**Medtronic MITQ**

**Research & Development**

**High Level Design for**

**Gateway Device Management Platform 4.0**

**RE00062903 Rev** **A**

|  |  |  |  |
| --- | --- | --- | --- |
| Issue Date: Refer to Agile | | | Page 1 of 43 |
|  | | | |
| Revision History: | | | |
| Revision | SDA | Description | |
| A | RC065211 | Initial Release. | |
|  |  | | |
| Author: Refer to Agile for approvals | | | |
| Maverick Zhang – Insigma US (development partner) | | | |
|  | | | |
| **Approvals:** Refer to Agile for approvals | | | |
| Ash Pandey – MITG R&D | | | |
| *Paul Ourada – MITG R&D* | | | |
| *Maureen Richard – MITG R&D* | | | |
| *Ted Regan – MITG R&D* | | | |

Contents

[1. INTRODUCTION 5](#_Toc463955655)

[1.1 Purpose 5](#_Toc463955656)

[1.2 Scope 5](#_Toc463955657)

[2. REFERENCED DOCUMENTS 5](#_Toc463955658)

[3. DEFINITIONS AND ACRONYMS 5](#_Toc463955659)

[4. GDMP REQUIREMENTS HIGHLIGHTS 6](#_Toc463955660)

[5. ASSUMPTION 7](#_Toc463955661)

[6. SYSTEM OVERVIEW 7](#_Toc463955662)

[6.1 GDMP System and Peripheral Systems 7](#_Toc463955663)

[6.2 Primary Data Flows 8](#_Toc463955664)

[7. ARCHITECTURE OVERVIEW 9](#_Toc463955665)

[7.1 High Level Architecture 10](#_Toc463955666)

[7.2 GDMP Server 11](#_Toc463955667)

[7.2.1 Component Overview 11](#_Toc463955668)

[7.2.2 Key Design Consideration 13](#_Toc463955669)

[7.2.3 RESTful API Interfaces 15](#_Toc463955670)

[7.3 GDMP Web 22](#_Toc463955671)

[7.4 GDMP Agent 24](#_Toc463955672)

[7.4.1 Key Design Consideration 25](#_Toc463955673)

[7.4.2 RESTful API Interfaces 26](#_Toc463955674)

[8. DATA MODEL 27](#_Toc463955675)

[8.1 Conceptual View 27](#_Toc463955676)

[8.2 Supplement Data Files 29](#_Toc463955677)

[8.3 Data Sources from External System 31](#_Toc463955678)

[8.4 LDAP Data Structure 31](#_Toc463955679)

[9. SECURITY 33](#_Toc463955680)

[9.1 Authentication, Authorization and Access Control 33](#_Toc463955681)

[9.1.1 Authentication 33](#_Toc463955682)

[9.1.2 Authorization 33](#_Toc463955683)

[9.1.3 Access Control 34](#_Toc463955684)

[9.2 Data Communication 34](#_Toc463955685)

[9.2.1 Communication between Clients and Agent 34](#_Toc463955686)

[9.2.2 Communication between Agent and Server 34](#_Toc463955687)

[9.3 Data Storage 34](#_Toc463955688)

[9.4 Data Privacy 35](#_Toc463955689)

[10. HIGH AVAILABILITY 35](#_Toc463955690)

[10.1 Service Resilience 35](#_Toc463955691)

[10.2 Single Point of Failure and Service Recovery 35](#_Toc463955692)

[10.3 Solution for High Availability 36](#_Toc463955693)

[11. SCALABILITY 36](#_Toc463955694)

[12. DEPLOYMENT AND DISTRIBUTION 36](#_Toc463955695)

[12.1 Monitoring and Hardware Support 38](#_Toc463955696)

[12.2 GDMP Agent Distribution 38](#_Toc463955697)

[12.2.1 Distribution and Key Protection 39](#_Toc463955698)

[13. MAINTENANCE 40](#_Toc463955699)

[13.1 Database Backup and Archiving 40](#_Toc463955700)

[13.2 Hot Fix and Upgrade 40](#_Toc463955701)

[14. DATA MIGRATION 41](#_Toc463955702)

[14.1 Data Category 41](#_Toc463955703)

[14.2 Data Migration Process 41](#_Toc463955704)

[14.2.1 Database Tables to LDAP 41](#_Toc463955705)

[14.2.2 Database Tables to New Database Schema 42](#_Toc463955706)

[14.2.3 Package repository and log files 43](#_Toc463955707)

[15. REQUIRED THIRD PARTY SOFTWARE/COMPONENTS 43](#_Toc463955708)

[16. PRODUCTION RELEASE PROCEDURE 45](#_Toc463955709)

[17. PRODUCTION SUPPORT & ESCLATION PATH 45](#_Toc463955710)

# INTRODUCTION

This chapter introduces purpose and scope of this document.

## Purpose

This document describes high level architecture design of the GDMP 4.0 system. It identifies the architectural components, software items, API interfaces and communication mechanisms between GDMP and peripheral applications.

## Scope

The scope of this document is describing high level architecture overview of Gateway Device Management Platform (GDMP) 4.0.

# REFERENCED DOCUMENTS

| Ref # | Reference Document | Number | Rev |
| --- | --- | --- | --- |
| [1] | R0030891\_ S\_System Requirements\_Specs.docx |  |  |
| [2] | R0054192\_A PLN-PROC DMP Production Support Gateway |  |  |
| [3] | RE00026719\_A System Requirements Specification CCv1.0 |  |  |
| [4] | RE00026722\_A High Level Architecture Document CC v1.0 |  |  |

# DEFINITIONS AND ACRONYMS

| Term or Abbreviation | Description |
| --- | --- |
| GDMP | Gateway Device Management Platform |
| LDAP | Lightweight Directory Access Protocol |
| LB | Load Balance |
| DB | MySQL database |
| ROLE | A set of user privileges to authorized user can use. |
| PUBLIC SCHEMA | Integration point with Enterprise one for exchange data |
| MVC | Module-View-Controller design pattern |
| CoT | Class of Trade |
| SPOF | Single Point of Failure |
| SMTP | Simple Mail Transfer Protocol |
| Legacy Client | Existing client applications, such as VLEX client application |
| VLEX | Valleylab™ Exchange (VLEX) Software Update System |
| SLA | Service Level Agreement |
| API | Application Programming Interface |
| AD | Active Directory Service |
| ETL | Extract, Transform and Load (ETL) refers to a process in database usage and especially in data warehousing that: Extracts data from homogeneous or heterogeneous data sources. Transforms the data for storing it in the proper format or structure for the purposes of querying and analysis. |

# GDMP REQUIREMENTS HIGHLIGHTS

The GDMP – Gateway Device Management Platform provides a secure and easy-to-use platform to manage service activities of Medtronic medical devices. In order to support medical devices management, it provides the following services for supported medical device product lines:

* Provide a managed repository of medical devices software, hardware, configurations, per-device related records and service-related records for each medical device product line.
  + Devices have software, hardware, configuration and correlation defined by multiple factors like device type, software version, firmware version, location and licensing information and so on.
  + GDMP stores medical device software packages and maintains such relationship in an easy-to-manage way.
  + Per-device records like device identification information (Serials Numbers, Country, Region…) and device data (e.g.: log files) are also collected and stored in GDMP.
  + Service-related records like support personnel’s training records, device service records and so on are properly stored in GDMP as well.
* Provide user interface to manage such a repository according to business requirements of each product line. Authorized users can use GDMP to manage the repository according to business needs. It includes but not limited to:
  + Upload new version of medical device software package, documents and manage configuration for specific device type
  + Make queries to GDMP for medical device log files by desired data criteria and download copy of query result
  + View or subscribe to email notifications and alerts on specific medical device types
* Provide application integration point to Medtronic internal applications (like VLEX, Common Client) to support device operation work in applicable circumstances. Medtronic internal application like Common Client has to access to the managed repository GDMP provides, in either online mode or offline mode via a local Agent application.
* Provide necessary reporting and auditing tools for repository management purposes. GDMP keeps audit trail of user activities like device service record (e.g.: software update) and allow authorized users to export such auditing records. Reporting tools generate reports for summarizing current status of devices like software version report, device historical information report and so on.

By leveraging industry-proven mature technologies, GDMP 4.0 shall not only provide a flexible and reliable design with well-organized data, comprehensive support to current business, ideal maintainability and extensibility for future enhancements and business growth, but also provide user friendly interfaces and backward compatibility to support existing users. It must follow all required Medtronic standards on security for a secured platform and provide sufficient service availability and stability to meet 99.9% system uptime SLA. Mechanisms on user/application authentication, authorization and access control have to be designed and implemented. Data communication and storage must have appropriate strong protection approaches applied.

