**Medtronic MITG**

**Research & Development**

**DMM Detail Design Document for**

**Common Client Application v1.0**

**RE00026724**

|  |  |  |  |
| --- | --- | --- | --- |
| Issue Date: Refer to Agile | | | Page 1 of 54 |
|  | | | |
| Revision History: | | | |
| Revision | SDA | Description | |
| A | RC033867 | Initial Release | |
|  |  |  | |
|  |  |  | |
|  |  | | |
| Author: Refer to Agile for approvals | | | |
| Maverick Zhang – Insigma US (Manual Approval) | | | |
|  | | | |
| **Approvals:** Refer to Agile for approvals | | | |
| Paul Ourada – MITG R&D | | | |
| Robert Boyer – PMR R&D | | | |
| Shawn Li- MI Tech Consultant | | | |
|  | | | |

Table of Contents

[1. INTRODUCTION 4](#_Toc433358408)

[1.1 Overview 4](#_Toc433358409)

[1.2 Scope 4](#_Toc433358410)

[2. REFERENCED DOCUMENTS 4](#_Toc433358411)

[3. DEFINATIONS AND ACRONYMS 4](#_Toc433358412)

[4. ASSUMPTION 4](#_Toc433358413)

[5. INTERNAL COMPONENTS 5](#_Toc433358414)

[6. REST HANDLER 5](#_Toc433358415)

[6.1 DMM RESTful Web Service APIs 5](#_Toc433358416)

[6.1.1 Message Pushing 6](#_Toc433358417)

[6.1.2 User Authentication 6](#_Toc433358418)

[6.1.3 Device Management 7](#_Toc433358419)

[6.1.4 Device Configuration Management 7](#_Toc433358420)

[6.1.5 Device Log Management 8](#_Toc433358421)

[6.1.6 Device Software Management 8](#_Toc433358422)

[6.1.7 Feature License Management 8](#_Toc433358423)

[6.1.8 Scheduler Management 9](#_Toc433358424)

[6.2 Vital Sync Web Service Reference 9](#_Toc433358425)

[6.3 GDMP Web Service Reference 10](#_Toc433358426)

[7. BUSINESS SERVICE 10](#_Toc433358427)

[7.1 Sign On & Session Management 10](#_Toc433358428)

[7.1.1 Class Diagram 10](#_Toc433358429)

[7.1.2 Functions 10](#_Toc433358430)

[7.2 Notification 15](#_Toc433358431)

[7.3 Device Management 16](#_Toc433358432)

[7.3.1 Class Diagram 16](#_Toc433358433)

[7.3.2 Functions 16](#_Toc433358434)

[7.4 Device List Management 17](#_Toc433358435)

[7.4.1 Class Diagram 17](#_Toc433358436)

[7.4.2 Functions 17](#_Toc433358437)

[7.5 Device Status Management 19](#_Toc433358438)

[7.5.1 Class Diagram 19](#_Toc433358439)

[7.5.2 Functions 20](#_Toc433358440)

[7.6 Device Configuration Management 28](#_Toc433358441)

[7.6.1 Class Diagram 28](#_Toc433358442)

[7.6.2 Functions 28](#_Toc433358443)

[7.7 Device Log Management 30](#_Toc433358444)

[7.7.1 Class Diagram 30](#_Toc433358445)

[7.7.2 Functions 30](#_Toc433358446)

[7.8 Software Management 32](#_Toc433358447)

[7.8.1 Class Diagram 32](#_Toc433358448)

[7.8.2 Functions 32](#_Toc433358449)

[7.9 Feature License Management 36](#_Toc433358450)

[7.9.1 Class Diagram 36](#_Toc433358451)

[7.9.2 Functions 36](#_Toc433358452)

[7.10 Scheduler Management 39](#_Toc433358453)

[7.10.1 Class Diagram 39](#_Toc433358454)

[7.10.2 Functions 39](#_Toc433358455)

[8. UTILITY 40](#_Toc433358456)

[8.1 Business Rule Handler 40](#_Toc433358457)

[8.2 Scheduler Generator 40](#_Toc433358458)

