Software Design Document

Burris Logistics Virtual Warehouse Project

*Approvals:*

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Jordan Hinshaw | Date: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Team Member | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Matt Kent | Date: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Team Member | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Aaron Ramsey | Date | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

*Document History:*

10/13/09 – Initial document creationTable of Contents

[1 Project overview 4](#_Toc243202793)

[2 Design Introduction 6](#_Toc243202794)

[3 Design Viewpoints 6](#_Toc243202795)

[3.1 Data description 6](#_Toc243202798)

[3.1.1 Models 6](#_Toc243202799)

[3.1.2 User Profiles 6](#_Toc243202800)

[3.1.3 Elkton Warehouse World Data 6](#_Toc243202801)

[3.1.4 Tour Information 6](#_Toc243202802)

[3.2 ARCHITECTURAL DESIGN 6](#_Toc243202803)

[3.3 GUI Design 7](#_Toc243202804)

[3.3.1 Main Menu 7](#_Toc243202805)

[2.1.1 HUD 8](#_Toc243202806)

[3.4 Component Design 9](#_Toc243202807)

[3.4.1 Overview 9](#_Toc243202808)

[2.1.1 VirtualWarehouse Class 9](#_Toc243202809)

[2.1.2 Vocollect Connectivity 10](#_Toc243202810)

[2.1.3 Database Connectivity 10](#_Toc243202811)

[2.1.4 SimpleMonkey GUI Package 10](#_Toc243202812)

# 1 Project overview

* Context/Background

Burris Logistics is a frozen goods storage and shipping company. They operate several warehouses, mostly along the Eastern Seaboard, that are responsible for loading and unloading outgoing and incoming shipments, respectively, and storing the received products until such a time as they will be shipped elsewhere. Since Burris specializes in refrigerated and frozen goods, all of their warehouses are refrigerated. Each warehouse has several different temperature zones that must be maintained at a given temperature based on the requirements of the product stored in that section of the warehouse.

This project concerns the internal workings of the Burris warehouse in Elkton, Maryland. Most of the tasks in the warehouse can be grouped into two broad categories: inbound and outbound. All of the goods that are received from incoming shipments must be unloaded, processed, and then moved to the proper storage location. Conversely, whenever a shipment leaves from the warehouse, the correct products must be collected from their storage locations, prepared for shipping, and loaded onto the truck.

Warehouse workers are assigned to the various parts of each process, along with some other tasks that do not fit directly into either category. In particular, there are warehouse workers assigned to unloading incoming good from a truck and processing them. There are also workers assigned to taking these goods and placing them in the correct storage location. Furthermore, there are workers assigned to collecting goods that must be placed on an outbound truck and place these collected goods in the correct loading dock. There are workers assigned to the processing of collected goods, ensuring that the correct goods were collected; and there are workers assigned to preparing and loading the goods onto the truck. There are also workers that are assigned to replenishing the pick-up location of each product that is collected for outgoing shipments.

Initially, this project will mainly concern the job of the workers assigned to the collecting of goods to be placed on the loading dock for outgoing shipments. These workers are called “*pickers*” or “selectors” (see definitions). Burris uses a voice recognition system, called Vocollect, to direct the pickers through their assigned tasks. The Vocollect system gives commands and then waits for the worker to confirm the command and/or give it more information, as required for any given situation. The most common sequence is where the picker is (1) told to go to a certain location in the warehouse, designated by aisle and bin numbers. (2) Once the worker has reached the given location he must relay a series of number, called *check digits*, that are used to double check that the worker has, in fact, reached the correct bin.

(3) Once the picker relays the correct check digits (s)he is told the how much product to pick up. The worker picks up the given amount of products and places them on the pallet or pallets that the goods will eventually be shipped on. (4) The picker then confirms that (s)he picked up the given amount of product. The process then repeats as the worker is directed to a new bin location for the next selection. Steps (1)-(4) are repeated until the picker has completed the *pick job*, at which point the worker places some labels on the product and drops the pallet or pallets off at the loading dock. The worker then tells the Vocollect system that (s)he is ready to begin another pick job, at which point a new pick task is assigned.

The rate at which Burris pays each worker is based, in part, on his or her productivity. This is done using a set of Engineering Labor Standards which specify how much time an average worker takes to complete a given task. Worker pay is partially based on productivity relative to the ELS.

