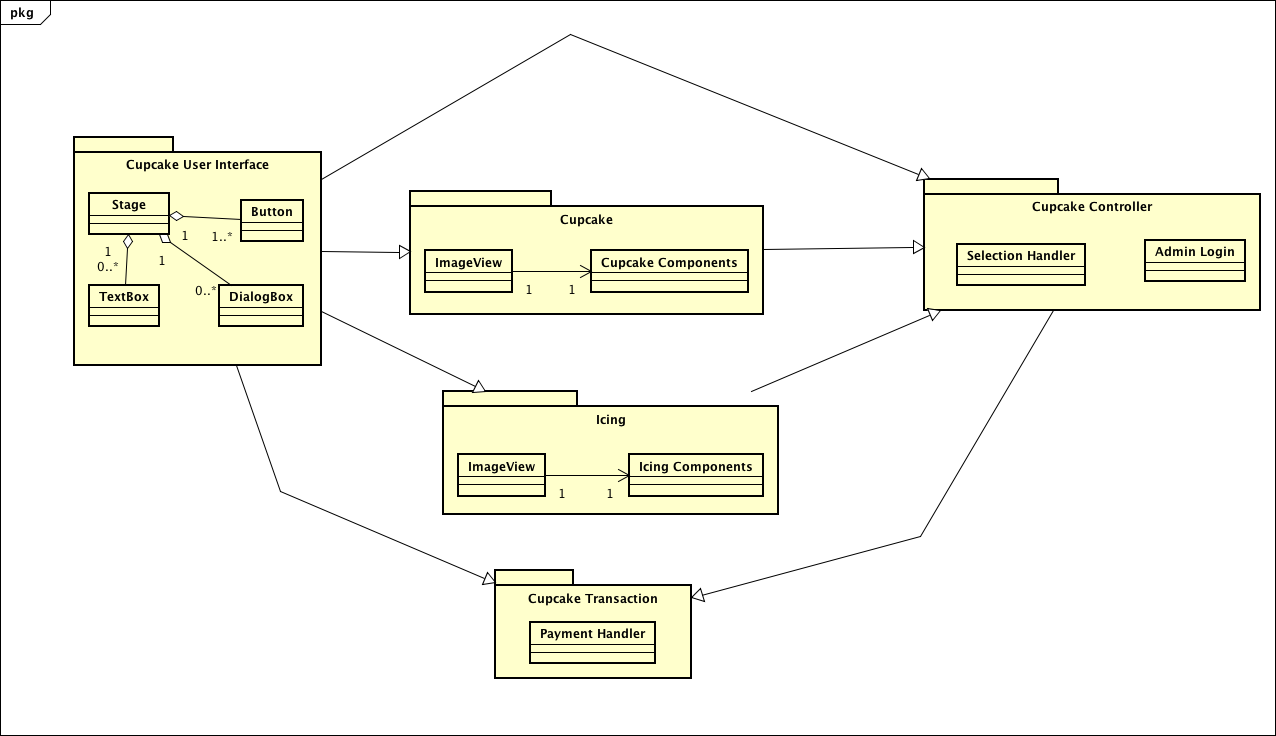
The Cupcake Express application design is based on the individual design of various components in which users will enter data and receive specific outputs. Our software architecture is designed to incorporate all data entries and modifications into the system in real-time as it’s manipulated. The two components which comprise the software architecture includes the client-side and server-side applications.

The client-side application is also separated into two parts:

* The functional components (written in Java) are identified as the user data entry modules which provide and receive user input in the user interface. These functional components are the main core of Cupcake Express and perform the majority of actions needed such as grouping products and performing transactions.
* The graphical components (written in both scene builder and FXML) simply compose the overall graphical user interface. It provides all of the buttons, text boxes, and other onscreen elements the application requires. These graphical components allow the user to access all of the application features available in Cupcake Express.

The server-side application involves managing all user input and output messages of the overall application.

*3.2 Subsystem Decomposition*



**Cupcake User Interface:** The UI system consists of classes that are responsible for the graphical user interface to the user. The UI includes a stage that contains each component, buttons that support the user’s selection or navigation, text boxes that are responsible for gathering input from the Administrator for login, and dialog boxes that show messages to the user.

**Cupcake**: Cupcake handles the physical and graphical implementation of the cupcake, consisting of an image and many components.

**Icing**: Icing handles the physical and graphical implementation of the icing, consisting of an image and many components.

**Cupcake Transaction**: Cupcake transaction handles the payment and receipt processing in the purchasing process.

**Cupcake Controller**: The cupcake controller handles the bulk of the behind-the-scenes application, including the cupcake and icing selections made by the user.

*3.3 Hardware/Software Mapping*

Cupcake Express vending is designed to incorporate both software and hardware. Software and hardware plays substantial roles in how the vending machine operates, mainly integrating the software to update and operate the hardware. Software maps to the hardware, managing and controlling the hardware by categorizing, storing, ejecting cupcakes, as needed. The hardware supports flexible functions for any case where the software may need to be re-implemented with changes. The software itself is easily reusable, due to not requiring drastic retooling before various deployments.

*3.4 Persistent Data Management*

Data is stored in temporary variables and removed once the program is exited. Cupcake Express has strong possibilities for implementing a database in the near future, but rather focus on creating a feasible system first.

*3.5 Access Control & Security*

* Security is controlled through a division of Administrator and user privileges, split through means of a login screen. The security issues that can arise are securing the management of authorized administrators and their login credentials.
* Security will be enforced to secure the readings of counterfeit cash or credit cards.
* Any incoming connections will be monitored through the network, and any tampering of the system will initiate an automatic lockout and will need to be inspected.

*3.6 Global Software Control*

The application developed involves only user interaction with the system. Global Software Control for this project is strictly event-driven.

*3.7 Boundary Conditions*

Start-Up

Software startup solely relies on the startup of the hosting hardware and system maintenance/updates as well. Booting the system takes no time at all, with the help of the framework organization within the code. The Administrator would restart the application after any major update or maintenance activity.

Error Behavior

The system is overall designed so the user cannot make any serious errors. In any case of an error, the system will respond by handling and directing the user to the best fit scenario based on the error. Navigating throughout the system is simplified, so errors are limited.

Shutdown

Shutting down the hardware that hosts the software application will automatically shut down the software application. Shut down depends entirely on both hardware and system maintenance/ updates. For updates and maintenance, the system shutdown will occur at the Administrator's request.

## **Section 4. Subsystem Services Glossary**

**Stage-** the stage provides a platform for the system’s application.

**Button-** an interactable component that performs an action upon being pressed.

**Textbox-** a box that allows user input or system information.

**Dialog box**- a box that appears on the display screen to present system information or request input.

**Selection handler**- function or method that handles the selection of any component.

**Transaction handler**- function or method containing program transaction statements that handle the computation various of monetary components.

**ImageView**- a representation of an image that allows interactive functionality.

**Icing Component**- a component that represents different icings available in the system.

**Cupcake Component**- a component that represents different cupcakes available in the system.