CGK - Community Gate Keypad

Software Architecture Document

SAD Version 1.2

Team 03: B-STACK
09 March 2021

Amun Kharel
Brandon Stringham (manager)
Cody Crane
Krista Conley
Shreeman Gautam
Tanner Hunt

CS 460 Software Engineering

Table of Contents

1 Introduction 3

2 Design Overview 4

3 Component Specifications 5

3.1 Keypad Manager 5

3.2 Code Scanner **Error! Bookmark not defined.**

3.3 Code Verification 6

3.4 Code Management 7

3.5 Code Storage Management 7

3.6 LCD display and Output Screen Operator 8

3.7 Gate Communicator 8

4 Sample Use Case 10

General User Use Case 10

5 Design Constraints 11

6 Definition of Terms 13

1 Introduction

"B-STACK" is a software design group consisting of Brandon Stringham, Shreeman Gautam, Tanner Hunt, Amun Kharel, Cody Crane, and Krista Conley. B-Stack's Software Architecture (SAD) document provides a comprehensive architectural overview of the "Community Gate Keypad" software(CGK).

The purpose of this project is to design a keypad control system for a vehicular access gate. The gate control keypad allows access into a gated facility via a code. It is a security software and hardware implementation that will communicate one-way with the gate software system to allow users access into the facility and to inform users of the keypad system messages via LCD display.

The purpose of this document is to map out the software and how the software is structured to be able to communicate between the software architect, B-STACK, and the customers, stockholders, and other developers who want to realize the project. In regards to the software architecture, the intended audience for this document is the developers for the final product. In essence, the purpose of this document is to link the idea and reality. This document will discuss the following topics:

- Design overview
- Component Specifications: A logical view
- Sample Use Case: Processing and Illustrating the execution process
- Design Constraints: Updates/New and Implementation directives
- Definition of terms

2 Design Overview

This system was designed in a modular form with components that handle a set of related actions. The reason for designing the system this way is for ease of development and maintainability. When small individual components handle the logic for a related set of actions, it keeps related code in the same places and reduces redundancy. It also allows the developers to work separately on small components that can plug together to form the completed system.