[8.3 Data Protector 40](#_Toc433358459)

[8.4 Log 41](#_Toc433358460)

[9. COMMON SERVICE 41](#_Toc433358461)

[9.1 Sign On & Entitlement 41](#_Toc433358462)

[9.1.1 User Session 41](#_Toc433358463)

[9.1.2 User Rights 42](#_Toc433358464)

[9.2 Resource Management 43](#_Toc433358465)

[9.2.1 CC Role Mapping File 43](#_Toc433358466)

[9.2.2 DMM Configuration File 44](#_Toc433358467)

[9.3 Data Operation Service 44](#_Toc433358468)

[9.3.1 Class Diagram 44](#_Toc433358469)

[9.3.2 Functions 44](#_Toc433358470)

[9.4 Metadata Service 49](#_Toc433358471)

[10. AUDITING TRAIL 49](#_Toc433358472)

[11. CC UPDATE 49](#_Toc433358473)

[12. CC TIMER 50](#_Toc433358474)

[12.1 CC Trigger 50](#_Toc433358475)

[13. Appendix A 50](#_Toc433358476)

[14. Appendix B 54](#_Toc433358477)

# INTRODUCTION

## Overview

This document describes the detail design of the DMM (Data Management Module). It defines the packages, classes, sequence chart and work flows.

## Scope

The scope of this document is describing detail design of DMM for Common Client.

# REFERENCED DOCUMENTS

[1] RE00026719 System Requirements Specification CC

[2] RE00026722 High Level Architecture Document CC

[3] RE00026725 VS Interface Dependency Document CC

[4] R0034524\_F\_Interface\_Spec\_Messaging.docx

[5] RE00026726 GDMP Agent Dependency Document CC

# DEFINATIONS AND ACRONYMS

| Term or Abbreviation | Description |
| --- | --- |
| CC | Common Client |
| DMM | Data Management Module, A core component of CC |
| GDMP | Gateway Device Management Platform |
| GDMPA | Gateway Device Management Platform Agent |
| VS | Vital Sync. A Medtronic remote monitoring platform. |
| LB | Load Balance |
| DPAPI | Data Protection API. It is .Net framework API, it will use the user specific information to generate a key and protect the data in memory |

# ASSUMPTION

DMM will be designed under the following assumptions:

* Vital Sync will provide all required RESTful Web Service APIs.
* GDMP will provide all required APIs as RESTful Web Service.

# INTERNAL COMPONENTS

DMM will include the following internal components:

|  |  |
| --- | --- |
| **Component Name** | **Sub Component Name** |
| REST Handler | DMM RESTful Web Service APIs |
| Vital Sync Web Service Reference |
| GDMP Web Service Reference |
| Utility | Business Rule Handler |
| Scheduler Generator |
| Data Protector |
| Business Service | Sign On & Session Management |
| Device Management |
| Configuration Management |
| Software Management |
| Notification |
| Log File Management |
| Scheduler Management |
| Feature License Management |
| Common Service | Sign On & Entitlement services |
| Data Operating Service |
| Resource Management |
| Metadata Service |
| Auditing Trail | Auditing Trail |
| CC Timer | CC Scheduler |
| CC Trigger |

Following diagram shows the relationship among the internal components.



# REST HANDLER

## DMM RESTful Web Service APIs

It will use ASP .NET Web API to build the APIs and will use SignalR to push notifications to client.

Classes in REST HANDLER are only the interface for Biomed App and CC Timer. In most cases, the requests will be passed to [Business Service](#_BUSINESS_SERVICE) directly.

### Message Pushing

DMM will use SignalR to push message to client. In order to push message to client correctly, any API which will generate async message should provide a connect ID so DMM knows the notification is for which client.



Push Message will use the connection ID to locate the target web page and then call a JS function of that page to pass a notification.

The JS function and Notification structure is the contracts between DMM and Biomed App for message pushing.

How the whole notification process work, please see [Notification](#_Notification).

Interface IHubContext is from SignalR framework.

### User Authentication

#### Class Diagram



#### Description

Authentication Controller will expose APIs for user login, logout, get facilities that associate with user, set facility that user will work on, get countries, get device types that available for facilities and prepare screen for standalone.