* Application (what will/should it do)

This project will produce a video-game-like application that can be used to help train warehouse workers, specifically the selector, on how to do their jobs. The application should accurately model the Elkton warehouse and relevant equipment used by selectors. The application should interface with the Vocollect technology so that the users can learn how to properly interact with the technology. The application should also be able to train the users in the proper use and operation of the needed equipment.

Initially, the application should simulate the daily tasks of a picker. However, the application should be developed in such a way that will provide the ability to generalize the application for any or all warehouse tasks.

* User(s)

The primary users will consist primarily of two parties: the warehouse trainers and the warehouse trainees. The trainers are well-versed in the inner workings of the warehouse and must become familiar with this application to understand its capabilities and limitations. The trainees may have some knowledge of various warehouse jobs based on previous employment, but will most likely have little to no experience with the Vocollect technology or Burris-specific procedures. Both parties may also have varying computer skills, according to their various backgrounds.

* Purpose (usefulness)

This application will help Burris train new employees for working in the warehouse. In particular, this application will help train those who have little to no experience with either Vocollect or the process of completing tasks in the warehouse.

# 2 Design Introduction

# 3 Design Viewpoints



## Data description

### Models

One of the largest sections of data in the application is the 3D models. All the objects in the warehouse, and the warehouse building itself, will be created in Blender. Some of the models will have texture files that go with them. These models can be exported as obj files from Blender (or md5 files if they are animations) to be imported into JME, or they can be converted, by one of JME’s utility classes, to jme files for faster loading into JME.

Models and their related information can be stored in the database.

### User Profiles

The user profile data will be stored in a database. The user’s name, location, and task will be stored.

### Elkton Warehouse World Data

A database is also needed to store the information about the world that represents the Elkton Warehouse. The database will need to store the placement of the models that are in the warehouse, including their location, rotation, and scale, and any other relevant information, such as if the object should have collision detection enabled.

### Tour Information

Another database will be required which stores descriptions that can be used for touring the warehouse and becoming familiar with its operations. There will be information bubbles in the warehouse environment which can be activated, and will appear as text on the screen, or possibly voice data.

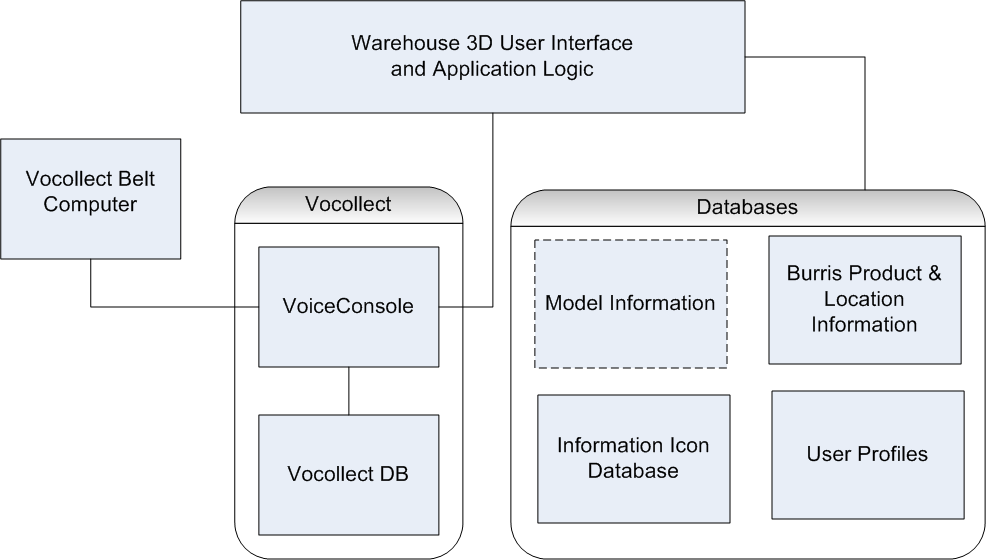
The database will need to store the location of the information icons in the world, the text that needs to be displayed, and possibly the type of data (audio or text).

## ARCHITECTURAL DESIGN

The Virtual Warehouse application will be contained in an executable file on the computer in which the trainee is going to be trained.

The application will communicate with a MySQL database to retrieve information about the models in the warehouse, the information bubbles that will be used for tours around the warehouse, the user profile information, and the product placement within the warehouse environment.