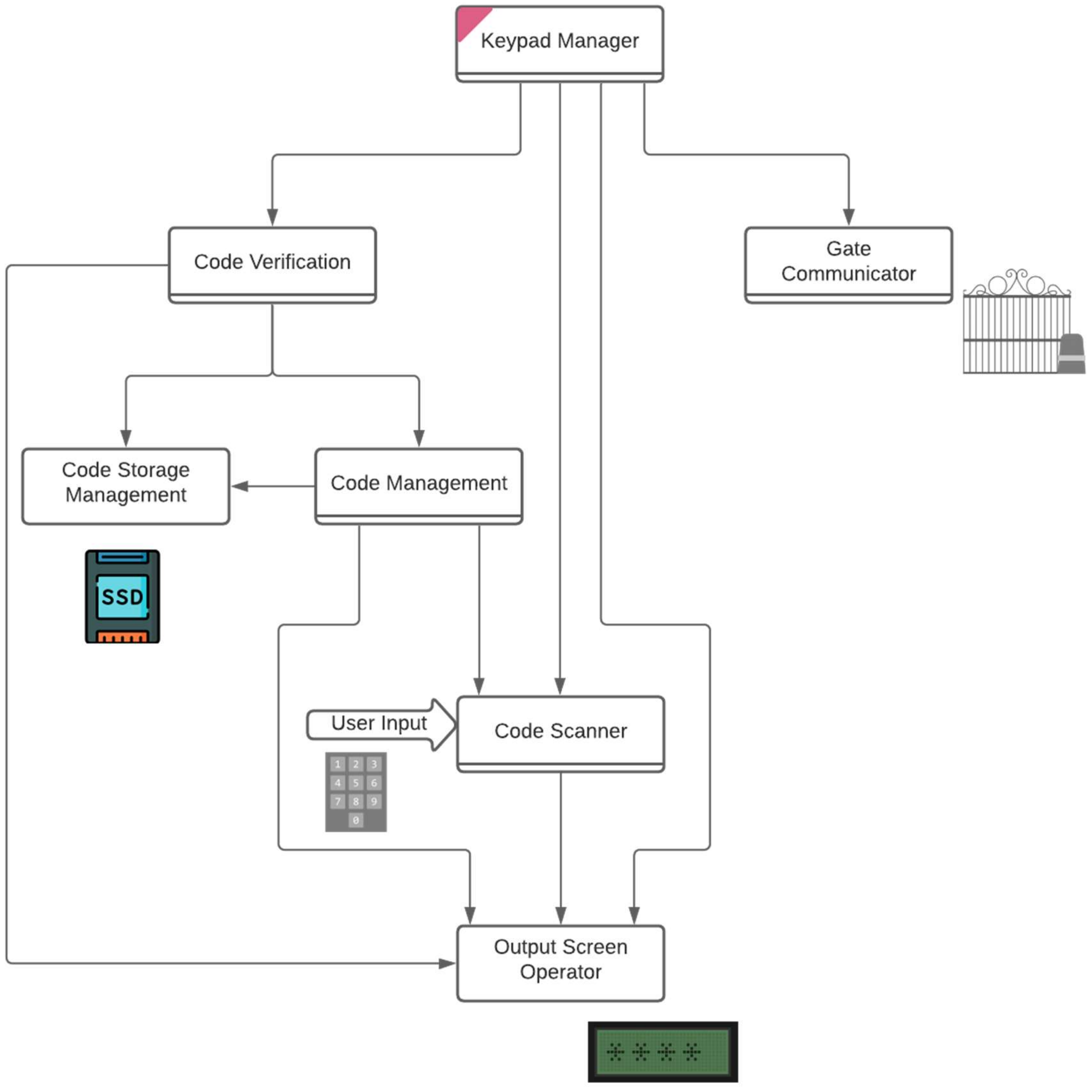

Figure 1: Software Architecture Design Diagram

In Figure 1: Software Architecture Design Diagram, there is an overarching controlling component that starts the other components when the whole system changes state. Most of these subcomponents, with the exception of the Code Verification and Code Management, do not talk directly with each other and instead return the results from their processing to the main controlling component which then takes the information and can pass it to the next component or change the state of the system appropriately.

There are a few external devices that the system connects to, the keypad, output screen, gate controller, and the data store. The Code Scanner will wait on input from the keypad, the Output Screen will display messages, the Gate Communicator tells the gate when to open, and the Code Storage Manager will manage the storage device.

3 Component Specifications

Each of the individual components in Figure 1 handle the logic related to a specific portion of the keypad system operation. Some of these components only have to handle a single action, but several handle more complex actions to accomplish their component's goal. Together, the components make up the functionality of the whole system.

3.1 Keypad Manager

The keypad manager is the overarching component that controls the keypad system and runs all of the sub-components at the proper time and manages their communication. When the system starts up, it starts up in the keypad manager. Then the keypad manager will start the code scanner and will wait for the code scanner to return a successfully entered code. Once the scanner returns a code, the keypad manager will pass the code to the code verification and wait for the verification to confirm whether the code was valid, invalid, or if the admin code section was invoked. If the code is valid, the keypad manager will output a success message and run the gate communicator and then once again launch the code scanner and wait for new input. If the input was invalid, it will send output to the output screen and wait on the code scanner for new input. If the admin code manager was invoked, the keypad will just return to waiting on the code scanner.

One of the features that the Keypad Manager is responsible for, instead of assigning to other components, is keeping track of the number of invalid code attempts. On the fifth incorrect code entry, and all subsequent incorrect codes thereafter, a brief ten second cooldown will commence before going back to the scanner and allowing a new code to be inputted. Once a correct code has been entered, the count of invalid code entries resets back to zero. This helps slow down brute force code guessing attacks.

