**Software Design Specification (SDS)**

Revision History: (Double-Bloom for unifications)

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 2019.4.8 | Renxiang Zhu | The sds document |
| 2019.4.8 | Zhi Zhou | The Detailed design |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[1. Introduction 1](#_Toc5602370)

[1.1. Intended Audience and Purpose 1](#_Toc5602371)

[1.2. How to use the document 1](#_Toc5602372)

[2. System Design 1](#_Toc5602373)

[2.1. Context 1](#_Toc5602374)

[2.2. Design Pattern 1](#_Toc5602375)

[2.3. Architecture 1](#_Toc5602376)

[2.3.1. <Component Diagram> 2](#_Toc5602377)

[2.3.2. <Deploy Diagram> 2](#_Toc5602378)

[3. Module Interface Design 2](#_Toc5602379)

[4. Detailed Design 2](#_Toc5602380)

[4.1. Server Detailed Design 2](#_Toc5602381)

[4.2. Client (Android) Detailed Design 2](#_Toc5602382)

[4.3. Client (Web) Detailed Design 2](#_Toc5602383)

[4.4. Intelligent Control Detailed Design 2](#_Toc5602384)

[4.5. Database Detailed Design 2](#_Toc5602385)

[A.    Appendices 3](#_Toc5602386)

[A.1    Definitions and acronyms 3](#_Toc5602387)

[A.1.1    Definitions 3](#_Toc5602388)

[A.1.2    Acronyms and abbreviations 3](#_Toc5602389)

[A.2    References 3](#_Toc5602390)

## Introduction

## Intended Audience and Purpose

<Every technical document should clearly specify who the document is written for and what purpose the document should serve for each intended audience. This section describes the purpose and audience for the Concept of Operations and the Software Requirements.>

## How to use the document

<Describes the document organization. This section should answer for the reader: “Where do I find particular information about X?”>

## System Design

<Use this section to give a detailed description of the system contexts from an architect's point of view. It should make clear the expected context of the software, such as the platform, design pattern, etc.>

## Context

<Specifies the system's operational context: i.e., the programming languages to develop the software with, the operating system your software runs on, the database management system your data will be stored, the internet protocol for the component communication, etc.>

Programming languages:python

Operating system:

Linux:

* Red Hat Enterprise Linux 5.5+1, 6.x (32-bit), 6.x (64-bit)2
* Red Hat Enterprise Linux 7.x (64-bit)2 (8u20 and above)
* Ubuntu Linux 12.04 LTS, 13.x
* Ubuntu Linux 14.x (8u25 and above)
* Ubuntu Linux 15.04 (8u45 and above)
* Ubuntu Linux 15.10 (8u65 and above)

Windows:

* Windows 10 (8u51 and above)
* Windows 8.x (Desktop)
* Windows 7 SP1
* Windows Vista SP2
* Windows Server 2008 R2 SP1 (64-bit)
* Windows Server 2012 and 2012 R2 (64-bit)

Mac OS X:

* Intel-based Mac running Mac OS X 10.8.3+, 10.9+

Internet protocol:IPV4

## Design Pattern

<Specifies the technical details of the software system: i.e., model-view-control division, restful service pattern, etc.>

## Architecture

## <Component Diagram>

<Component Diagram (CD) specifies how the system is parted according to the use cases analyzed from RS. >

The whole system is divided into five parts according to the SRS document, known as the Intelligent Control(IC), the Server, the Database(DB), the Web Client and the App Client. The server part mainly deals with the interaction functions with hardware, intelligent control modules, applications and databases. Its role can be considered as a transit station, as a hub for various information processing and interaction. According to the SRS file, the server mainly provides six services: (1) stable link between hardware (2) receiving hardware response (3) accepting hardware status data (4) accepting end user requirements (5) responding to end user query (6) End user access to the database to provide access.

There are totally 5 ways for other parts to get access to the server, which are server\_report, server\_register, server\_query, server\_command, server\_DB. All the details are implemented in the fourth quarter.