Most API call returns DMMResp which contains Response Code and Response Msg. Biomed App should use Response Code to find string of CC’s installation language, only use Response Msg when there is no corresponding message in the resource file.

Class SignOnReq is the model for sign on request.

Class Facility is the model for facility detail information.

Class DeviceType is the model for device type detail information.

Class Country is the model for country detail information.

Class FacilityListResp is the model for response of getting facility list request.

Class CountryListResp is the model for response of getting facility list request.

Class DeviceTypeListResp is the model for response of getting facility list request.

Please see [Business Service](#_Sign_On_&) for how these requests are handled by DMM.

### Device Management

#### Class Diagram



#### Description

Device Controller will expose APIs for getting device list, getting device list of virtual dock, search device, filter device list.

Class Device is the model for device information.

Class DeviceListResp is the model for a list of device.

Please see [Business Service](#_Device_Management) for how these requests are handled by DMM.

### Device Configuration Management

#### Class Diagram



#### Description

Device Configuration Controller will expose APIs for Sync Device Configuration and View Device Configuration.

Please see [Business Service](#_Device_Configuration_Management) for how these requests are handled by DMM.

### Device Log Management

#### Class Diagram



#### Description

Device Log Controller will expose APIs for Sync Device Log and View Device Saved Log.

Please see [Business Service](#_Device_Log_Management) for how these requests are handled by DMM.

### Device Software Management

#### Class Diagram



#### Description

Device Software Controller will expose APIs for Register Software, Update Latest Software and Update Specific Software.

Please see [Business Service](#_Device_Software_Management) for how these requests are handled by DMM.

### Feature License Management

#### Class Diagram



#### Description

Feature License Controller will expose APIs for Download Feature License, Update Feature License, View Feature License and Feature License Matrix

Please see [Business Service](#_Device_Software_Management) for how these requests are handled by DMM.

### Scheduler Management

#### Class Diagram



#### Description

Schedule Controller will expose APIs for Schedule Device Log, Schedule Latest Software Update, Schedule Specific Software Update and Schedule Job Report.

Please see [Business Service](#_Feature_License_Management) for how these requests are handled by DMM.

## Vital Sync Web Service Reference

This is automatically generated by reference to Vital Sync Web Service. DMM will use this reference class directly when call VS Web Service API. For the detail of Vital Sync Web Service, please see *Vital Sync Services for Common Client-v0.1.docx* and *CommonClientVitalSyncDependencies.xlsx*.

## GDMP Web Service Reference

This is automatically generated by reference to GDMP Web Service. DMM will use this reference class directly when call GDMP Web Service API. For the detail of GDMP Web Service, please see *CommonClientGDMPDependencies.xlsx*.

# BUSINESS SERVICE

## Sign On & Session Management

### Class Diagram

This class will handle the business logic for Sign On and Session Management.



### Functions

#### Sign On

* Workflow



* Description

This function will try to login to Vital Sync first and then get role from Vital Sync. Vital Sync will help to login to GDMP. For the first time, VS will ask for GDMP credential and then save it in VS. When GDMP credential is changed, VS login to GDMP will fail. VS will ask for it again and update it in VS.

If login to VS success and VS login to GDMP success, Vital Sync will return user’s training records; GDMP roles and GDMP Agent session ID.

Roles can be mapped into CC roles Class A, B, C. Based on that information, system generate user rights and save it in user session.

Note: The Biomed App should setup a timer to show system busy message if Sign On doesn’t return in pre-configured time.

If login to Vital Sync failed too many times, user account will be locked by Vital Sync. DMM only need to show the error message.

CC support one user account with multiple logins at same time; each logins will have its own user session.

* CC Version

When login successfully, it also should send a CC Version message to GDMP to report current CC version and GDMP should return the latest CC version back. Compare the latest version with current version to see if there is a new version of CC. If there is, auto update if it’s standalone mode; alert user if it’s centralized mode.

#### Login GDMP

* Workflow



* Description

This function only gets called when user first time login to VS, or user’s GDMP credential changed. It will continue the process paused in Sign On.