3.2 Code Scanner

The Code Scanner component is initialized by the Keypad Manager. The Code Scanner will be in the Idle State awaiting input from the user which will be a 4-digit code. As the user inputs the digits, it will send a signal to the Output Screen Operator for each number entered. At each Key Event, it will start an internal timer. In the event that the user takes more than 10 seconds to input the next Key Event, it will output a message to the Output Screen Operator informing the user that they took too long to enter the code. Every time a Key Event is entered before the 10 seconds have expired, it will reset this internal timer. If the user inputs the 4-digits in the specified time allotment, then the Code Scanner sends this input to the Keypad Manager. If the Code Scanner is called by the Code Management then it will wait for an input of 1-digit to send back to the Code Management. Upon the second call to the Code Scanner, it will be waiting for a 4-digit code from the Admin. It will send this new code to Code Management.

The Code Scanner is basically reading in the signals from the actual hardware device of the Keypad as inputs that it will send to the Keypad Controller, Code Management, and the Output Screen Operator. It is the main interface between the user and the Keypad System.

3.3 Code Verification

Code verification handles the authentication of codes entered by the users. Code verification checks a 4 digit code received from the Keypad Manager against the codes stored locally in the Code Storage via the Code Storage Management component. If the code matches one of the codes that open the gate (User or Public Service) it notifies the Keypad

Manager, so that it may open the gate. If the code matches the admin code, control is passed to Code Management to handle the admin code functions. If the code does not match any of the valid codes in the Code Storage, control is passed back to the Keypad Manager to notify the user that the code entered is not valid.

3.4 Code Management

Code management handles the functionality of the admin code i.e. the ability to change the user and public service codes. This displays to the LCD a simple message prompting the admin to enter either 1 to change the User code or 2 to change the Public Services code. It then asks the Code Scanner for a single input, and waits for its response. If the output returned is not valid i.e. not a 1 or 2, a message is displayed and Code Scanner is asked again for a single input. Once a selection is made the admin is prompted to enter the new code and Code Scanner is asked for a four digit input. When that input is received, Code Management checks it against the stored codes and if it is valid, updates them and notifies the admin. If the code received is not valid, i.e. it matches one of the three previous codes, the admin is notified and Code Scanner is asked for another four digit input. Timeouts are handled by the Code Scanner for input/output operations.

3.5 Code Storage Management

The Code Storage Management manages the current codes for the Admin, Users and Public Services, as well as the previous two codes for the Users and Public Services, which are stored persistently in the Code Storage component, the stored codes are labeled so as to facilitate easy updates and access. These labels are managed and updated by the Code Storage Management whenever a code is changed. Code Storage Management is accessed by Code Verification to verify inputted codes. The component does not actually perform the verification, rather, it simply provides the correct code for Code Verification to check against. The codes stored here are updated by Code Management when an admin goes through the code update process and changes one or more of the codes. When this occurs, the oldest code stored for the code changed is deleted, and the new code is stored as the current valid code, as by the functionality of a queue.

3.6 LCD display and Output Screen Operator

The LCD display is a hardware component that basically displays what it is told to by the Output Screen Operator software component. Other software components utilize this LCD display to give the user feedback on their entries of digits from the actual keypad hardware device. The Output Screen Operator receives input from the Code Scanner, Keypad Manager, and Code Management to output the messages to the user via the LCD display.

The Output Screen Operation is initialized by the Keypad Manager, and will be awaiting input from the code scanner and will output a message to the user. The Code Scanner will tell it to output an '*' to the user for each input of a digit received from the scanner. It will also output the messages from the Code Management component to the user whether the code is valid or the code is invalid. It will receive an input from the Code Management if the user was the admin, to display the menu to push a 1 or 2 to choose which user type to change. The Code Management will also ask it to display a message to the admin prompting them for the new code. It will receive an input from the Code Management in the event that the user entered the wrong code 5 times in a row and will output a message to the user.

3.7 Gate Communicator

The gate communicator is a simple component that tells the gate when to open. This component is started by the Keypad Manager and gets invoked any time a correct user/public service code gets entered in the system. When the gate communicator is enabled, it sends a simple signal to the gate system through the buried wire telling the gate to open. After this quick pulse, the gate controller exits and returns control back to the keypad manager.

4 Transportation System

Transportation System is a mediator between the central control system and components of transportation systems such as self-driving cars and local passenger token stores. Transportation system is waiting for messages from Self Driving Cars which are queued and sent to the Central Control for monitoring purposes. Additionally, the Transportation System is also waiting for messages from the Central Control, which can command self-driving cars to go in a particular path or initiate emergency vehicular operation.

