MEETING ROOM INSTANT BOOKING

DESIGN DOCUMENT

October 1, 2021

Author: Raghu Srivatsa M P
Approver: Ammiraju Choudhary

Contents

1 CHANGE HISTORY

Issue	Author	Date	Comment
1A	Raghu Srivatsa M P	08-Nov-2017	Initial Version
1B	Raghu Srivatsa M P	23-Nov-2017	Updated Version
1C	Raghu Srivatsa M P	15-Dec-2017	Updated the LLR

2 INTRODUCTION

This document is intended to set out the high level design, low level design and ICD for the Meeting Room Instant Booking System. This document takes the system requirement specifications and constraints of the target platform. This decomposes the software into different entities whose detailed behaviour can be designed and implemented in isolation. The architectural design takes into account the communication between different architectural layers. ICD defines the interface between hardware and software components.

3 STAKEHOLDERS

The following stakeholders have been identified in this project :

Engineering Specialist Capability Manager(Hariharan Ganesan)
Control Software Team Lead(Ammiraju Choudhary)
Principal SMDA Engineer, Digital(Mallika Menon)

4 SOFTWARE ARCHITECTURE

HLR/001/01: Software Architecture

The software architecture of the meeting room instant booking system is structured into three layers namely Hardware abstraction layer, Functional layer and Application layer. Hardware is interfaced with the sensors, mailing system and LED Display through Hardware Abstraction Layer. The functional layer consists of three modules mainly health monitoring, mailing system and LED Display and sensors. These three modules interact with one another through Application Layer. Processing of data and control actions are taken care by the Application Layer.

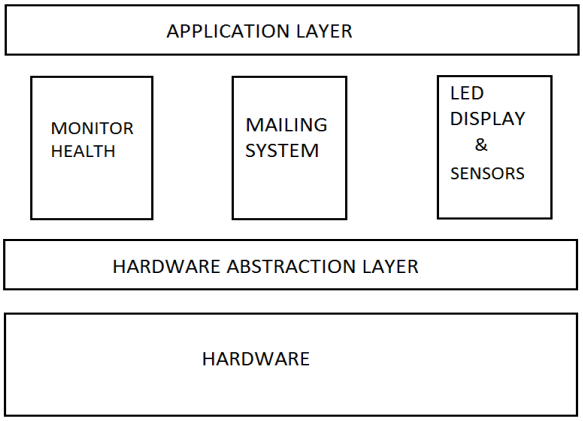

Figure 1: Software Architecture

System Architectural Layout as shown in below figure.

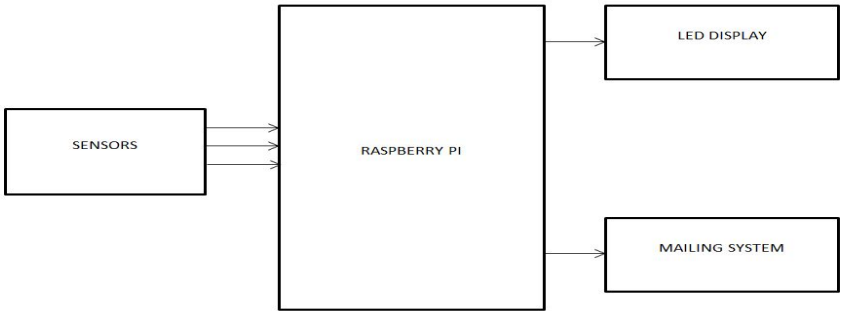

Figure 2: System Architectural Layout

Sensors are used as input devices to detect the human presence in meeting room. Sensor signals are sampled at specific intervals of time and send as the input to Raspberry pi. The received data are processed and the the

output is indicated through LED Display and mailing system.

5 LOW LEVEL REQUIREMENTS

LLR/001/02: instantBooking

BEGIN

Set GPIO pins 13 and 29 into GPIO.IN mode by calling GPIO.Setup function

Set GPIO pins 36 and 12 into GPIO.OUT mode by calling GPIO.Setup function

Call **glowLED** function

Call **sendMail** function

Every 5 minutes software shall ensure that the system is free from failure via a call to the function **monitor_health**

END

LLR/002/01: glowLed

BEGIN

Call GPIO.INPUT function with parameter 13 to set pirA to GPIO pin 13's value

Call GPIO.INPUT function with parameter 29 to set pirA to GPIO pin 29's value

If pirA or pirB is TRUE within a timeperiod of 180 seconds(3 mintues)

then

Call GPIO.OUTPUT function to set GPIO pin 36 to False

Call GPIO.OUTPUT function to set GPIO pin 12 to True

Otherwise

Call GPIO.OUTPUT function to set GPIO pin 12 to False

Call GPIO.OUTPUT function to set GPIO pin 36 to True

end if

END

LLR/003/02: monitor_health

BEGIN

If green LED is ON and red LED is ON

Set LED_FAIL_FLAG to True

Otherwise

Set LED_FAIL_FLAG to False

If the LED_FAIL_FLAG is True

FAIL_INDICATION_FLAG is set to True

Otherwise

FAIL_INDICATION_FLAG is set to False

Notify the pi admin regarding the health of the system via a call to the function **notify_pi_admin**