#### Get Facility List

* Workflow



* Description

This function will send get facility list request to GDMP and return response message. GDMP will return all facilities associated with the user.

#### Set Facility

* Workflow



* Description

This function will save facility in User Session for later use.

#### Get Country List

* Workflow



* Description

This function is for Prep Screen. It will get all supported countries from Agent.

#### Get Device Type List

* Workflow



* Description

This function is for Prep Screen. It will get all available device types of facilities from GDMP.

#### Prep Screen

* Workflow



* Description

This function is only available for standalone mode. DMM will prepare software, feature license file download for all devices within the facilities.

#### Logout

* Workflow



* Description

This function will clean up the resource that has been allocated to this user and then logout from Vital Sync.

In order to handle the time out scenario, when HTTP Session time out, call this function to logout from Vital Sync.

## Notification

DMM will use same interface to push messages to client. A dedicated thread will be created to push message to all clients. All notification of a user should be put into a notification queue and wait for the monitor thread to push it. It’s a producer consumer pattern.



The monitor thread will be created and destroyed during CC startup and shutdown. The monitor thread will keep running and push notifications to clients.

For notification detail, please see [Appendix A](#_Appendix_A)

## Device Management

### Class Diagram



### Functions

#### Get Devices

* Workflow



* Description

This function will try to get devices from Vital Sync and GDMP. If it’s not the first time, DMM get the device list from Device List Manager directly and filter it by user access rights. When it’s first time, it will subscribe user session to Device List Manager and Device Status Manger. It’s Device List Manager’s responsibility to detect devices changes and notify user session that there is something changed. [Please see Device List Management](#_Device_List_Management) for detail of retrieve device list from Vital Sync and GDMP. [Device Status Manager](#_Device_Status_Management) is response for monitoring device status and do initialize job such as stat device, create device, register software, download feature for each device.

#### Get Devices Virtual Dock

* Workflow



* Description

This function will try to get devices from Vital Sync and GDMP. For each device, it will perform the device dock process.

## Device List Management

This class will help to get device list and keep an eye on device list changes, such as device connected, device disconnected, new device discovered. All user session will subscribe to it for device list. This class can see all devices within the system; it’s user session’s responsibility to filter the device list by user rights.

When deploy mode is centralized, the monitor thread should start when system startup; when deploy mode is standalone, the monitor thread should start when first user login successfully.

### Class Diagram



### Functions

#### Subscribe

* Description

This function will add a user session to subscriber list. Every time the device list changed, Device List Manager will update all subscribed user session with new device list.

#### Unsubscribe

* Description

This function will remove a user session from subscriber list.

#### Retrieve Device List

* Workflow



* Description

This function will try to get device list from Vital Sync and GDMP based on CC deployment mode and GDMP Server status.

#### Refresh Device List Main

* Workflow



* Description

This function will keep running and refresh the device list. When the device changed it will call all subscribers’ update device list method.

## Device Status Management

This class will handle the following status of a device and update device status to user session.

When deploy mode is centralized, the monitor threads should start when system startup; when deploy mode is standalone, the monitor threads should start when first user login successfully.

|  |  |
| --- | --- |
| **Status Name** | **Conditions** |
| Can Sync Device Configuration | Device connected |
| Can View Device Configuration | Device connected |
| Can Retrieve Log | Device connected |
| Can View Saved Log | There are saved log in Agent |
| Can Register Software | There are missing package and user has rights and trained |
| Can Update Latest Software | Device connected. There is registered latest software for updating. User has rights and trained. |
| Can Update Specific Software | Device connected. There is registered software for updating. User has rights and trained. |
| Can Schedule Update Latest Software | There is registered latest software for updating. User has rights and trained. |
| Can Schedule Update Specific Software | There is registered software for updating. User has rights and trained. |
| Can Download Feature License | GDMP Server has latest feature license available for device. User is trained. |
| Can Update Feature License | Device connected. GDMP has latest feature license available for device. User is trained. |
| Can View Features | Device connected. |

### Class Diagram



### Functions

#### Subscribe

* Description

This function will add a user session to subscriber list. Every time a device status changed, Device Status Manager will update all subscribed user session with new device status.