Necessary data in current GDMP system must be migrated to GDMP 4.0 for a clear cut-over.

# ASSUMPTION

GDMP 4.0 will be designed under the following assumptions:

1. GDMP 4.0 will reuse the existing GDMP production infrastructure environments.
2. GDMP Web user will turn on cookies in web browsers.
3. All GDMP servers shall participate in the Corporate IS remote security monitoring infrastructure.
4. A dedicated Medtronic team takes cares of Database backup and recovery.
5. Medtronic Active Directory will provide Web service interface for GDMP to integrate with.
6. Medtronic will provide feature/license generation routine/algorithm to GDMP for generating feature/license file from sales record which stored in PUBLIC SCHEMA.
7. User password will be migrated as-is without asking all GDMP users to reset passwords (in the current GDMP, user password is stored as MD5 hashed string without salt).
8. Disaster Recovery (DR) is not a valid scenario to this project. Hence, the solution is not going to support DR.

# SYSTEM OVERVIEW

This chapter provides an overview to GDMP 4.0 system and peripheral systems and applications.

## GDMP System and Peripheral Systems

GDMP works with several external systems/applications to support client applications used for managing individual medical devices. Those systems and applications include:

* PUBLIC SCHEMA – Medtronic’s data source that holds full set of customer, device, and sales records.
* Common Client – A next generation client application. It is a web based application r for interfacing between Vital Sync platform and GDMP Agent. Common Client is supposed to replace legacy client in the future where device teams are in agreement.
* Legacy Client – All existing clients except Common Client, and those developed for GDMP v4.0. It’s responsible for interfacing between supported medical devices and GDMP Agent. For example, VLEX client is one of legacy clients that manage Valleylab™ LS10 generator.
* Medtronic Active Directory Service – It is Medtronic’s internal directory service that stores Medtronic user data. A web service is provided for integration with GDMP.
* Medtronic SMTP Server – Medtronic’s email server which handles all outgoing e-mail messages from GDMP.



Figure 1 Overview of GDMP with Peripheral Systems

## Primary Data Flows

There are six primary data flows that GDMP and peripheral systems would communicate with.

* Data flow between Legacy Client and GDMP Agent:

For medical device management, Legacy Client connects to GDMP Agent for various operations like checking software versions, uploading medical device logs and so on.

A TCP socket connection is established between Legacy Client and GDMP Agent for communication. All requests and responses are wrapped in XML. In all cases, data connection is initiated by Legacy Client.

* Data flow between Common Client and GDMP Agent:

For medical device management, Common Client connects to GDMP Agent for various operations as well, like checking software versions, uploading medical device logs and so on.

A HTTPS based RESTful API interface is invoked to perform communication between Common Client and GDMP Agent. In general, the communication is initiated by Common Client. All requests and responses are wrapped in XML in the same specification Legacy Client follows.

* Data flow between GDMP users’ web browsers and GDMP Web Server:

Authorized GDMP users can access GDMP Web by modern web browsers to perform management tasks and make queries to device or system information via HTTPS web service.

* Data flow between Public Schema and GDMP Server

GDMP Server will fetch customers, locations, location roles and device data in CSV files via SFTP from Public Schema. Such data are published by “Enterprise Systems” to Public Schema. An ETL process will run and import these data into GDMP database. A modified ETL process will process these input data files and update database of GDMP 4.0.

GDMP Server will publish service events to Public Schema for back end “Enterprise Systems” to consume as needed. This will occur via a similar ETL process as retrieving data from the Public Schema.

NOTE: Public Schema, while an actual entity today, will be used in this document as an alias for “enterprise data repository.” In the near future Public Schema will be retired and the legacy MDT “Data Integration Hub” (DIH) will be the source/sink of this data.

* Data flow between GDMP and Medtronic Active Directory Service

GDMP Server will invoke web service provided by Medtronic Active Directory Service to get specified Medtronic user information.

* E-Mail Notifications

GDMP Server will connect to Medtronic SMTP server to deliver e-mail notifications.

# ARCHITECTURE OVERVIEW

GDMP 4.0 is designed with a multitier architecture including GDMP Front-End Tier, GDMP Core Tier and Storage Tier, GDMP Data Tier. GDMP Front-End Tier consists of GDMP Web and GDMP Agent, while GDMP Core Tier consists of GDMP App Server. GDMP Data Tier consists of the GDMP OLTP/OLAP Database Server(s).

The following sections will address these in detail.

## High Level Architecture

The current GDMP is built based on Drupal framework, along with more and more business needs, the maintenance and supporting cost of the current GDMP increases rapidly. The intent of GDMP 4.0 is to redesign internal component to reduce on-going maintenance and supporting cost, along with enhancement in security and critical system functionalities.

GDMP 4.0 will keep current application interfaces and behavior intact to all existing Legacy Client and focus on renovation of the components inside GDMP system.

GDMP 4.0 will consist of multiple components and services.

* GDMP Web

GDMP Web is a Web application providing web-based user interface to allow end users, such as CoT admin and application support, to use GDMP to conduct medical device management and configuration tasks including hardware/software configuration, report generation, device/user management, and so on.

* GDMP Agent

GDMP Agent is a Java application working as integration point between Common Clients/Legacy Clients and GDMP Server. It is mainly responsible for downloading specified software and document from GDMP Server for clients and devices, and pushing device status information and operation audit information (e.g.: log files) to the GDMP Server. GDMP Agent is designed to support both online (working with an Internet connection to GDMP Server) and offline (working without an Internet connection to GDMP Server) mode.

* GDMP Server

GDMP Server is the core business logic tier to manage hardware, software, document, named configuration, device, reports, alert, and user information. It provides RESTful interfaces to GDMP Web and GDMP Agent.

* Database Server

Inside GDMP, database will store all information except user information, Privileges and ROLEs information.

* LDAP Server

LDAP Server stores all roles & permission information and non-Medtronic users’ profiles. Besides, LDAP Server will provide such roles & permission and user profile information to GDMP Server, GDMP Server will determine how to authorize and control access based on this information.



Figure 2 High Level Architecture

The diagram above illustrates high level components of GDMP 4.0 and how components communicate with each other inside the system.

## GDMP Server

This chapter describes detailed GDMP Server component.

### Component Overview

GDMP Server will be reorganized from individual business workflow centric to system workflow centric. With this changes, most components will be redesigned and new framework will be introduced.

The diagram below shows modularized view of GDMP Server.



Figure 3: GDMP Server

The table below provides a detailed description of each component.

| Component Name | Description |
| --- | --- |
| RESTful Handler | Provide RESTful interfaces to GDMP agents and GDMP Web, and it is built on Jersey framework. |
| Message Parser | Parse XML message to object; or generate XML message from object. It will be built on JAXB. |
| Message Routing | This is a JNDI wrapper. it will route request to the proper services for further processing either to a local service or a remote service (Based on configuration, it will be able to route request other GDMP Servers when supported) |
| Device Service | Manage device related operations and provide detailed device information. |
| Feature License Service | Manage feature license file for each device |
| File Request Service | Manage file uploading and downloading. |
| Configuration Service | Manage all configuration (Software/Hardware/System Configuration) for each device type. |
| User Service | Authenticate user and provide user login information, such as user permissions, training records, and user access privileges. |
| Access Control Service | Determine functions and features a user is authorized to use as per user permission information fetched from user login information, |
| Report Service | Provide reports according to user specified data criteria. |
| Alert Service | Manage alert message templates, notifications and alert subscriptions. |
| Trade Embargo Service | Manage software embargo control |
| Auditing | Provide an aggregated audit trail of operations including authentication events, access control administration, demographic operations, device operations and notification/alert events. |
| Exception Handler | Provide an exception class hierarchy for handling unexpected exceptions that are thrown during the runtime. |
| Process Workflow Management Framework | Manage business process workflow based on configuration.  Business process will be reorganized and packaged into atomic java classes, and these classes will be organized and invoked based on a pre-configured workflow. |
| Data Validation | Define how to verify the integrity of the data when receiving data from outside. |
| Configuration | Load and manage all configuration files. |
| Crypto Functions | Handle all encryption/decryption functions |
| DB | Manage access to database. |
| File | Manage access to file system. |
| LDAP | Manage LDAP operations. |
| Email | Maintain an email sending job queue and send emails to Medtronic SMTP server |
| AD | Manage data interaction with Active Directory service |

### Key Design Consideration

Key points in the design of GDMP 4.0 are explained in the following sections.