## <Deploy Diagram>

<这里应该是个总图>

## Module Interface Design

< It specifies the contracts with which the modules communicate.>

  (Mogic for System Interface Specifications, extra template available; all groups should contribute via interface design of her own module.)

### 3.1 The hardware build socket connection with server.

When hardware is running, it should try to build sockets with server, in the following steps:

1. Try to build a reporting socket with server.(If the hardware is a actuator, it should build another receiving socket for receiving command.)
2. When socket is built, the hardware should sent a package to server using this socket for authenticating it’s identification.
3. The hardware will received a package in form of ‘{‘status’:0, ‘msg’:’...’}’ indicating whether the socket is admitted by the sever.

### 3.2 When actuator received a command.

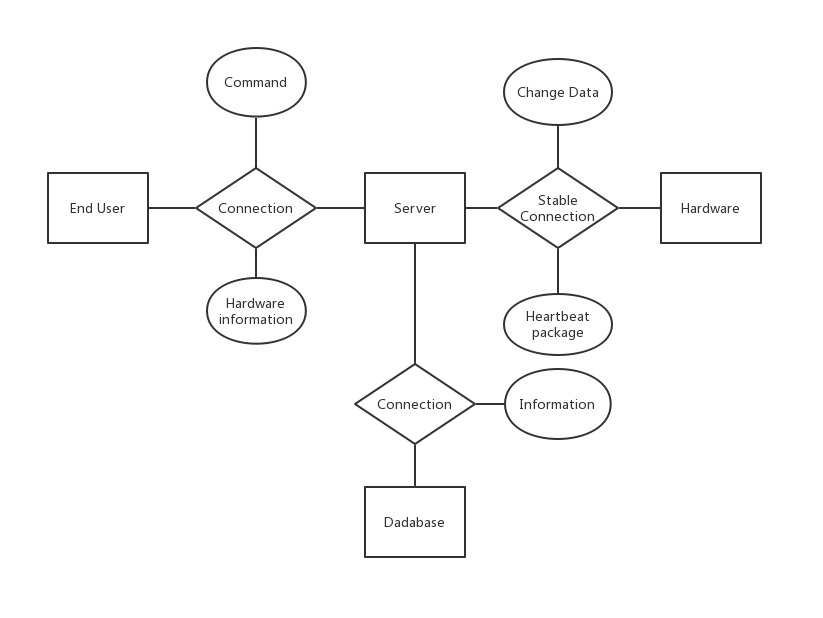
When actuator received a command from server and this command let the actuator’s state changed. The actuator should report this command with its up-to-date state to server using the ‘server\_report’ method through the socket.

## Detailed Design

< It specifies the design information inside the modules.>

(Each group should contribute, via diagrams either for the whole system or for her module. Optional diagrams are ER diagram, Sequence diagram, Class diagram)

## Server Detailed Design



## Services Provided

|  |  |  |  |
| --- | --- | --- | --- |
| # | Service | Provided By | Tested By |
| 1 | Hardware want to build stable connections with server. | server\_register | TC 5.1 |
| 2 | Hardware report data to server when it changes. | server\_report | TC 5.3 |
| 3 | Hardware send heartbeat package to server. | server\_report |  |
| 4 | End user want to query hardware’s information. | server\_query, DB\_checkAuthority | TC 5.2 |
| 5 | End user want to send command to hardware. | server\_command, DB\_queryHardware, DB\_checkAuthority | TC 5.2 |
| 6 | End user want to communicate with DB.(When it comes to something about user, building, room and so on.) | server\_DB, DB’s other functions. | TC 5.4 |