END

LLR/008/01: notify_pi_admin

BEGIN

If the FAIL_INDICATION_FLAG is True

Drop a mail to Pi admin stating that "Failure Detected" and Turn off both the LEDs

Otherwise

Do nothing

END

LLR/004/02: sendMail

BEGIN

Create an array of data structure demo_cases (dc) with data members

lower_limit_time,

upper_limit_time, (lower and upper limit of the time(in minutes) within which condition is being checked)

status, (booked or unbooked condition of the room)

name, (name of the person who booked the room)

mail_id, (mail id of the person who booked the room)

Initialise dc as { {0,10,'B',Maria,"maria.mathews@rolls-royce.com" },
{10,20,'U',Neenu,"neenu.thankachan@rolls-royce.com"},
{20,30,'U',Raghu,"raghu.srivatsa@rolls-royce.com"},
{30,40,'B',Amrutha,"amrutha.unniyappan@rolls-royce.com"} }

Set admin as archana.tripathi@rolls-royce.com

Initialise i=0, hour_hand=9 ,min_hand=0

Set case_size as the number of elements in demo_cases array

While i less than case_size

Set status=dc[i].status

Set block_mail_sent and alert_mail_sent to FALSE

Set counter to 0

Set OK to 0

If status is 'U'

While status is 'U' and OK is 0 and min_hand less than or equal to dc [i].upper_limit_time

If pirA or pirB is TRUE and block_mail_sent is FALSE and the function confirm_timer with parameter pin#12 and counter returns TRUE

If block_mail_sent is FALSE

send mail to admin to block the room

Set block_mail_sent to TRUE

Set admin_flag to TRUE

Set unblocked to FALSE

Set counter to 0

else If admin_flag is TRUE and unblocked is FALSE and function confirm_timer

with parameter pin#36 and counter returns TRUE
send mail to admin to unblock the room
Set unblocked to TRUE
Set status to 'B'
else if run_flag is FALSE
Call function run_timer with parameter min_hand
Set OK to 1
End while
else
While status is 'B' and OK is 0 and alert_mail_sent is FALSE and min_hand less than or equal to dc[i].upper_limit_time
If pirA and pirB are FALSE and confirm_timer function with paramteter pin#36 and counter returns TRUE
If alert_mail_sent is FALSE
send mail to dc[i].mail_id for confirmation
Set reply_flag according to the reply of the alert mail
Set alert_mail_sent to TRUE
If reply_flag is FALSE
set status to 'U'
send mail to admin to unblock
else if run_flag is FALSE
Call function run_timer with parameter min_hand
Set OK to 1
End while
Increment i
End while
END

LLR/005/02: confirm_timer

BEGIN
Set led to confirm_timer function parameter1
Set counter to confirm_timer function parameter2
If led is TRUE and counter less than 5
Delay for 1 minute
Increment counter
Increment min_hand
Delay for 1 minute
If counter is 5
return TRUE
else
return FALSE
Set run_flag to 1

Return FALSE

END

LLR/006/01: run_timer

BEGIN

Delay by 1 minute

Increment min_hand

END

6 INTERFACE CONTROL DOCUMENT

6.1 Introduction

An interface control document (ICD) in systems engineering and software engineering, provides a record of all interface information (such as drawings, diagrams, tables, and textual information) generated for a project. The underlying interface documents provide the details and describe the interface or interfaces between subsystems or to a system or subsystem.

An ICD is the umbrella document over the system interfaces; examples of what these interface specifications should describe include:

- The inputs and outputs of a single system.
- The interface between two systems or subsystems.
- The complete interface protocol from the lowest physical elements to the highest logical levels would each be documented in the appropriate interface requirements spec and fall under a single ICD for the "system".

This includes all possible inputs to and all potential outputs from a system for some potential or actual user of the system. So Interface control documents are a key element of systems engineering as they control the documented interface(s) of a system, as well as specify a set of interface versions that work together, and thereby bound the requirements.

6.2 Purpose

Interface Control Document (ICD) documents and tracks the necessary information required to effectively define the Meeting Room Instant Booking system's interface as well as any rules for communicating with them in order to give the development team guidance on architecture of the system to be developed.

The purpose of this ICD is to clearly communicate all possible inputs and outputs from the system for all potential actions whether they are internal to the system or transparent to system users. This Interface Control is created during the Planning and Design Phases of the project. Its intended audience is the project manager, project team, development team, and stakeholders interested in interfacing with the system. This ICD helps ensure compatibility between system segments and components.