#### Unsubscribe

* Description

This function will remove a user session from subscriber list.

#### Add Devices

* Workflow



* Description

This function will push the devices to an internal queue; the InitJobThread will pop them one by one and perform device dock for it.

#### Init Job Thread

* Workflow



* Workflow of sub function HandleCentralized



* Workflow of sub function HandleStandalone



* Description

This function will pop a device from internal queue and do the stat device, create device, sync configuration, register software and download feature license job.

The initialize job is different for Centralized mode and Standalone mode. Please see the workflow chart for detail.

#### Status Monitor Thread

* Workflow



* Description

This thread will keep running to check the status for each device, if any device’s any status changed, it will call all subscriber’s update method.

* Sub function CanSyncCfg

This function will get the connection status of the device from Device List Manager directly.

All user session will share same status.



* Sub function CanViewCfg

This function will get the connection status of the device from Device List Manager directly.

All user session will share same status.



* Sub function CanRetrieveLog

This function will get the connection status of the device from Device List Manager directly.

All user session will share same status.



* Sub function CanViewSavedLog

This function will ask GDMP Agent if there is any uploaded log file in its cache.

All user session will share same status.



* Sub function CanRegisterSoftware

This function will get registered package info from Vital Sync, get available software list from GDMP and figure out if there is any software available but not registered to Vital Sync yet.

Different user rights will have different result, so it will do it one by one.



* Sub function CanUpdLatestSoftware

This function will get registered package info from Vital Sync, get installed package info from Vital Sync and figure out if the latest software registered but not installed yet. User should have certain rights to performance the update action.



* Sub function CanUpdSpecificSoftware

This function will get registered package info from Vital Sync, get installed package info from Vital Sync and figure out if there is specific software registered but not installed yet. User should have certain rights to performance the update action.



* Sub function CanSchUpdLatestSoftware

This function has same workflow with CanUpdLatestSoftware exception it doesn’t require that device is connected.

* Sub function CanSchUpdSpecificSoftware

This function has same workflow with CanUpdSpecificSoftware exception it doesn’t require that device is connected.

* Sub function CanDownloadFeatureLic

This function will ask GMDP if there is new feature license on server for downloading.



* Sub function CanUpdateFeatureLic

This function will ask GMDP if there is new feature license downloaded for updating.



* Sub function CanViewFeatureLic

This function will get the connection status of the device from Device List Manager directly.



## Device Configuration Management

### Class Diagram



### Functions

#### Sync Device Configuration

* Workflow



* Description

This function will get device running configuration from Vital Sync and then send it to GDMP.

#### View Device Configuration

* Workflow



* Description

This function will get device running configuration from Vital Sync and then return the configuration to client.

## Device Log Management

### Class Diagram



### Functions

#### Sync Device Log

* Workflow



* Description

This function will request device log from Vital Sync, once Vital Sync return the log file URI, [Data Operation Service](#_Transfer_Log_File) will step in and take the responsibility to transfer the log from Vital Sync to GDMP by chunking.

#### View Saved Device Log

* Workflow



* Description

This function will request saved device log from GDMP, once GDMP return the log file URI, [Data Operation Service](#_Data_Operation_Service) will step in and take the responsibility to retrieve the saved log from GDMP by chunking.

## Software Management

### Class Diagram



### Functions

#### Register Software

* Workflow



* Description

This function will get software list from GDMP and installed package list from Vital Sync. If there is any missing software, register them one by one. [Data Operation Service](#_Transfer_Software_Binary) will be response for transferring the software binary from GDMP to Vital Sync.

#### Update Latest Software

* Workflow



* Description

This function will get registered software list from Vital Sync and installed package list from Vital Sync. If there is any latest software not installed, update it.

It will send a acknowledge message to GDMP to inform the update when update start. When it finished or failed, it will send a completed or failed message to GDMP to inform the result.

If update software successfully, it will sync configuration for device.

#### Update Specific Software

* Workflow



* Description

This function will get installed package list from Vital Sync. If the specific software is not installed and user has the rights, then update it.

