# Table of Contents

## TABLE OF CONTENTS ................................................................................................................................. 2

## DEVICE MANAGEMENT FEATURES ........................................................................................................... 3

- **mPRM Generic Device Management Framework** ..................................................................................... 3
- **OSGi Device Management** ....................................................................................................................... 5
  - **OSGi Device Registration and Initial Provisioning** ................................................................................. 5
  - **mPRM Management Protocol** .............................................................................................................. 6
  - **Device Directory** .................................................................................................................................. 7
  - **Retrieving and Storing of the Device Configuration** .......................................................................... 8
  - **OSGi Framework Settings Management** ............................................................................................ 9
  - **OSGi Bundle Management** ................................................................................................................. 9
  - **Generic Software Management** ......................................................................................................... 10
  - **OS-Level Management and Monitoring** ............................................................................................. 10
  - **Bundle Configuration (Settings) Management** .................................................................................... 10
  - **Preferences Data Management and Data Synchronization** ............................................................... 10
  - **Providing of Custom Management Interfaces** ................................................................................. 11
  - **Home Network Devices Management** ............................................................................................... 12
  - **Retrieving Device Logs** ...................................................................................................................... 12
  - **Fault Alerts** ........................................................................................................................................ 12
  - **Application Monitoring** ..................................................................................................................... 12
  - **mPRM Device Driver Locator** ............................................................................................................ 13
- **TR-069 Device Management** ................................................................................................................. 13

## MASS MANAGEMENT OPERATIONS .......................................................................................................... 15

## SOFTWARE REPOSITORY ............................................................................................................................ 16

## MPRM GUI – MANAGEMENT CONSOLE .................................................................................................... 17

## API AND INTEGRATION INTERFACES ....................................................................................................... 18

## SECURITY .................................................................................................................................................... 18

- **User Authentication and Authorization** .................................................................................................. 18
  - **User Authentication** ............................................................................................................................ 18
  - **User Authorization** .............................................................................................................................. 18
- **Secure Network Communication** ........................................................................................................ 19
- **Certificate Management** ...................................................................................................................... 19

## MPRM SCALABILITY AND HIGH AVAILABILITY .......................................................................................... 20

- **Single-Server Configuration** .................................................................................................................. 20
- **Dynamic-Clustered Configuration** ........................................................................................................ 21
- **Mixed-Clustered Configuration** ........................................................................................................... 22

## PLATFORM REQUIREMENTS ...................................................................................................................... 23

- **Software Requirements** ....................................................................................................................... 23
- **Hardware Requirements** ....................................................................................................................... 23
- **Database Requirements** ...................................................................................................................... 23

## APPENDIX A: RESOURCES .......................................................................................................................... 24
Device Management Features

ProSyst mPower Remote Manager (mPRM) is a backend management and monitoring system, supporting centralized configuration, software provisioning and service operation support for various class of devices based on (but not limited) to OSGi, OMA-DM and/or TR-069 standards. In short, mPRM is applicable to realize these use cases:

- Application & SW Component Lifecycle Management
- Firmware & File Update
- Remote Configuration & Provisioning
- Remote Diagnostics (life status checks, logging, monitoring, etc.)
- Remote Security Administration

mPRM is not limited to these use cases, support for additional cases (standardized or customized) can be added at any time.

ProSyst mPRM is unique in its openness, boosting rapid adoption of new requirements and integration with 3rd party systems. The open design has the following dimensions:

- Open for additional (i.e. custom) management protocols
- Open for custom business logic extensions hosted on mPRM
- Open for integration with 3rd party systems through an extensive set of APIs (Java, REST/HTTP, SOAP or JEE JCA based APIs)
- Open for customized management consoles (UIs)

This document describes the main features of the mPRM system.

mPRM Generic Device Management Framework

The foundation of the mPRM system is a general model and framework for management and monitoring of various device types in a uniform way. The Generic Device Management framework employs a general abstraction for device representation and a number of basic features, which are common for management and monitoring of any device type - regardless of the concrete device capabilities, the network communication protocols and the management interfaces supported by the device.
The interface to the devices of particular type is provided by a set of plugins (adapters), which adapt the concrete device management interface (e.g. OMA DM, TR-069, etc) supported by the devices to the common device model of the mPRM.

This model allows extending of the mPRM with support of new device types with minimal effort, leveraging the basic common device management functionality.

mPRM provides build-in support for the following device classes:

- devices running OSGi compliant frameworks – like residential gateways, home device controllers, set-top-boxes, telematics head units, etc.
- mobile devices supporting OSGi Mobile R4 specification (JSR - 232)
- mobile devices supporting OMA Device Management protocol (version 1.1 and 1.2)
- devices supporting CPE WAN Management protocol (TR-69) defined by Broadband Forum
- generic native devices running ProSyst mBedded System Agent
OSGi Device Management

mPRM supports management of devices equipped with OSGi Release R2–R4+ compliant framework. It order to enable an OSGi device to be managed by the mPRM, it has to be equipped with the mPRM Management Agent. The mPRM Management Agent represent a set of OSGi bundles installed on the target OSGi framework of the device. mPRM provides versions of the Management Agent for Java SE, Java Micro Edition (CDC) and Android (OSGi over Android) based devices.