4.1 Self-Driving Car Controller

Self-Driving Car Controller sends messages to and receives messages from the transportation system. Self-driving sends status messages of each vehicle to the central control system. Status message of vehicle includes the location of vehicle, total number of passengers in each vehicle including token number of each passenger, and alerts messages during token mismatch. These messages are received by the central control system for monitoring and regulating cars and its passengers. Information of passengers and their respective token number in each vehicle is received by the transportation system and is also retrieved by the local passenger token storage management for storing.

Self-driving car controller also receives messages from the Transportation System, which commands self-driving cars to move in a particular path. It also receives messages to initiate emergency vehicle protocol, which makes the vehicle behave differently from the normal vehicle operation.

4.1.1 Self Driving Cars

Self-Driving Cars is managed by the Self Driving Car Controller. Self-Driving cars are directed on where to go by the Controller. Also, when there is an emergency protocol, self-driving behaves differently than the normal operation. Self-Driving cars provide their location constantly to the Self-Driving Controller. The event of offboarding passenger and onboarding passenger is also reported to the Central Control system. In an event of Token mismatch when a passenger is leaving from west to east, it is reported to the Central Control, where appropriate authority is called.

4.1.2. RFID Scanners

RFID Scanners scans the token of each passenger in each vehicle and tracks the passengers that are residing in the particular vehicle. This information is monitored in the Central Control System and tracked locally in the local passenger token storage.

4.2 Local Passenger Token Storage Management

Local Passenger Token Storage Management receives passenger token id and their unique vehicle name from the Transportation System. This information is stored in the Local Passenger Token Store.

4.2.1 Local Passenger Token Store

Information about each vehicle and its passenger is stored locally inside this SSD device.

4 Sample Use Case

Section 4 outlines one of the two use cases and will focus on the general user use case.

General User Use Case

Use case: OpenGate

Primary Actor: General user

Goal in context: Open the gate when a correct code has been entered.

Preconditions: Keypad is programmed by an admin with a general user code.

Trigger: General user decides to enter a code to open the gate.

Scenario:

1. General user observes the keypad.
2. General user enters code.
3. General user observes the gate opening.
4. General user enters through the gate.

Exceptions:

1. If the code is incorrect, the LCD displays an error message and the user has to try again.
2. If the code is left incomplete for ten seconds, the LCD resets.
3. If the code is entered incorrectly five or more times, there will be a ten second cooldown period.

When the user enters the code in the system, it is handled by the Code Scanner component which waits for the code and then sends it to the Keypad Manager component. The Keypad Manager asks the Code Verification to make sure that the code is correct and if it is, it invokes the Gate Communicator to open the gate and allow the user access.

5 Design Constraints

The keypad system will work with a gate controller that is already implemented. The communication is one-way from the keypad system to the gate controller via a buried hardwire. Our keypad system does not close the gate after a user enters nor does it open the gate from the inside; that task will be for the gate controller.

Due to the keypad system being constrained to be available 24/7, there will be a light attached to the keypad control box and the keypad system will have surge protection and waterproof sealing for inclement weather. The codes will be stored in non-volatile memory to protect the two previously used codes for both user and public service, the current user code for both user and public service, and the admin code in the event of a power outage.

The code combination will be constrained in that it will only allow input from the physical combination of the keypad numbers 0-9. The sequences are required to be 4 digits or 1 digit in length only.

The keypad system is also limited to simple message outputs to the user or admin because it only has a simple LCD display. The keypad system will not be connected to the property management wifi or phone service, therefore the system cannot call security or the property manager. Due to the one way communication, in the event of a gate malfunction, the keypad system will not be aware of the malfunction, and it will be required of the user to call the property management.

All these system constraints ensure that the community gate keypad is simple, consistent, and convenient, and it also deters brute forcing into the facility for safety.

6 Definition of Terms

Term	Description
CGK	Community Gate Keypad
B-STACK	Our software company’s name
SAD	Software Architecture Document
SRS	Software Requirements Specifications
RDD	Requirements Definition Document
LCD	Liquid Crystal Display