It will send a acknowledge message to GDMP to inform that update starts. When it finished or failed, it will send a completed or failed message to GDMP to inform the result.

If update software successfully, it will sync configuration for device.

#### Get Installed Packages

* Workflow



* Description

There is no Vital Sync API for getting installed package list. This function will get the running configuration from Vital Sync for a device and extract the installed software list from the running configuration.

## Feature License Management

### Class Diagram



### Functions

#### Download Feature License

* Workflow



* Description

This function will ask Agent to download latest Feature License File from GDMP Server and cache it on Agent.

#### Update Feature License

* Workflow



* Description

This function will download the Feature License File from Agent in memory and update device with it. [Data Operation Service](#_Transfer_Feature_License) will be response for getting the Feature License File from GDMP and keep it in memory.

#### View Feature License

* Workflow



* Description

This function will get the Feature License Information from Vital Sync and get latest Feature License File from GDMP. [Data Operation Service](#_Data_Operation_Service) will be response for getting the Feature License File from GDMP and keep it in memory.

GDMP doesn’t want to know the file structure of Feature License File, so DMM has to parse it.

#### Feature License Matrix

* Workflow



* Description

This function will get the device list and filter it by target device type, then get the feature license information one by one.

#### Get Feature Info

* Workflow



* Description

This function will get the Feature License Info from Vital Sync for a device.

## Scheduler Management

### Class Diagram



### Functions

#### Schedule Device Log

* Workflow



* Description

This function will schedule a task to retrieve device log in Vital Sync and save the request ID in local request ID file. CC Timer will use the request ID to query the result of schedule task.

#### Schedule Update Latest Software

* Workflow



* Description

This function will schedule a task to update latest software for a device in Vital Sync and CC Timer will trigger to get the update result. If update successfully, CC Timer will sync device configuration.

#### Schedule Update Specific Software

* Workflow



* Description

This function will schedule a task to update specific software for a device in Vital Sync and CC Timer will trigger to get the update result. If update successfully, CC Timer will sync device configuration.

#### Schedule Job List

* Workflow



* Description

This function will get all scheduled job from Pending Request ID file.

# UTILITY

## Business Rule Handler

## Scheduler Generator

It will be used to add scheduled task to run CC Timer.



CC Timer could be used to run following tasks (these tasks are pre-defined during CC installation):

* Virtual Dock
* Retrieve Scheduled Device Log
* Update Latest Software
* Update Specific Software
* Timeout Request Re-check

Please see [CC Trigger](#_CC_Trigger) for how task will be handled when time comes.

## Data Protector

This class will leverage Windows Data Protection API to protect data in memory.



ProtectedData is the C# build-in class for DPAPI

|  |  |  |
| --- | --- | --- |
| **Function** | **Parameter** | **Description** |
| Protect | data | The data need protect |
| Return value | Protected data |
| Unprotect | data | The data need unprotected |
| Return value | Unprotected data |

## Log

DMM will use Log4Net to do the log.

# COMMON SERVICE

## Sign On & Entitlement

### User Session

User session will be used to save GDMP session ID, user rights, facilities for user.

User session will also keep device list, device status for user as cache.

It subscribes itself to device list manager and device status manager, provides callback functions to device list manager and device status manager. So when anything changed, device list, device status can be updated for user in user session and corresponding notification will be pushed to clients.

If the deploy mode is standalone, threads of Device List Manager and Device Status Manager should be stopped when user logout or user session timeout.

#### Class Diagram



#### Functions

##### Update Device List

* Workflow



* Description

This is a callback function for device list update. Device List Manager will call this function if device list is changed in system.

This function will generate the device notification based on the new device list provided by Device List Manager.

##### Update Device Status

* Workflow



* Description

This is a callback function for device status update. Device status Manager will call this function if any device status is changed in system.

This function will update one of the Can XXX statuses.

##### Update Alert

* Description

This is a callback function for alerts. Device status Manager will call this function periodically to generate alerts on device list page.

This function will generate notification for how many devices are ready for software update and how many devices are ready for feature license update according to the device status list.