#### Session Management between GDMP Web and Server

The GDMP 4.0 will be designed to support both High availability and load balance cluster. In order to achieve that objective, the Session management of the GDMP 4.0 will be designed to use Session-In-Cookie. The data in the cookie will be encrypted and encoded.

Session processing workflows are defined as follow:

* Process inbound request:
* Get the session cookie from request
* Decrypt the cookie with specified secret key by using standard AES encryption algorithm
* De-serialize the decrypted value into a session object.
* Process outbound response:
* Serialize the session into a cookie
* Encrypt the cookie with specified secret key
* Attach encrypted cookie into response

A session usually has a session timeout to expire the session after certain amount of inactivity. There is also a security-relevant reason for such a timeout: to limit the attackers. To achieve this goal, GDMP Server records original login time for each session. Each time GDMP Server receives the request, original login time will be checked and the session will be expired according to pre-configured max session timeout and session idle time based on last received request time. If a session is expired, user has to log in again by using user name and password.

#### Inversion of Control for Fine-Grained Business Workflow by Medical Device Type

GDMP supports multiple medical devices products and types. Business requirements vary by product and GDMP must implement different business workflows respectively while part of business logic could be shared across different devices. In current GDMP, business workflows are controlled by cascaded switches and if-else statements by device type. It results in redundant source codes, less readable and less maintainable.

In GDMP 4.0, business workflows will be highly abstracted as multiple logic steps. Actual implementation of each logic step may vary by product line or device type. To fulfill a business workflow for a specific device type, required logic steps are instantiated and assembled by reading meta-data configuration which describes actual implementations of such business logic step to use. Common logic steps can be shared. When additional business processing steps or enhancements are necessary, new logic steps will be developed and configured to get invoked in relevant workflows only, without major impact to the overall processing workflow. Related functional testing and impact to irrelevant device product or device type become more manageable.

#### Non-session based communication between GDMP Agent and Server

The communication between GDMP Agent and GDMP Server will be totally stateless; no more session exists between Agent and Server. Each request from GDMP Agent will convey user email and MD5 hashed password string (Given the fact that MD5 is already used in current system and user password already saved as MD5 hash string in database. New system won’t be able to migrate from MD5 to other checksum algorithm).

#### Exception Management

A unified exception handling mechanism will be introduced into the GDMP 4.0. And the pattern below will be followed throughout the entire development lifecycle of the project.

* Capture details

The more detail captured when an error occurs; the easier it will be to resolve the issue. Besides, log all of the parameters to the service.

* Log errors only at service boundaries

Log the error just before sending the fault exception back to the client.

* Associate unique exception event IDs

A unique event ID will be associated with a specific exception type.

* Exception message must be meaningful and formatted well.

The purpose of exception message is to help troubleshoot and make supporting effort easier.

#### On-boarding new device into GDMP system

On boarding a new device into GDMP system will actually causes 2 things change inside the GDMP system, metadata and device specific workflow.

Metadata, it includes CoT, device type those one-time created data, GDMP system will leverage systematic approach for those data setup according to the following steps:

1. Create an SQL to populate CoT and Device when it needs
2. Put them into a special folder (says “initjob”)
3. Package the folder and SQL files into WAR
4. Push the WAR to Chief
5. Chief will deploy the new WAR to the destination environment (QA, for example)
6. QA will reboot the GDMP application
7. When GDMP application starts, it will check is there any SQL command should be executed in “initjob” folder
8. Destroy the “initob” folder and all its content after #7 success. And log all these execution into log file

Device specific workflow, it is a device specific logic when manipulating the request for a given device. We will abstract these existed business rules and create a set of atomic business rules (as plug-in component), and build a business rules engine into the GDMP system which is a configuration driven sub-system, and all the individual pluggable atomic business rule will be invoked based on the configuration. These approach will minimize the impact when new business rules adding into the system and maximizing the reusability.

### RESTful API Interfaces

There are two sets of RESTful API interfaces on GDMP Server. One is for GDMP Web and the other is for GDMP Agent.

#### API Interface for GDMP Agent

The table below enumerates RESTful API interface provided for GDMP Agent, including functions for GDMP server check, login/logout, download/upload, general request processing and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/agent/api/users/login | GET | User login from Client App |
| /medtronic/rest/agent/api/users/logoff | GET | User logoff from Client App |
| /medtronic/rest/agent/api/users/permission | POST | Agent resync user permission |
| /medtronic/rest/agent/api/config/hwsw | POST | Agent resync HW/SW configuration |
| /medtronic/rest/agent/api/config/named/{device\_type} | GET | Get named configuration of device type |
| /medtronic/rest/agent/api/device/stat2 | POST | Stat device for 2.0 device |
| /medtronic/rest/agent/api/device/stat3 | POST | Stat device for 3.0 device |
| /medtronic/rest/agent/api/device | PUT | Create device |
| /medtronic/rest/agent/api/regurl | GET | Get register URL |
| /medtronic/rest/agent/api/forgotpassword | POST | Reset password |
| /medtronic/rest/agent/api/changepassword | POST | Change password |
| /medtronic/rest/agent/api/snreprogram | POST | Device SN re-programming |
| /medtronic/rest/agent/api/auditlog | PUT | Create audit log for software download |
| /medtronic/rest/agent/api/clientupgrade | PUT | Create audit log for Agent/Client upgrade |
| /medtronic/rest/agent/api/device/config2 | POST | Sync device config for 2.0 device |
| /medtronic/rest/agent/api/device/config3 | POST | Sync device config for 3.0 device |
| /medtronic/rest/agent/api/clientapp | POST | Check client app new version |
| /medtronic/rest/agent/api/device/software/upgrade | POST | Device software upgrade acknowledge |
| /medtronic/rest/agent/api/device/license/upgrade | POST | Device license upgrade acknowledge |
| /medtronic/rest/agent/api/connection/check | GET | Check if server is up |
| /medtronic/rest/agent/api/device/log/upload | POST | Upload device log file to Server |
| /medtronic/rest/agent/api/certificate | GET | Get permanent certificate for agent |
| /medtronic/rest/agent/api/countries | GET | Get all countries |
| /medtronic/rest/agent/api/facilities | GET | Get all facilities that associated to user |
| /medtronic/rest/agent/api/facility/{facility\_id}/devices | GET | Get all devices that associated to the facility |
| /medtronic/rest/agent/api/devicetypes | GET | Get all device types that facility actually have devices. |
| /medtronic/rest/agent/api/{device\_type}/software | GET | Get all software that associated to the device type |
| /medtronic/rest/agent/api/{device\_type}/{serial\_number}/license | GET | Get latest feature license of a device |
| /medtronic/rest/agent/api/device/licenses | GET | Get latest feature licenses of facilities |

#### API Interface for GDMP Web

To fulfill GDMP Web’s functionalities and features, GDMP Server provides multiple sets of RESTful API interfaces.

##### **Customer and user related API**

The table below enumerates RESTful API interface provided for GDMP Web on managing customers and users, including functions to adding/modifying/deleting customer/user, customer/user related information inquiry and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/customers | POST | Create new customer |
| /medtronic/rest/web/api/customers/{customer\_id} | PUT | Update customer information |
| /medtronic/rest/web/api/customers/{customer\_id} | GET | Get single customer information |
| /medtronic/rest/web/api/customers | GET | Get all customers |
| /medtronic/rest/web/api/customers/{customer\_id} | DELETE | Delete a customer |
| /medtronic/rest/web/api/facilities | POST | Create new facility |
| /medtronic/rest/web/api/facilities/{facility\_id} | PUT | Update facility information |
| /medtronic/rest/web/api/facilities/{facility\_id} | GET | Get single facility information |
| /medtronic/rest/web/api/facilities | GET | Get all facilities |
| /medtronic/rest/web/api/facilities/{facility\_id} | DELETE | Delete a facility |
| /medtronic/rest/web/api/users/login | POST | User login |
| /medtronic/rest/web/api/users/logoff | POST | User logoff |
| /medtronic/rest/web/api/users/forgotpassword | POST | User forgot password |
| /medtronic/rest/web/api/users | POST | Create new user |
| /medtronic/rest/web/api/users/{user\_id} | PUT | Update user information |
| /medtronic/rest/web/api/users/{user\_id} | GET | Get single user information |
| /medtronic/rest/web/api/users | GET | Get all users |
| /medtronic/rest/web/api/users/{user\_id} | DELETE | Delete a user |
| /medtronic/rest/web/api/users/{user\_id}/facilities | GET | Get facilities that associate with a specific user |
| /medtronic/rest/web/api/customers/{customer\_id}/users | GET | Get users that belong to a specific customer |
| /medtronic/rest/web/api/customers/{customer\_id}/facilities | GET | Get facilities that belong to a specific customer |
| /medtronic/rest/web/api/usertypes | GET | Get user type list (Covidien or Non-Covidien) |
| /medtronic/rest/web/api/userstatus | GET | Get user status list (Locked, Deactivated, etc.) |