There will also need to be a server running the Vocollect software, called VoiceConsole, which also requires a MySQL database. Vocollect belt computer will communicate with VoiceConsole, which in turn will communicate with its database and then communicate back to the Vocollect belt computer and the warehouse application.



Figure

## GUI Design

### Main Menu

The Main Menu, FIGURE 2, will be displayed at game start up and will also be accessible during game play by pressing “Escape” (this is a common computer game hotkey).

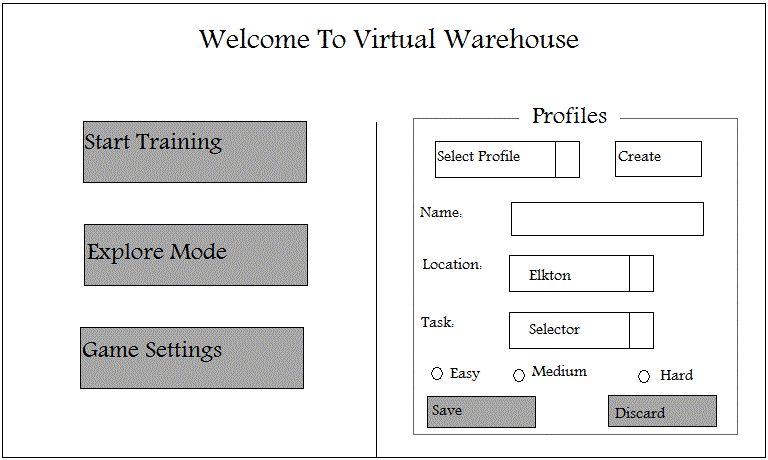


Figure : Proposed Main Menu

### HUD

FIGURE 3 conceptually represent the Head’s Up Display (HUD) interface that will be use in our application. It should be noted that this “screenshot” is notional and not an *exact* representation of how the application will look. The FIGURE 3 is intended to show that the 3D game world will be overlaid with a minimal amount of text and graphical information that will be useful in the game environment. This layout will be similar to the layout of many popular 3D first-person-shooter games.

The user interface will utilize a mini-map that will show the user where he/she is in the warehouse. This can be seen in the top left corner of the figure. Along the bottom edge of the screen there are several numbers and symbols. In our application this area will display useful information such as current amount of time elapsed, the users score, and perhaps the current game mode (such as “Free Exploration Mode” or “Training Mode”) and the current task assigned. In the top right corner of the figure, a message is shown. This area can be used to display messages that will help the user to find his/her way in the game environment. These messages could display anything from where the user should head next to which buttons the user should press for a given command.

Also in FIGURE 3, there are two “information bubble”. In the game environment these bubble represent areas of interest. The information accessed in these bubble can be accessed by walking up to them and pressing a given button (most likely the action button, “E”). The information will be displayed in textual format on the screen.



Figure : A Conceptual user interface for the 3d game environment

## Component Design

### Overview

At the component level, our application exists as a VirtualWarehouse class that extends the BaseGame class of the Java Monkey Engine. BaseGame provides the update and render loop that is responsible for the running the game. Some of the methods in BaseGame are overwritten by our VirtualWarehouse class to create the game. Running the VirtualWarehouse class will run the game until it is terminated.



Figure

### VirtualWarehouse Class

The VirtualWarehouse class is the main class of the application.

* initSystem method – This method is called once at the start of the application, and sets up the window and display, providing an OpenGL context for the application.
* initGame method – This method is run after initSystem and sets up the world, loading and placing all the models and loading the rest of the data for the game.
* update method – This method is called after initGame, and runs in a loop, alternating with the render method. Game logic is performed by this method, which includes player movement.
* render method – This method is called after the update method and handles the draw calls. It clears the screen and draws the updated scene graph.

### Vocollect Connectivity

Coordinates communication between the Vocollect system and the JME application.

### Database Connectivity

A MySQL JDBC driver is used for connecting to the database from our JME application.

### SimpleMonkey GUI Package

SimpleMonkey is a layout manager package that assists with the creation of GUIs. This is used for creating in-game menus and the HUD that appears during the application.

SimpleMonkey allows containers to be created, which can contain widgets, which are buttons and other containers. It provides a mouse listener class to listen to JME’s mouse events.