### User Rights

User rights will be generated after user login to Vital Sync successfully. Related classes are in the following diagram,



It will map GDMP user role of CoT to CC user role (see role mapping [here](#_CC_Role_Mapping)), and then generate rights for each device type of the CoT.

CC user role’s rights is defined in following table,

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Class A** | **Class B** | **Class C** |
| **Sync Configuration** | Y | Y | Y |
| **Retrieve Log** | Y | Y | Y |
| **Update Software** | Latest Only (trained device type) | All Release (trained device type) | All and Test (trained device type) |
| **Update Feature License** | Y (trained device type) | Y (trained device type) | Y (trained device type) |

## Resource Management

### CC Role Mapping File

This class will load CC Role mapping file into memory. Given a GDMP role name, it will return the corresponding CC role name.



The content of this file should be:

Application Support = Class A

Biomed = Class A

CoT Admin = Class A

Field Service Technician = Class B

Marketing = Class C

QA = Class C

R&D = Class C

Sales Rep = Class B

Service Center Technician = Class B

Service Manager = Class B

User Admin = NA

The configuration file should be encrypted and should not allow to be changed after installed.

### DMM Configuration File

This class will manage the DMM configuration.



Detail description of class

|  |  |  |
| --- | --- | --- |
| **Function** | **Parameter** | **Description** |
| LoadCfg | fileName | The DMM configuration file. |
| Return value | True, load without error. False, error happened. |
| GetCfg | cfgName | The configure name |
| Return value | The configure value |

CFGException should be handled.

The configuration file will look like:

DeployMode = (Centralized, Standalone)

VitalSyncBaseURL=https://192.168.0.2/

AgentIP=192.168.0.3

AgentPort=12345

DeviceListRefreshInterval=60s

## Data Operation Service

### Class Diagram



### Functions

#### Transfer Log File

* Workflow



* Description

This function will transfer the log file from Vital Sync, to GDMP by chunking.

#### Retrieve Saved Log File

* Workflow



* Description

This function will get the saved log file from GDMP by chunking.

#### Transfer Software Binary File

* Workflow



* Description

This function will transfer the software binary from GDMP to Vital Sync by chunking.

#### Transfer Feature License File

* Workflow



* Description

This function will transfer the feature license file from GDMP to Vital Sync by chunking.

#### Retrieve Feature License File

* Workflow



* Description

This function will retrieve the feature license file from GDMP by chunking.

## Metadata Service

This class will maintain all metadata



* Supported Device Types

Get supported device types list from GDMP and save it for later use.

* Business Rules

Get Business Rules from GDMP for each device type and save them for later use.

# AUDITING TRAIL

Auditing Trail Class will log all user activity in auditing log file.

* Each DMM API should call AuditingTrail.Log before handling the user request.
* Each notification with final result should also call AuditingTrail.Log to log the activity result.



# CC UPDATE

See [Sign on](#_Sign_On)

# CC TIMER

## CC Trigger

This is an executable application that is able to perform the tasks scheduled in Job Scheduler.

* Virtual Dock

This task is created during installation when it’s centralized mode. It will need a user/password for login. It will call the DMM API DeveListVirtualDock instead of DeviceList to perform virtual dock. See [Virtual Dock](#_Get_Devices_Virtual) Workflow.

* Retrieve Scheduled Device Log

When this task runs, it will try to get all schedule sync log request ID from a pending Request ID file. For each request ID, query Vital Sync to see if the log file is ready. If it is, retrieve the file from VS and upload it to GDMP.

* Update Latest Software

When this task runs, it will try to get all schedule latest software update request ID from a pending Request ID file. For each request ID, query Vital Sync to see if the update is done. If it is, send a log update activity to GDMP and remove the ID from the file. If update successfully, CC Trigger should perform sync device configuration.

* Update Specific Software

It’s same with Update Latest Software

* Timeout Request Re-check

This task is to check the result of all timeout requests. So far only update software and update feature license need to be taken care of.

For update software, check the result, send the acknowledge message to GDMP. Perform sync running configuration if the result is success.

For update feature license, it’s same as update software.