##### CoT and device type related API

The table below enumerates RESTful API interface provided for GDMP Web on CoT and device type management, including functions to adding/updating/deleting CoT/device type, CoT/device type related information inquiry and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/cot | POST | Create new CoT |
| /medtronic/rest/web/api/cot/{CoT\_id} | PUT | Update CoT information |
| /medtronic/rest/web/api/cot/{CoT\_id} | GET | Get single CoT information |
| /medtronic/rest/web/api/cot | GET | Get CoT list |
| /medtronic/rest/web/api/cot/{CoT\_id} | DELETE | Delete a CoT |
| /medtronic/rest/web/api/devicetypes | POST | Create new device type |
| /medtronic/rest/web/api/devicetypes/{devicetype\_id} | PUT | Update device type information |
| /medtronic/rest/web/api/devicetypes/{devicetype\_id} | GET | Get single device type information |
| /medtronic/rest/web/api/devicetypes | GET | Get all device types |
| /medtronic/rest/web/api/devicetypes/{devicetype\_id} | DELETE | Delete a device type |
| /medtronic/rest/web/api/cot/{CoT\_id}/devicetypes | GET | Get device types that belong to a specific CoT |

##### User training related API

The table below enumerates RESTful API interface provided for GDMP Web on user training records management.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/trainingrecords | POST | Create new training record |
| /medtronic/rest/web/api/trainingrecords/{record\_id} | PUT | Update a training record |
| /medtronic/rest/web/api/trainingrecords/{record\_id} | GET | Get single training record |
| /medtronic/rest/web/api/users/{user\_id}/trainingrecords | GET | Get all training records of a user |
| /medtronic/rest/web/api/trainingrecords/{record\_id} | DELETE | Delete a training record |
| /medtronic/rest/web/api/trainingstatuses | GET | Get training statuses (Certificated, Uncertificated) |
| /medtronic/rest/web/api/trainers | POST | Create new trainer |
| /medtronic/rest/web/api/trainers/{trainer\_id} | PUT | Update a trainer information |
| /medtronic/rest/web/api/trainers/{trainer\_id} | GET | Get a trainer information |
| /medtronic/rest/web/api/devicetypes/{type\_id}/trainers | GET | Get all trainers of a device type |
| /medtronic/rest/web/api/trainers/{trainer\_id} | DELETE | Delete a trainer |
| /medtronic/rest/web/api/trainerstatuses | GET | Get trainer status (Authorized, Unauthorized) |

##### User role related API

The table below enumerates RESTful API provided for GDMP Web to manage user roles including functions to create/update/delete a role, role related information inquiry and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/roles | POST | Create new role |
| /medtronic/rest/web/api/roles/{role\_id} | PUT | Update a role |
| /medtronic/rest/web/api/roles/{role\_id} | GET | Get single role information |
| /medtronic/rest/web/api/roles | GET | Get all roles |
| /medtronic/rest/web/api/cot/{CoT\_id}/roles | GET | Get all roles that belong to a specific CoT |
| /medtronic/rest/web/api/roles/{role\_id} | DELETE | Delete a role |
| /medtronic/rest/web/api/userroles | POST | Create new user role relationship with access type |
| /medtronic/rest/web/api/userroles/{userroles\_id} | PUT | Update a user role relationship |
| /medtronic/rest/web/api/userroles/{userroles\_id} | GET | Get a user role relationship |
| /medtronic/rest/web/api/user/{user\_id}/userroles | GET | Get roles of a specific user |
| /medtronic/rest/web/api/userroles/{userroles\_id} | DELETE | Delete a user role relationship |
| /medtronic/rest/web/api/accesstypes | GET | Get access types (None, Latest Only, All Release) |

##### Device related API

The table below enumerates RESTful API interface provided for GDMP Web to manage devices including functions to retrieve device related information like device information, service history, hardware/software items, log files and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/devices/{device\_id} | GET | Get single device information |
| /medtronic/rest/web/api/devices | GET | Get all devices |
| /medtronic/rest/web/api/devicetypes/{type\_id}/devices | GET | Get all devices of a device type |
| /medtronic/rest/web/api/devices/{device\_id}/services | GET | Get all service history of a specific device |
| /medtronic/rest/web/api/devices/{device\_id}/hardwares | GET | Get hardware items of a specific device |
| /medtronic/rest/web/api/devices/{device\_id}/softwares | GET | Get software items of a specific device |
| /medtronic/rest/web/api/devices/{device\_id}/logs | GET | Get log file list of a specific device |
| /medtronic/rest/web/api/devices/{device\_id}/discrepancies | GET | Get discrepancy list of a specific device |

##### Hardware, software and document related API

The table below enumerates RESTful API interface provided for GDMP Web to manage hardware, software and documents including functions to create/update/delete hardware/software/document, related regulatory information inquiry and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/hardwares | POST | Create new hardware |
| /medtronic/rest/web/api/hardwares/{hardware\_id} | PUT | Update a hardware |
| /medtronic/rest/web/api/hardwares/{hardware\_id} | GET | Get single hardware information |
| /medtronic/rest/web/api/hardwares | GET | Get all hardware |
| /medtronic/rest/web/api/devicetypes/{type\_id}/hardwares | GET | Get all hardware of a device type |
| /medtronic/rest/web/api/hardwares/{hardware\_id} | DELETE | Delete a hardware |
| /medtronic/rest/web/api/hardwaretypes | GET | Get hardware types (MOTHERBOARD, PCB) |
| /medtronic/rest/web/api/softwares | POST | Create new software |
| /medtronic/rest/web/api/softwares/{software\_id} | PUT | Update a software |
| /medtronic/rest/web/api/softwares/{software\_id} | GET | Get single software information |
| /medtronic/rest/web/api/softwares | GET | Get all software |
| /medtronic/rest/web/api/devicetypes/{type\_id}/softwares | GET | Get all software of a device type |
| /medtronic/rest/web/api/softwares/{software\_id} | DELETE | Delete a software |
| /medtronic/rest/web/api/softwaretypes | GET | Get software types (software, business rule, etc.) |
| /medtronic/rest/web/api/documents | POST | Create new document |
| /medtronic/rest/web/api/documents/{document\_id} | PUT | Update a document |
| /medtronic/rest/web/api/documents/{document\_id} | GET | Get single document information |
| /medtronic/rest/web/api/documents | GET | Get all documents |
| /medtronic/rest/web/api/devicetypes/{type\_id}/documents | GET | Get all documents of a device type |
| /medtronic/rest/web/api/documents/{document\_id} | DELETE | Delete a document |
| /medtronic/rest/web/api/documenttypes | GET | Get document types (Release notes, User Guide, etc.) |
| /medtronic/rest/web/api/softwareregulatory | POST | Create new software regulatory |
| /medtronic/rest/web/api/softwareregulatory/{ id} | PUT | Update a software regulatory |
| /medtronic/rest/web/api/softwareregulatory/{ id} | GET | Get single software regulatory |
| /medtronic/rest/web/api/softwares/{software\_id}/ softwareregulatory | GET | Get all software regulatory of a software |
| /medtronic/rest/web/api/softwareregulatory/{ id} | DELETE | Delete a software regulatory |

##### Trade embargo related API

The table below enumerates RESTful API interface provided for GDMP Web to manage trade embargo settings including functions to list/create/delete embargo country/embargo exception and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/embargo/countries | POST | Create new embargo country |
| /medtronic/rest/web/api/embargo/countries | GET | Get all embargo countries |
| /medtronic/rest/web/api/embargo/countries/{id} | GET | Get single embargo country |
| /medtronic/rest/web/api/embargo/countries/{id} | DELETE | Delete an embargo country |
| /medtronic/rest/web/api/embargo/exceptions | POST | Create new embargo exception |
| /medtronic/rest/web/api/embargo/exceptions | GET | Get all embargo exceptions |
| /medtronic/rest/web/api/embargo/exceptions/{id} | GET | Get single embargo exception |
| /medtronic/rest/web/api/embargo/exceptions/{id} | DELETE | Delete an embargo exception |