## Access Method

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Access Method** | **Parameter name** | **Parameter type** | **Description** | **Exceptions** | **Map to services** |
| server\_report | Hardware use the socket which built in register stage to send data. | data | Text in JSON format | The content of data could be different when hardware want to report its data or send heartbeat package. |  | 2, 3 |
| server\_register | Hardware built a socket with server on ‘IP:443’. And then send data. | data | Text in JSON format | The JSON format data should contains the following fields:  ‘hid’: Hardware’s unique ID.  ‘type’: The type of this socket. (Report / Receive)  ‘auth’: The authenticating key which will be confirmed by server. | Wrong hardware’s ID or authenticating key. | 1 |
| server\_query | Send post/get request to ‘IP:80/api/hardware’ | data | HTTP request parameters package | The JSON format data should contains the following fields:  ‘uid’: The user’s unique ID.  ‘sid’: The user’s security ID.  ‘hid’: The target hardware’s unique ID. | The user don’t have authority to access the hardware. | 4 |
| server\_command | Send post/get request to ‘IP:80/api/command’ | data | HTTP request parameters package | The JSON format data should contains the following fields:  ‘uid’: The user’s unique ID.  ‘sid’: The user’s security ID.  ‘hid’: The target hardware’s unique ID.  ‘cmd’: The user command. | The user don’t have authority to access the hardware. | 5 |
| server\_DB | Send post/get request to ‘IP:80/interface/<type>/<task>’ | data | HTTP request parameters package | The data should contain everything that the interface of DB need. | According to the DB’s response. | 6 |

## Access Method Effects

|  |  |
| --- | --- |
| **Access Method** | **Description** |
| server\_report | If the received JSON data is empty which indicating that hardware sent a heartbeat package, the server will update the hardware’s last updating time to keep it in a online state.  If the received JSON only contains ‘data’ filed indicating that the sensor want to report it latest data, the server will record the data in the RAM or Redis.  If the received JSON data contains ‘data’ and ‘cmd’ filed which indicating that the actuator’s state is change by a command, the server will record it’s latest data and the latest efficient command in RAM or Redis. |
| server\_register | The server will check whether the authenticating key is correct or not.  After, the server will check the hardware’s identification according to the the information which got from DB.  The server will build a report / receive socket according to the filed ‘type’. |
| server\_query | The server will check whether the user has authority to access this hardware.  Next, the server will request hardware’s persistent information like it’s nickname, type and so on from DB.Combining the real-time data which stored in server, the server will return this to clients. |
| server\_command | The server will check whether the user has authority to access this hardware.  Next, server will ask the IC for command which need to be sent to hardware.  Finally, the server will send the command to hardware. |
| server\_DB | The server will redirect this request to DB’s API and return what DB response to client. |

## Services Required

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Access Method** | **Parameter name** | **Parameter type** | **Description** | **Exceptions** | **Map to services** |
| DB\_checkAuthority | Send post/get request to ‘...’ | data | HTTP request parameters package | The data should contains the following fields:  ‘uid’: The user’s unique ID.  ‘sid’: The user’s security ID.  ‘hid’: The target hardware’s unique ID. |  | 4, 5 |
| DB\_queryHardware | Send post/get request to ‘...’ | data | HTTP request parameters package | The data should contains the following fields:  ‘uid’: The user’s unique ID.  ‘sid’: The user’s security ID.  ‘hid’: The target hardware’s unique ID. |  | 4 |
| DB’s other functions. | Send post/get request to ‘...’ | data | HTTP request parameters package | According to the functions of DB. |  | 6 |

## Client (Android) Detailed Design

## Client (Web) Detailed Design

## Intelligent Control Detailed Design

## Database Detailed Design

## A.    Appendices

## A.1    Definitions and acronyms

## 

## A.1.1    Definitions

|  |  |
| --- | --- |
| **Keyword** | **Definitions** |
|  |  |
|  |  |
|  |  |
|  |  |

## A.1.2    Acronyms and abbreviations

|  |  |
| --- | --- |
| **Acronym or**  **Abbreviation** | **Definitions** |
|  |  |
|  |  |
|  |  |

## A.2    References