# Appendix A

Notifications

* Prep Screen Notification

Notification Type = “PrepScreenNotification”

Notification = {

int totalSoftwareCnt;

int totalFeatureLicenseCnt;

int completedSoftwareCnt;

int completedFeatureLicenseCnt;

bool jobDone;

}

* Software Alert

Notification Type = “SoftwareAlert”

Notification = {

int CanUpdateSoftwareDeviceCount

}

* Feature License Alert

Notification Type = “LicenseAlert”

Notification = {

int CanUpdateLicenseDeviceCount

}

* Device Connection Notification

Notification Type = “DeviceConnectionNotification”

Notification = {

String DeviceType,

String SerialNumber,

Bool Connected

}

* Device Status Notification

Notification Type = “DeviceStatusNotification”

Notification = {

String DeviceType,

String SerialNumber,

Bool CanSyncCfg,

String CanSyncCfgReason,

Bool CanViewCfg,

String CanViewCfgReason,

Bool CanRetrieveLog,

String CanRetrieveLogReason,

Bool CanViewSavedLog,

String CanViewSavedLogReason,

Bool CanRegisterSoftware,

String CanRegisterSoftwareReason,

Bool CanUpdLatestSoftware,

String CanUpdLatestSoftwareReason,

Bool CanUpdSpecificSoftware,

String CanUpdSpecificSoftwareReason,

Bool CanSchUpdLatestSoftware,

String CanSchUpdLatestSoftwareReason,

Bool CanSchUpdSpecificSoftware,

String CanSchUpdSpecificSoftwareReason,

Bool CanDownloadFeature,

String CanDownloadFeatureReason,

Bool CanUpdateFeature,

String CanUpdateFeatureReason,

Bool CanViewFeature,

String CanViewFeatureReason

}

* Sync Device Configuration Notification

Notification Type = “SyncCfgNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber

}

* View Device Configuration Notification

Notification Type = “ViewCfgNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

HWComponent [] components,

SWComponent [] components,

Feature [] features

}

HWComponent = {

Name,

Revision,

Description

}

SWComponent = {

Name,

Version,

Description

}

Feature = {

Name,

Description,

Term,

Status

}

* Sync Device Log Notification

Notification Type = “SyncLogNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

Double Percentage

}

* View Saved Log Notification

Notification Type = “ViewSavedLogNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

String LogName,

String DateTime,

String LogType,

String SoftwareVersion,

String FileVersion,

LogEntry [] entries

}

LogEntry = {

String Date,

String Time,

String Event,

String Code,

String Type,

String Notes

}

* Register Software Notification

Notification Type = “RegisterSoftwareNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

Double Percentage

}

* Update Latest Software Notification

Notification Type = “UpdateLatestSoftwareNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

Double Percentage

}

* Update Specific Software Notification

Notification Type = “UpdateSpecificSoftwareNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

Double Percentage

}

* Download Feature License Notification

Notification Type = “DownloadFeatureLicenseNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber

}

* Update Feature License Notification

Notification Type = “UpdateFeatureLicenseNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber

}

* View Feature License Notification

Notification Type = “ViewFeatureLicenseNotification”

Notification = {

String RespCode,

String RespMsg,

String DeviceType,

String SerialNumber,

Feature [] features

}

Feature = {

Name,

Description,

TermType,

RemainingCnt,

TotalCnt,

TerminationDate,

Status,

PurchaseStatus

}

* Feature License Matrix Notification

Notification Type = “FeatureLicenseMatrixNotification”

Notification = {

String RespCode,

String RespMsg,

String CoT,

String DeviceType,

String SerialNumber,

String SoftwareVersion,

Feature [] features

}

Feature = {

Name,

Status

}

# Appendix B

|  |  |
| --- | --- |
| Response Code | Description |
| 00000 | General Success |
| 00001 | General Request Accepted |
| 00002 | General In Progress |
| 00003 | General Failed |
| 10001 | Wrong VS user name or password |
| 10002 | GDMP login required |
| 10003 | Wrong GDMP user name or password |
| 10004 | No management rights |
| 10005 | Account locked |