##### Named configuration related API

The table below enumerates RESTful API interface provided for GDMP Web to manage named configuration including functions to list configuration types, create/update/delete a named configuration and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/configtypes | GET | Get config types (Named System Config, Named Hardware Config, etc.) |
| /medtronic/rest/web/api/configs/{config\_type} | POST | Create a new named config by config type |
| /medtronic/rest/web/api/configs/{config\_type}/{config\_id} | PUT | Update a named config by config type |
| /medtronic/rest/web/api/configs/{config\_type}/{config\_id} | GET | Get single named config by config type |
| /medtronic/rest/web/api/configs/{config\_type} | GET | Get all named config of config type |
| /medtronic/rest/web/api/devicetypes/{type\_id}/configs/ {config\_type} | GET | Get all named config by config type and device type |
| /medtronic/rest/web/api/configs/{config\_type}/{config\_id} | DELETE | Delete a named config of config type |

##### Alert related API

The table below enumerates RESTful API interface provided for GDMP Web to manage GDMP alert event, message template, related alert events subscriptions and delivery, and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/alert/generaleventtypes | GET | Get general event types (System Upgrade, General Notice, etc.) |
| /medtronic/rest/web/api/alert/generals | POST | Create a general alert event |
| /medtronic/rest/web/api/alert/generals/{id} | PUT | Update a general alert event |
| /medtronic/rest/web/api/alert/generals/{id} | GET | Get single general alert event |
| /medtronic/rest/web/api/alert/generals | GET | Get all general alert event |
| /medtronic/rest/web/api/alert/generals/{id} | DELETE | Delete a general alert event |
| /medtronic/rest/web/api/alert/generals/history | GET | Get all occurred general notification |
| /medtronic/rest/web/api/alert/categories | GET | Get alert categories (Business Alert, Information Alert) |
| /medtronic/rest/web/api/alert/eventtypes/{category\_id} | GET | Get alert event types by category |
| /medtronic/rest/web/api/alert/templates | POST | Create a message template |
| /medtronic/rest/web/api/alert/templates/{id} | PUT | Update a message template |
| /medtronic/rest/web/api/alert/templates/{id} | GET | Get a message template |
| /medtronic/rest/web/api/alert/templates | GET | Get all message template |
| /medtronic/rest/web/api/alert/templates/{id} | DELETE | Delete a message template |
| /medtronic/rest/web/api/alert/{devicetype\_id}/{category\_id}/admin/ events | PUT | Enable/Disable events by device type and event category |
| /medtronic/rest/web/api/alert/{devicetype\_id}/{eventtype\_id}/admin/ subscriptions | PUT | Set up event subscriptions for role by device type and event type |
| /medtronic/rest/web/api/alert/{devicetype\_id}/{category\_id}/user/ subscriptions | PUT | Set up event subscriptions for user by device type and event category |
| /medtronic/rest/web/api/alert/notificationstatuses | GET | Get all notification status (Pending, Sent, Duplicated, etc.) |
| /medtronic/rest/web/api/alert/notifications | GET | Get all occurred notifications |
| /medtronic/rest/web/api/alert/notifications/{id} | GET | Get a notification detail |
| /medtronic/rest/web/api/alert/notifications/{id} | PUT | Update a notification detail |

##### Reporting related API

The table below enumerates RESTful API interface provided for GDMP Web to generate various reports on current configuration, history configuration, country change, device component discrepancy, software upgrade/audit/version, service record, audit trail, training, named configuration and so on.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/reporting/device/currentconfig | GET | Device current configuration report |
| /medtronic/rest/web/api/reporting/device/historyconfig | GET | Device history configuration report |
| /medtronic/rest/web/api/reporting/device/countrychange | GET | Device country change report |
| /medtronic/rest/web/api/reporting/device/discrepancy | GET | Device Component discrepancy report |
| /medtronic/rest/web/api/reporting/software/upgrade | GET | Software upgrade report |
| /medtronic/rest/web/api/reporting/software/aduit | GET | Software audit report |
| /medtronic/rest/web/api/reporting/software/version | GET | Software version report |
| /medtronic/rest/web/api/reporting/servicerecords | GET | Service records report |
| /medtronic/rest/web/api/reporting/aduittrail | GET | Audit trail report |
| /medtronic/rest/web/api/reporting/training | GET | Training report |
| /medtronic/rest/web/api/reporting/namedconfig | GET | Named configuration report |

##### User activity related API

The table below enumerates RESTful API interface provided for GDMP Web to get user activities including functions to list activity types and activities of all user or specific user.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/activitytypes | GET | Get all activity types (Login, Logoff, etc.) |
| /medtronic/rest/web/api/activities | GET | Get all user activities |
| /medtronic/rest/web/api/users/{user\_id}/activities | GET | Get all activities of a user |

##### Other API

Several other RESTful API functions are also provided to GDMP web to get information including functions to list countries, languages, component status and get GDMP Agent Certificates.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /medtronic/rest/web/api/countries | GET | Get country list |
| /medtronic/rest/web/api/languages | GET | Get language list |
| /medtronic/rest/web/api/componentstatus | GET | Get component status (In Production, Limited Release, etc.) |
| /medtronic/rest/web/cert/requestCert | GET | Get the GDMP Agent certificate. |

## GDMP Web

GDMP Web is a GUI based web interface which allows users to use GDMP management features. It will utilize HTML5 and Bootstrap for visual effects, and AngularJS to render dynamic views in a Single Page Application pattern with data fetched from GDMP Server.

The diagram below shows the details of the modularization of the GDMP Web.



Figure 4: GDMP Web

* **Hardware Catalog:** it provides hardware list for each device type, and user can create new hardware and edit hardware.
* **Software Catalog:** it provides software list for each device type, and user can create new software and edit software.
* **Document Catalog:** it provides document list for each device type, and user can create new document and edit document.
* **Configuration Management:** it provides named configuration list for each device type, and user can create new configuration and edit configuration.
* **User Management:** it lists user account information, such as user PERMISSIONs, and training records.
* **Report:** it provides an aggregated result for specified criteria.
* **Device:** itprovides device list for each device type and provides detailed device information for each device.
* **Feature License:** itmanages feature license file for each device.
* **Trade Embargo:** it manages trade embargo country list and exception list.
* **Alert:** it manages message templates, notifications and alert subscriptions.
* **Device Type Management**: it manages device types.
* **CoT Management:** it manages class of trades.

## GDMP Agent

GDMP Agent is the integration point between GDMP server and Legacy Client/Common Client. All client applications must communicate with GDMP Server via GDMP Agent. It will provide two sets of interface - one is TCP socket interface used by legacy clients for backward compatibility, and the other is RESTful API interface for new Common Client. It will support all the features which have already been implemented in current GDMP Agent.

The following diagram shows an overview of GDMP Agent components.



Figure 5: GDMP Agent

The following table shows a description of each component in GDMP Agent.

|  |  |
| --- | --- |
| Component Name | Description |
| RESTful Services | For Common Client only and it is the bridge between Common Client and Netty Socket Channel. |
| Netty Socket Channel | Provide services to legacy clients via socket. It will be invoked by RESTful Services too. |
| Request Dispatcher | Route requests to request handler by request type |
| Request Handlers | A set of handlers each will take care of one kind of request |
| Message Parser | Parse xml message to object; or generate XML message from object |
| Crypto Functions | Handle all encryption/decryption job |
| Session Management | For compatibility purpose, agent will keep session between client and agent, but there will be no session between GDMP Agent and GDMP Server |
| Login Management | Manage user login information, such as user PERMISSIONs, training records |
| Download Management | Manage file downloading includes software, documents and feature license. |
| Upload Management | Manage file uploading; only log files for now |
| Device Management | Manage devices locally |
| Device Configuration Management | Manage configuration of device type; includes hardware configuration, software configuration and named configuration |
| Mailbox Management | Manage notifications for clients; save the requests in out box when server is offline temporarily |
| Business Rules Management | Manage business rule files, such as prep step and trade embargo |
| Cached File Management | Manage downloaded files, extracted files and temporary files |
| Resender | Resend messages that have been saved in out box when GDMP Agent is working in offline mode |
| Connection Checker | Check if GDMP Server is reachable or not. Resynchronize user login information when GDMP Server is reachable. |
| Purge | Clean up Agent such as timeout session, temporary files during upload, mailbox and expired files in cache |
| Boot Loader | Monitor Agent service and perform Agent upgrade when there is a new version available |
| Message Service | Provide interface between GDMP Agent and GDMP Server |
| Proxy | Take care of network proxy if needed |

### Key Design Consideration

#### Session-less communication

The major design consideration is to simplify the complexity of communication mechanism between GDMP Agent and GDMP Server by removing session between them.

Current GDMP uses session for communication, causing unexpected results when GDMP Agent online/offline modes are flapping in a very short period and/or in an instable network environment.

In the GDMP 4.0, all communication between GDMP Agent and GDMP Server will not be session-based. Instead, an additional internal user identification information, user email and MD5-hashed password strings, will be attached to each request. GDMP Server will authenticate each request.