6.3 Raspberry Pi: Pin Specification

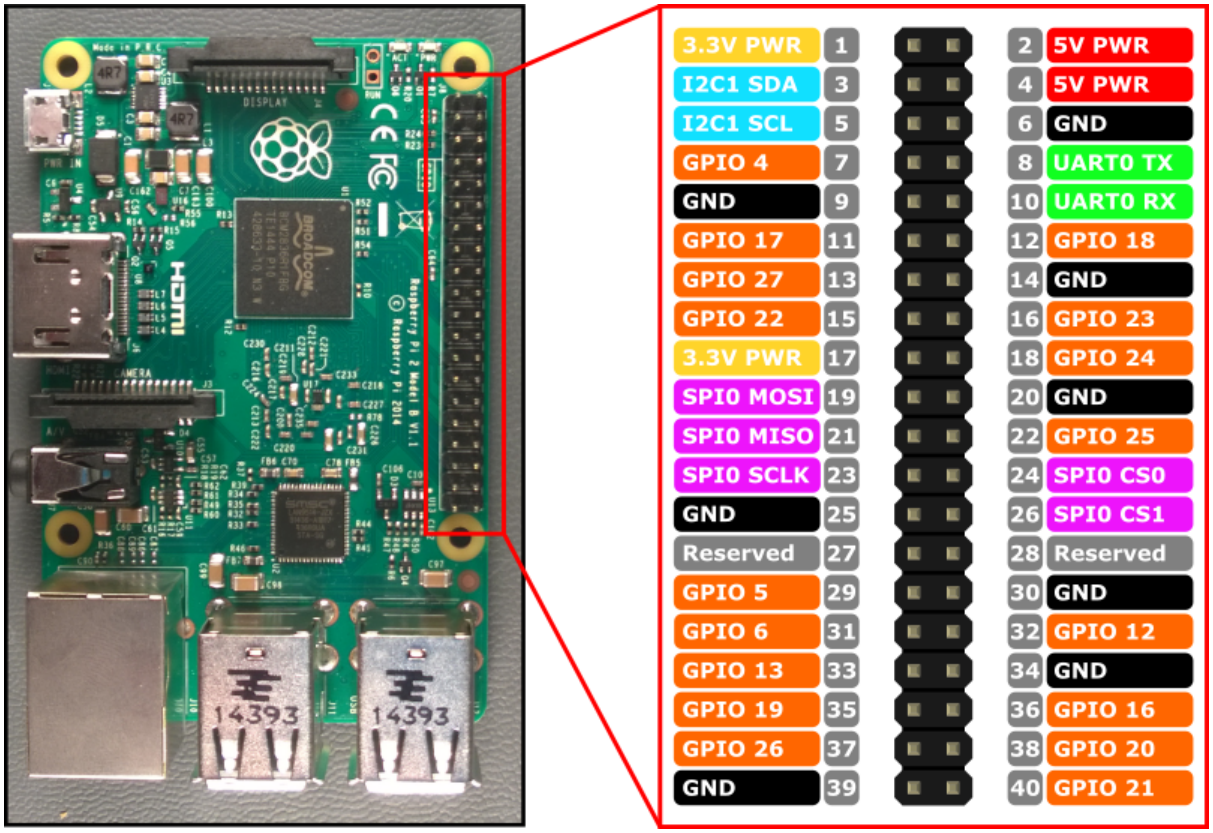

Figure 3: Raspberry Pi 3: Pin diagram

- GPIO (General Purpose Input/Output) are the standard pins that simply be used to turn devices on and off. For example, a LED.
- I2C (Inter-Integrated Circuit) pins allow us to connect and talk to hardware modules that support this protocol (I2C Protocol). This will typically take up 2 pins.
- SPI (Serial Peripheral Interface Bus) pins can be used to connect and talk to SPI devices. Pretty much the same as I2C but makes use of a different protocol.
- UART (Universal asynchronous receiver/transmitter) are the serial pins used to communicate with other devices.
- DNC stands for do not connect, this is pretty self-explanatory.
- The power pins pull power directly from the Raspberry Pi.
- GND are the pins used to ground the devices. It doesn't matter which pin we use as they are all connected to the same line.

6.4 Pin Configuration

This section describes that which raspberry pins are connected to which peripherals.

Raspberry Pins	Peripherals	Description
13	SensorA(PIR sensor)	Pin #13 is used to store output of SensorA
29	SensorB(PIR sensor)	Pin #29 is used to store output of SensorB
36	Green LED	Pin #36 is used to provide input to Green LED
12	Red LED	Pin #12 is used to provide input to Red LED

Table 1: Pin Configuration

7 TRACEABILITY

Low level Requirement Tag	Software Requirement Tag
LLR/001/02	SRS/001/01, SRS/002/01, SRS/003/01, SRS/004/01, SRS/005/01, SRS/006/01, SRS/007/01, SRS/008/01, SRS/009/01,
LLR/002/01	SRS/001/01, SRS/003/01, SRS/004/01, SRS/005/01
LLR/003/02	SRS/009/01
LLR/004/02	SRS/001/01, SRS/002/01, SRS/005/01, SRS/006/01, SRS/007/01, SRS/008/01
LLR/005/02	derived from LLR/004/02
LLR/006/01	derived from LLR/004/02

High level Requirement Tag	Software Requirement Tag
HLR/001/02	SRS/001/01, SRS/002/01, SRS/003/01, SRS/004/01, SRS/005/01, SRS/006/01, SRS/007/01, SRS/008/01, SRS/009/01,