Dev Comm. Service

V0.0

Table of Contents

[1. PROJECT DESCRIPTION 3](#_Toc456111241)

[1.1 Project Name 3](#_Toc456111242)

[1.2 Project Description 3](#_Toc456111243)

[1.3 Document Information 3](#_Toc456111244)

[1.4 Revision History 4](#_Toc456111245)

[2. INTRODUCTION 5](#_Toc456111246)

[2.1 Purpose 5](#_Toc456111247)

[2.2 Scope 5](#_Toc456111249)

[2.3 Overall Description 5](#_Toc456111250)

[2.3.1 Overall System View 5](#_Toc456111251)

[2.3.2 Game Information 5](#_Toc456111256)

[2.4 Hardware Environment 5](#_Toc456111257)

[2.5 Software Environment 5](#_Toc456111258)

[2.5.1 Technologies Used 5](#_Toc456111259)

[2.5.2 Supported Browsers 5](#_Toc456111260)

[2.6 Descriptions, Acronyms, and Abbreviations 5](#_Toc456111261)

[3. FUNCTIONALITIES SUPPORTED 5](#_Toc456111262)

[3.1 Summary 6](#_Toc456111263)

[3.2 Internal Architecture and design 6](#_Toc456111340)

[4. Risks and Known Issues 8](#_Toc456111341)

[4.1 Raspberry PI, Plugload, and PI Connection 8](#_Toc456111342)

[5. REFERENCES 8](#_Toc456111348)

[5.1 Documents 8](#_Toc456111349)

[6. APPENDIX 8](#_Toc456111351)

1. PROJECT DESCRIPTION
   1. Project Name

DCS (Dev Comm. Service)

* 1. Project Description

DCS is integral part of Social Game. It works between Social Game Web layer and raspberry pi to execute command given by web layer.

* 1. Document Information

|  |  |
| --- | --- |
| Document Title: Dev Comm. Service Software Requirements Specification | |
| Department: IT TEAM | Document State: Draft |
| Author: Prakhar Vishnoi | Creation Date: 16/5/2017 |
| Checked by: | Checked Date: |
| Approved by: | Approved Date: |
| Version Number: 0.0 | No. of Pages: |

* 1. Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Revised version. No. | **REVISED CONTENTS** | | | | |
| Version no. | Section Numbers Changed | Summary of Changes | Date Changed | Revised By | Approved By |
| 0.0 | All | Draft | 16-May-2017 | Prakhar |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. INTRODUCTION  
   1. Purpose

The purpose of this document is to list required functionalities and user interfaces of the DCS Project.

This document is give understanding of architecture, usage and functionality of DCS.

* 1. Scope

The DCS is use for stabilising communication between web-portal and raspberry pi. DCS is comprises of web service, which get command from web portal depending on user input and corresponding to that pass instruction to program in raspberry Pi to perform action for Eg. Switching on-off plug loads.

* 1. Overall Description

The project will be comprises of many web services, which will perform different actions. Some Web services are manipulation of others (Eg. Switching one or multiple plug loads).

1. Process Instruction – To switch on/ off the plug loads and return the response.
2. Process Multiple Instruction- To switch on/off multiple plug loads and return response corresponding to input.
3. Get Data- To get data for plug loads (Eg. Relay, energy, power)
4. Get Status- To get status of plug loads (mac id for single plug load and “all” for all plug loads).
5. Switch on All Plug Loads- To switch on all plug loads.
6. Summary Of All Plug Loads- To get summary of all plug loads( how many plug load on, off and offline).
   1. Hardware Environment

N/A

* 1. Software Environment
     1. Technologies Used
* Database – PI Archive
* Web Application - Java, Spring
  1. Descriptions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Acronyms/Abbreviations** | **Description** |
| UI | User Interface |
| GUI | Graphical User Interface |
| PI Data Archive | OSIsoft PI System Data Archive |
| DCS | Dev Comm. Service |
| RP | Raspberry pi |

1. FUNCTIONALITIES SUPPORTED

DCS main functionality to provide process instruction and give information to web portal, with this DCS maintain internal cache and continue copy-incoming data to Pi Db.