GDMP Agent will use a built-in tool on Microsoft Windows platforms named ‘certutil’ to import GDMP Agent’s certificates to Windows certificates storage and disable the export option.

#### Handle password expiration

Agent will always be forced to get the latest update of user profile (not just the password, but also the permission) and metadata (password policy, maximum retry count) periodically. So, the Agent has enough knowledge to handle the password expiration situation when user log in at offline, even the Agent never backs to online for a period longer than the password expiration duration.

### RESTful API Interfaces

The following table shows RESTful APIs of GDMP Agent.

|  |  |  |
| --- | --- | --- |
| API End Point | HTTP Method | Purpose |
| /rest/Agent/GetUserPermission | GET | Return user’s PERMISSIONs |
| /rest/Agent/GetServerStatus | GET | Return GDMP Server is reachable or not |
| /rest/Agent/StatDevice | GET | Check if device exists in GDMP |
| /rest/Agent/CreateDevice | POST | Create device in GDMP |
| /rest/Agent/UpdateAcknowledge | PUT | Send software, license file update acknowledge to GDMP |
| /rest/Agent/GetCountryList | GET | Get country list from GDMP |
| /rest/Agent/GetFacilityList | GET | Get facilities that associated with a specific user |
| /rest/Agent/GetDeviceList | GET | Get all devices of a facility |
| /rest/Agent/GetDeviceTypes | GET | Get all supported device types of GDMP |
| /rest/Agent/GetSoftwares | GET | Get all software by device type |
| /rest/Agent/DownloadSoftware | GET | Request Agent to download the specific software from GDMP Server. |
| /rest/Agent/DownloadSavedLogFile | GET | Get URL of latest saved log file of a device in Agent cache |
| /rest/Agent/DownloadFeatureLicense | POST | Request Agent to download the feature license files from GDMP Server |
| /rest/Agent/GetCachedFeatureLicense | GET | Get the feature license files from GDMP Agent cache |
| /rest/Agent/DownloadFileInit | GET | Get chunk count of target file |
| /rest/Agent/DownloadFileChunk | GET | Download  the specific file chunk of target file |
| /rest/Agent/UploadFileInit | POST | Request a upload task ID from Agent |
| /rest/Agent/UploadFileChunk | POST | Upload the specific file chunk to Agent |
| /rest/Agent/UploadRunningCfg | POST | Upload the running configuration of a device |
| /rest/Agent/GetSoftwaresForDevice | POST | Get available software list that can be applied to the specific device based on the device’s running configuration |

# DATA MODEL

This chapter describes the conceptual view of data model in GDMP 4.0.

## Conceptual View

The following diagram illustrates the conceptual view of data model in GDMP 4.0.



Figure 6: Conceptual View of Data Model

* "CoT" – Class of Trade defines top level types of product line. Each “CoT” could have multiple “Device Type”.
* By “Device Type”, “Device(s)”, “System Configurations” are categorized. Each “Device” may have “Feature License(s)”.
* A “System Configuration” consists of one software configuration and one hardware configuration as well as Feature Licenses configuration.
* A “Software Configuration” contains one or multiple software and a “Hardware Configuration” contains one or multiple hardware as well as Feature License configuration.
* Each software could have several documents associated.
* Each hardware could have several documents associated.
* Each document could associate to several software or hardware.

## Supplement Data Files

GDMP 4.0 loads several data files as supplement data input.

The following table describes a list of such data files and contents, along with component names that will make use of the data.

\*All properties are key-value formatted.

|  |  |  |
| --- | --- | --- |
| File Name | Description | Used in Component Name |
| country.properties | Country name and code.  e.g.  United States = US  China = CN | User Service;  Device Service  Named Configuration Service  Report Service  Alert Service  Trade Embargo Service |
| language.properties | Language name and code.  e.g. International English = qab | Device Configuration Service |
| region.properties | Region name and digital value.  e.g. Asia = 1 | Device Service  Report Service |
| software\_type.properties | Software type name and digital value.  e.g. Software Bundle = 1 | Device Configuration Service;  Named Configuration Service; |
| hardware\_type.properties | Hardware type name and digital value.  e.g. Motherboard = 0 | Device Configuration Service;  Named Configuration Service; |
| document\_type.properties | Document type name and digital value.  e.g.  User Guide = 0  Release Notes = 1 | Device Configuration Service; |
| software\_status\_type.properties | Software status name and digital value.  e.g.  Unknown = 0  Limited Release = 1 | Upload/Download Service  Device Configuration Service |
| hardware\_status\_type.properties | Hardware status name and digital value.  e.g.  Unknown = 0  Limited Release = 1 | Device Configuration Service |
| document\_status\_type.properties | Document status name and digital value  e.g.  Limited Release = 1  In production = 2 | Upload/Download Service  Device Configuration Service |
| named\_config\_status\_type.properties | Named configuration’s status name and digital value  e.g.  Limited Release = 1  In production = 2 | Named Configuration Service |
| alert\_category.properties | Category for alert  e.g.  Business alert = 0  Informational alert = 1 | Alert Service |
| alert\_status.properties | Alert’s status enumeration  e.g.  Pending = 1  Sent = 2 | Alert Service |
| named\_config\_type.properties | Named configuration’s type name and digital value  e.g.  Limited Release = 1  In production = 2 | Named Configuration Service |
| system\_config\_warning\_status.properties | Name and digital value for the warning status of system named configuration  e.g.  invalid\_hardware = 0  incompatible-software = 1 | Named Configuration Service |
| hardware\_config\_warning\_status.properties | Name and digital value for the warning status of hardware named configuration  e.g.  warning = 0 | Named Configuration Service |
| software\_config\_warning\_status.properties | Name and digital value for the warning status of software named configuration  e.g.  warning = 0 | Named Configuration Service |
| device\_service\_type.properties | Device service type name and digital value  e.g.  Configuration Update = 0  Log Retrieval = 1 | Device Service  Report Service |
| device\_upgrade\_status.properties | Device upgrade status name and digital value  e.g.  installed = 0  not attempted = 1  log retrieval = 2  failed = 3 | Device Service  Report Service |

## Data Sources from External System

PUBLIC SCHEMA holds full set of customer, device, and sales record information.

Customer and device information are imported from PUBLIC SCHEMA to GDMP by ETL tool. Files are in CSV format.

Feature license file format are to be decided. And the feature license file will be generated during the ETL process.

## LDAP Data Structure

User, PERMISSION and ROLE information will be managed by LDAP. The GDMP Server will keep user accounts and their PERMISSION in a single LDAP location logically. LDAP service is used to authenticate non-Medtronic user. While for Medtronic users, only Medtronic Active Directory can do the authentication. After successful authentication, Medtronic user information will be exported into LDAP service. So eventually LDAP service will contain both non-Medtronic users and Medtronic users,

The diagram below shows an illustration of LDAP directory in GDMP 4.0.



Figure 11: Users and Roles & Permissions

As shown in the diagram above, the individual PERMISSION will be defined as “Named Attributes” on LDAP schema level, and these PERMISSIONs will be grouped as ObjectClasses (ROLE) then used to define individual group, such as Compression CoT Admin. User must be assigned to a group and the PERMISSIONs of assigned group will be validated by GDMP Server processing requests.

By default, there are eleven types of permissions inherited from current GDMP.

|  |  |
| --- | --- |
| Permission Name | Value |
| Devices | “No” or “Yes” |
| Reports | “No” or “Yes” |
| Hardware\_catalog | “No”, “View Only”, or “View & Edit” |
| Software\_catalog | “No”, “View Only”, or “View & Edit” |
| Document\_catalog | “No”, “View Only”, or “View & Edit” |
| Configuration\_management | “No”, “View Only”, or “View & Edit” |
| Alert | “No”, “View Only”, or “View & Edit” |
| Feature\_license | “No”, “View Only”, or “View & Edit” |
| Trade\_embargo | “No”, “View Only”, or “View & Edit” |
| User\_management | “No”, “View Only”, or “View & Edit” |
| Limited\_release\_software | “No” or “Yes” |

# SECURITY

This section provides security guidance that will be applied across the system.

## Authentication, Authorization and Access Control

This section discusses security consideration on authentication, authorization and access control.

### Authentication

User must be authenticated by user account and password before using GDMP system via Common Client, Legacy Client or GDMP Server Web UI. The authentication will be handled by GDMP Server. When GDMP Server is unreachable by GDMP Agent (offline mode), GDMP Agent will authenticate user by using local cached user credential.

In order to support offline mode, GDMP Agent will cache user credential in file. User will need to sign on at least once on GDMP Agent before using offline mode.

When user logins to the system via Common Client or GDMP Server Web UI, a MD5-hashed password string will be sent instead of plain-text password. Legacy client will remain the old way.

### Authorization

User needs to register a user account to use GDMP System. System admin will approve the user account application and assign certain permission to user.

### Access Control

User permission is role based. User cannot access resources (Web UI, devices, software, documents, etc.) without corresponding user role assigned.

## Data Communication

This section discuss security consideration on data communication.

### Communication between Clients and Agent

Considering backward compatibility, the communication mechanism between GDMP Agent and Legacy Client will remain the same as existing system.

The communication between GDMP Agent and Common Client will be based on HTTPS with a Medtronic signed X.509 Certificate on GDMP Agent. Common Client will verify GDMP Agent’s Certificate.

### Communication between Agent and Server

There are two protection levels.

First level is channel security protection. The communication between GDMP Agent and GDMP Server will be based on HTTPS. Both GDMP Server and GDMP Agent will have its own X.509 Certificate. To prevent man-in-the-middle attack, both GDMP Agent and GDMP Server will verify the certificate of each other.

Second level is data security protection. GDMP Agent will encrypt data by using PBE (Password Based Encryption). For the transferred data, GDMP Agent will generate MD5 hash code of the data and then perform the digital signature with another X.509 Certificate (it’s devoted for digital signature). To protect the password, the public key of Agent Certificate will be used to encrypt the password and only Agent has the private key to decrypt it.

## Data Storage

On GDMP Server and GDMP Agent, access to database is based on database user authentication.

All sensitive configuration files on GDMP Agent will be encrypted by PBE (Password Based Encryption).

In order to support offline mode, GDMP Agent will cache and encrypt user credential in file by PBE.

All software packages, documents and business rules on GDMP Server and GDMP Agent will be encrypted. The encryption algorithm, key and IV (Initialization Vector) are configurable. By default the algorithm is AES/CFB8. GDMP Server and GDMP Agent should share the same algorithm, key and IV. To protect the algorithm, key and IV, the public key of Agent Certificate will be used to encrypt them and only Agent has the private key to decrypt them.

For compatibility reason, the way GDMP Agent transferring files to Legacy Client will remain same as existing system which means packages, documents and business rule files will be decrypted on GDMP Agent when Legacy Client asks for them. Decrypted files will be deleted when expiration time is reached.

## Data Privacy

GDMP 4.0 will follow the HIPPA regulation.

# HIGH AVAILABILITY

This chapter discusses consideration on high availability of GDMP system and SPOF along with service recovery strategy.

## Service Resilience

According to current production deployment, there are two web servers where GDMP Web will be deployed. Load balancing mechanism applies and service resilience of GDMP Web is implemented.

## Single Point of Failure and Service Recovery

Single Point of Failure (SPOF) exists in current GDMP deployment. There is only one Application Server where GDMP Server will be deployed and another Application Server where Database Server and LDAP Server will be deployed.

In this case if any of these two Application Servers or any deployed service stops working, GDMP system service will become unavailable.

In event of service failure, appropriate actions have to be taken to recover GDMP service.

|  |  |
| --- | --- |
| Failure Point | General Recovery Strategy |
| Database Failure | Fix database server issue or recover last data backup in a new instance |
| LDAP Failure | Fix LDAP server issue or recover last data backup in a new instance |
| GDMP Server Failure | Fix issues or deploy another instance |

## Solution for High Availability

Since current GDMP deployment has SPOF issue, GDMP 4.0 provides support to High Availability from architectural perspective.

* GDMP 4.0 Server will follow stateless principle of RESTful Service and multiple GDMP Server instance can be easily deployed in a cluster. When any instance of GDMP Server in the cluster fails, GDMP 4.0 is still available to end users without service interruption because remaining GDMP Server instances can process requests seamlessly.
* Failover mechanism and data replication will be adopted in Database/LDAP Server. To achieve that, it will require a pair of two servers and configured as active-standby cluster. In event of the active server failure, standby server will take over the service to process requests as soon as cluster software detects failure and completes service switchover. Database and LDAP data are replicated across servers automatically.

# SCALABILITY

There are two type scale out approaches the GDMP 4.0 supports. They are traditional Horizontal scale-out approach and Enterprise Service Bus (ESB) type scale-out.

**Scale-Out (Horizontal) approach** will be considered to support GDMP 4.0 scalability. With this approach, each GDMP Server or GDMP Web instance will run in a virtualized environment (VM) or a physical server. When the current system reaches capacity, adding more VM or physical servers with additional instances and configuring Load Balance (LB) proxy to route requests to more destination could increase system processing capacity.

As the GDMP 4.0 Server is a service-oriented architecture, all services within the GDMP 4.0 are independent from each other, and the **ESB type scale-out approach** is deploying different services on different nodes (VM) or physical server. Requests Routing component will be responsible to redirect service request to corresponding service and keeping the main communication endpoint (e.g.: URL) to GDMP Agent and GDMP Web no change.

# DEPLOYMENT AND DISTRIBUTION

The following table contains the target Production Environment (inherited from the current GDMP). And the GDMP 4.0 will be deployed into the same Production Environment.

|  | Web Server1 | Web Server2 | Application Server | Database Server |
| --- | --- | --- | --- | --- |
| Operating System | Red Hat Enterprise Linux 6.2 | Red Hat Enterprise Linux 6.2 | Red Hat Enterprise Linux 6.2 | Red Hat Enterprise Linux 6.2 |
| System type | Linux | Linux | Linux | Linux |
| Memory | 4 GB | 4 GB | 24 GB | 24 GB |
| CPU | 2 CPUs | 2 CPUs | 8 CPUs | 8 CPUs |
| Model | Intel(R) Xeon(R) CPU  X5570  @ 2.93GHz | Intel(R) Xeon(R) CPU  X5570  @ 2.93GHz | Intel(R) Xeon(R) CPU  X5570  @ 2.93GHz | Intel(R) Xeon(R) CPU  E5450  @ 3.00GHz |
| Serial Number | vm1489401 | vm1489501 | vm1489101 | ph1443401 |
| Physical Location | Terremark Farm RSSPROD  (11476801) | Terremark Farm RSSPROD  (11476801) | Terremark Farm RSSPROD  (11476801) | Terremark Farm RSSPROD  (11476801) |

From the table above, we can see there are three kinds of servers, two Web Servers, one Application Server and one Database Server.

GDMP Web will be deployed in both Web Servers.

GDMP Server will be deployed in Application Server.

MySQL database server and OpenDJ LDAP server will be deployed in Database Server.

The following diagram shows how GDMP will be deployed.



Figure 12: Deployment View

## Monitoring and Hardware Support

All GDMP server and GDMP Web shall participate in the Corporate IS remote security monitoring infrastructure.

## GDMP Agent Distribution

This section describes redistributable component of GDMP 4.0 Agent and applicable strategy.

### Distribution and Key Protection

This section discusses the distribution and key protection of component GDMP Agent.

#### Packaging

GDMP Agent will be packaged in a silent-installation enabled installer and wrapped in a normal interactive installer.

The GDMP Agent installer will only contain Agent program and a one-time used certificate which contains a temporary key for the GDMP Agent.

#### Distribution

GDMP Agent will be downloaded from GDMP Server by GDMP user who has been permissioned to download after a successful login.

#### Installation

Installation must be performed by user with Administrative privilege.

The interactive installer will take the following actions:

* Check/create a user account named “gdmpagent” and this account should be hidden from Windows login screen - prohibited from interactive logon.
* Run silent-installer under the special user account.
* The silent-installer will register GDMP Agent as a Windows Service running as user “gdmpagent”.
* The Agent will be installed under the special user’s profile instead of a general place.

#### GDMP Agent Workflow

GDMP Agent will stop functioning if it is not running as user “gdmpagent”.

During the first time startup of GDMP Agent, it will check if there is a pre-defined certificate under Windows key store of “gdmpagent” profile. If not, GDMP Agent will establish the secure channel with the GDMP Server by using the one-time certificate and send request to the GDMP Server. The certificate will be installed into Windows key storage space of the “gdmpagent” user profile and exporting of private key is disabled.

Once the certificate is installed successfully, the one-time certificate will be destroyed permanently.

# MAINTENANCE

This chapter describes high-level design consideration on regular maintenance of the GDMP 4.0 System.

## Database Backup and Archiving

Two databases need regular backup:

* Backup of primary MySQL database, where most system data are stored, will be managed by dedicated Medtronic team.
* Backup of LDAP server’s backend database - Berkeley DB Java Edition must be scheduled regularly.