* 1. Summary

|  |  |  |  |
| --- | --- | --- | --- |
|  | Category | Function | Remarks |
| 1 | Web Services | 1-processInstruction  2-processMultipleInstruction  3-getData  4-getStatus  5-switchOnAllPlugLoads  6-summaryOfAllPlugLoads | Summary of these module has already been given in2.3.1 |
| 2 | Cache Service | Caching Service helps to create internal cache of information process instruction. |  |
| 3 | Copy data in Pi Server | Service used to copy data from sensors to pi server |  |

* 1. Internal Architecture and design

Basic Diagram- RP

Inserting and Receiving (DCS)

Data from UI System to integrate RP

Between System and RP

RP

Client

Server

RP

RP

RP

RP

RP

ob

ob

ob

QUEUE

CACHE

PI SERVER

Component and Architecture-

* Component

Client/Server

Cache

PiServer

QUEUE

**Raspberry pi**

Basic Diagram – In basic diagram, instruction come from UI module to DCS module and corresponding to input instruction either turn ON/OFF instruction goes to raspberry pi or information fetch from internal cache.

DCS internally maintain few module to run application. Eg. Client and Server, cache etc. these are internally connected with each other to do communication. Details of DCS module has shown in second diagram.

Data Flow and component usage-

1. Client/ Server – Client used to contact with raspberry pi server and pass the instruction.

Server is use to connect with different raspberry pi and get data from it. Data receive from client (RP) every minute for relay, energy and power. After passing the instruction from client, server again receive corresponding response.

1. Cache- Internal cache has used to save the information about all plug loads. It has used to retrieve information of loads and while running the command it has used to cross verify the plug load information.
2. Pi Server- Pi server used for saving the real time data in database. It has pi tags in database and real time data get save in it.
3. Queue- Queue has used for producer- consumer problem. Queue has used to put instruction object and consumer thread run on it and process it.
4. Raspberry pi- RP is use for receiving the command and to execute it on plug loads.

Architecture Overview and Flow- DCS main component is Client and Server. These (Server/Client) run by parallel thread. Server helps to continue fetching the data from different plug loads saving it in internal cache.

Client helps to send any operational request to RP and swathing on and off the loads. Client take the request from queue and process it.

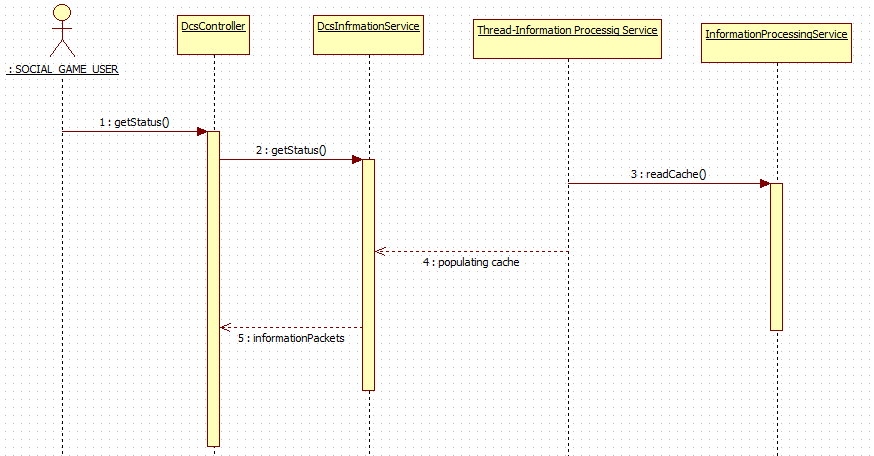
Once request of instruction reach to controller and after making the instruction packet object it get save in processing queue. Separate thread take these object and process it through client, which write the request on specific port of RP Server.

Corresponding to processed request, the output come to Server port of DCS system. System receives the request and update the information object that is in cache. Server on separate thread consumes information of all plug loads and save it in cache; same information goes to Pi database server under pi tags.

Flow Diagram-

For any get request to retrieve information it goes to server and then it calls the service layer. From service, layer, it check the internal cache and get the info and send it back.

Eg. Get Status



1. Risks and Known Issues
   1. Raspberry PI, Plug load, and PI Connection

The system cannot guarantee the connection between Raspberry PI and Plug load. In the event that Raspberry PI or Plug load is inaccessible or down:

* + Commands sent to the plug loads may not be executed.

If a player owns 3 plug loads, but 1 plug load is inaccessible and an OFF schedule is to be executed, then only 2 plug loads will be turned off

* + there will be no data sent from Raspberry PI to PI Interface at this moment – which may result to gaps in graph data
  + If Raspberry PI is inaccessible, the “ON” and “OFF” buttons in Game Page will be disabled

1. REFERENCES
   1. Documents

* DCS\_RequirementsDocument\_WebApp\_v0.1.docx -

1. APPENDIX

End