Records archiving for primary MySQL database is necessary as well. Retention period of records is configurable given that there is requirement on 1-year searchable device logs on GDMP for example.

## Hot Fix and Upgrade

After deployment in production, patching and upgrading strategy applies to meet SLA and Medtronic release procedure must be followed. There are two categories of patching and upgrading.

* Operating System and Fundamental Server Software

In case critical security patches are needed for Operating System (like OpenSSL security patch) and Fundamental Server Software (like security patch to Apache Web Server, LDAP Server, MySQL Server), Medtronic IS team shall notify GDMP product management team and schedule relevant upgrade. GDMP service might be interrupted as service restart could be necessary to complete such upgrades.

* GDMP Components

To apply critical hot fixes or functional upgrade to GDMP Web or GDMP Server, upgrade can take place on several selected instances in cluster as first batch and hot fixes must be verified before upgrading remaining instances. If upgrade is a failure, deployment team has to roll back GDMP installation to previous working version. Remaining instances can continue providing service and get upgraded after first batch is confirmed upgraded.

In addition, GDMP Agent is designed to upgrade itself when new version is available.

In current implementation, when GDMP Agent startup and connect to GDMP Server successfully for the first time, it will check if there is new version available on GDMP Server. If there is, GDMP Agent will download the new version. When the new version is downloaded, GDMP Agent will shut down itself and then the monitor service (boot loader) will upgrade GDMP Agent to new version.

# DATA MIGRATION

This chapter explains data migration strategy from current GDMP system to GDMP 4.0.

## Data Category

Data in the current GDMP can be categorized into 4 major types:

**Business Data**: This category contains basic business related data, such as software, hardware, configuration, device, device type, CoT, alert, notification and so on. These data must be migrated to GDMP 4.0 database to ensure no data loss or corruption.

**User and Entitlement Related Data**: Contains all group, permission settings, user, user related subscription, tanning record, and permissions. These data must be migrated to GDMP 4.0 LDAP and Database to ensure no data lost or corruption.

**Historical Related Data**: Contains maintenance log, activity log, service history. These data ideally will be migrated to ensure as less lost as possible.

**Temporary Data**: Contains all in-progress data, such as pending list. These data will not be migrated.

**Files**: package repository and logs. These data will be migrated.

## Data Migration Process

In GDMP 4.0, user and role & permission related data will be stored in LDAP, and other data will be stored in database.

There are two kinds of data to be migrated. The following sections will describe them in detail.

### Database Tables to LDAP

Firstly, migrate user and entitlement related data to LDAP as below:

1. Migrate role & permission
   1. Export the relations between role and permission in a matrix file from database.
   2. Review the role-permission matrix.
   3. If review passed, import the role and permission data to LDAP.
   4. Check all the role and permission information on LDAP.
2. Migrate users
   1. Export the relations between user and role in a matrix file from database.
   2. Review the user-role matrix.
   3. If review passed,
      1. For non-Covidien user: import full users profiles (e.g.: user name, email, password, etc.) to LDAP and assign the user with role on LDAP accordingly.
      2. For Covidien user: import users’ basic information (e.g.: user name, email) to LDAP and assign the user with role on LDAP accordingly.
   4. Check all the users’ basic information.
   5. Check all the relation between user and role.

### Database Tables to New Database Schema

Secondly, migrate other data to new designed database as below:

1. Backup the source database for migration.
2. Migrate training record and download permission to new database.
   1. Select a user as a sample, extract the information of this user from old database and migrate it to the new database.
   2. Compare the user permission between old system and new system from UI.
   3. If the user permissions between two systems matches, write scripts to automate the operation described above and migrate all permission information.
   4. Check the number of data migrated, if the number doesn’t match with existing system, identify the root case and adjust the script unless the data is dirty.
   5. Check the result between old system and new system from UI under a specified coverage.
3. Migrate the basic management business data such as COT, device type, hardware, software, document, named configuration, trade embargo country and etc.
   1. For every category, pick a sample for each type, extract relevant data from old database and migrate it into the new database.
   2. Compare the sample data one by one between the old system and the new system from UI
   3. If the data between two systems matches, write scripts to automate the operation described above and migrate all relevant data.
   4. Check the number of data number migrated, if the number doesn’t match with existing system, identify the root cause and adjust the script unless the data is dirty.
   5. Check the result between old system and new system from UI under a specified coverage.
4. Migrate the device and component information
   1. Pick one device per device type to migrate to the new database.
   2. Compare the device data one by one between the old system and the new system from UI
   3. If the data between two systems matches, then test the device behavior with mockup tool.
   4. If all passed, write scripts to automate the operation described above and migrate all relevant data.
   5. Check the number of data number migrated, if the number doesn’t match with existing system, identify the root cause and adjust the script unless the data is dirty.
   6. Check the result between old system and new system from UI under a specified coverage.
5. Migrate device history and activity log
   1. Pick one device as a sample to retrieve the device history and activity log, migrate the data to the new database.
   2. Compare the device data one by one between the old system and the new system from UI
   3. If the data between two systems matches, write scripts to automate operation described above and migrate all relevant data.
   4. Check the number of data migrated and get a percentage of migration.
   5. Check the result between old system and new system from UI under a specified coverage.

### Package repository and log files

These are physical files on File System, the data in DB will only store the location. When migrate these physical files, the location record in database will be updated accordingly.

# REQUIRED THIRD PARTY SOFTWARE/COMPONENTS

This chapter lists all required third party software/components used in GDMP 4.0.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Name | Version | License | Description |
| 1 | Spring Framework | 4.0.6 | Apache License, Version 2.0 | The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform. |
| 2 | MyBatis | 3.2.8 | Apache License, Version 2.0 | MyBatis is a first class persistence framework with support for custom SQL, stored procedures and advanced mappings. MyBatis eliminates almost all of the JDBC code and manual setting of parameters and retrieval of results. |
| 3 | OpenDJ | 2.6.0 | Common Development and Distribution License (CDDL) 1.0 | OpenDJ provides open source directory services, including a high performance, highly available, secure directory server, built-in data replication, client tools, and an LDAP SDK. |
| 4 | Oracle MySQL community version | 5.1.73 | GPL license | MySQL is one of the world's most popular open source databases |
| 5 | Jersey | 1.3 | COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL - Version 1.1) and GPL - Version 2 | Jersey is an open source, production quality framework for developing RESTful Web Services in Java that provides support for JAX-RS APIs and serves as a JAX-RS (JSR 311 & JSR 339) Reference Implementation. |
| 6 | H2 | 1.3.176 | MPL 2.0 (Mozilla Public License Version 2.0) or under the EPL 1.0 (Eclipse Public License) | H2 is a relational database management system written in Java. It can be embedded in Java applications or run in the client-server mode. |
| 7 | Netty | 3.5.2 | Apache License, Version 2.0 | Netty is a NIO client server framework which enables quick and easy development of network applications such as protocol servers and clients. It greatly simplifies and streamlines network programming such as TCP and UDP socket server. |
| 8 | Slf4j | 1.7.7 | MIT license | The Simple Logging Facade for Java (SLF4J) serves as a simple facade or abstraction for various logging frameworks (e.g. java.util.logging, logback, log4j) allowing the end user to plug in the desired logging framework at deployment time. |
| 9 | Maven | 3.2.5 | Apache License, Version 2.0 | Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information. |
| 10 | Jenkins | TBD | MIT Licenses | CI tool |
| 11 | Chef | TBD | Apache License, Version 2.0 | Deployment tool |
| 10 | Ant | 1.9.4 | Apache License, Version 2.0 | Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other. |
| 11 | Tomcat | 7.0.29 | Apache License, Version 2.0 | Apache Tomcat™ is an open source software implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies. |
| 12 | Apache | 2.2.20 | Apache License, Version 2.0 | Apache http server is an open-source HTTP server for modern operating systems including UNIX and Windows NT. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards. |
| 13 | Robot Framework | 2.9.2 | Apache License Version 2.0 | Robot framework is an open source software used for automation test, which support keyword-driven test and data-driven test. |
| 14 | Python | 2.7.10 | GPL-compatible | Runtime support for Robot Framework |
| 15 | Selenium | 1.7.4 | GPL-compatible | Selenium is used for automating web applications for testing purposes, but it is not limited to just that. Boring web-based administration tasks can also be automated as well. |

# PRODUCTION RELEASE PROCEDURE

GDMP will follow Medtronic product release standard for the production release.

# PRODUCTION SUPPORT & ESCLATION PATH

Align with Medtronic standard, follow existing support and escalation path defined in R0054192\_A PLN-PROC DMP Production Support Gateway.