The mPRM Agent communicates with the management server over ProSyst proprietary messaging protocol. This approach provides rich and efficient management and monitoring functionality of the standard OSGi framework and application resources (bundles, configuration settings, log messages, user preferences, security permissions), as well as management and monitoring of some of the OS level elements, like processes, platform resource utilization (RAM, CPU, network interfaces) and applying of OS level software updates. The protocol is open for extensions providing management and monitoring capabilities to custom platform and/or application specific resources.

OSGi Device Registration and Initial Provisioning

mPRM keeps record of each manageable device. The process of inserting the record about a particular device into the mPRM database is referred as device registration. mPRM supports two general models for OSGi device registration:

- Explicit registration – the information about the new device is added through the mPRM backend – either by custom application (through the mPRM API) or manually - by the operator (through the mPRM GUI console. This method implies that device is registered before it connects to the mPRM for the first time. mPRM supports assigning of initial set of configuration settings to the registered device which will be applied upon first device connection.

- Automatic - the new OSGi device is automatically registered in the mPRM when it connects to the backend server for the first time. mPRM automatically retrieves the device identification and configuration information and stores it into the mPRM DB. This model can be disabled if there is a requirement (e.g. for security reasons) only explicitly registered devices to be allowed to connect to the mPRM system.

In order to become manageable by the mPRM, the OSGi device must be provisioned with the mPRM Management Agent bundles and suitable configuration settings for establishing management session with the server. This process is called initial provisioning.

mPRM supports two general ways for initial provisioning:

- Pre-provisioned - the mPRM Management Agent and the related configuration can be pre-installed on the device and/or manually provided upon device deployment.
Dynamically provisioned – mPRM / mBS support the standard OSGi Initial Provisioning mechanism, which defines the way for provisioning of the OSGi framework with the agent bundles and the related configuration settings needed for connecting to a particular management system (from arbitrary vendor).

In order to start the initial provisioning process, the OSGi Initial Provisioning Service has to be provided with settings for connecting to a provisioning server (e.g. URL, device ID, authentication parameters). These initial provisioning settings can be provided either through static configuration of the OSGi framework, or can be provided dynamically by custom module/application. mPRM can act as an OSGi Initial Provisioning server, but it is also possible to use a separate provisioning server.

**mPRM Management Protocol**

The communication between mPRM Management Agent running on the manageable OSGi devices and the mPRM server is based on ProSyst proprietary communication mechanism - mPRM Message Service. The mPRM Message Service utilizes a highly optimized binary protocol - in regards of network data volume, CPU and memory utilization (on device and on server side).

Some of the key features provided by the mPRM Message Service are:

- Asynchronous and synchronous message-based communication
- Bi-directional message exchange between the backend server and the OSGi devices - both the server and device can instantly send messages to each other.
- Communication sessions – the messages are carried over an active management session. The network connection is always initiated by the device, but once the session is open – the server can also send messages to the device (server push).

mPRM supports different models for session establishment:

- Always online mode – in this mode, the mPRM Management Agent tries to maintain the management session permanently active - once the session is open, it is kept alive indefinitely and if it is closed due to some network problem - the agent periodically tries to re-establish it.
- Periodical connection – the connection is open at scheduled time intervals (e.g. once per day or week) and after the device updates its status and executes any pending operation for it – the session is closed (or kept active only for some configured time interval of inactivity).
- Event-driven – the connection may be open in result of fault alert or error log message generated on the device, which must be instantly sent to the backend.
- By user or application request – through the mPRM Management Agent API.
mPRM Message Service can operate over various network transport protocols – the design is based on pluggable transport binding components, which handle underlying communication mechanism. mPRM provides build-in transport bindings for TCP, SSL/TLS, HTTP(S) and UDP network protocols. Bindings for additional transport protocols can also be added.

Optional secure transport layer (in addition to the SSL/TLS based one), which provides secure certificate based authentication and data encryption capabilities on top of any of the available transport bindings. The secure transport layer is useful only if the underlying transport binding does not provide build-in security features (e.g. UDP).

Device Directory

mPRM provides flexible and convenient mechanisms for grouping, filtering and searching of the registered devices by various built-in and user defined device characteristics and status.

mPRM supports two approaches of device grouping (which can be combined):

- **Static device grouping** – the devices can be statically organized in user-defined groups. The groups can be organized hierarchically, i.e. one group may have sub-groups, in arbitrary level of nesting. The devices can be moved by the operator from one group to another - at any time.

- **Dynamic device grouping** – mPRM supports defining of custom groups based on specified filtering condition. The members of such groups consist of the devices which satisfy the specified condition. The condition can be an arbitrary expression based on any of the static and the dynamic characteristics of the OSGi device maintained by mPRM - e.g. online status, device capabilities (e.g. HW, OS, OSGi parameters, etc.), available software components, their runtime status and configuration, etc. including additionally associated custom properties with the devices.

Device grouping helps operator to monitor and locate devices through the mPRM GUI console and can be also used for targeting of mass management operations (e.g. perform particular operation over the devices belonging to a given group).

The mPRM also provides features for direct searching of devices – either by unique identifier or by more complex filtering condition (as for the dynamic groups).

The mPRM supports associating of additional custom properties with each single device or with whole device group. When such property is associated with a device group – it is “inherited” by all sub-groups and devices belonging to that group. Example for such properties can be location, customer-related data, address, properties for integration with external systems, etc.

These properties are in the form of key-value pairs, where the keys are strings and the values can be of type string, integer, boolean, byte, double or arrays of the above. Manipulating of these properties can be done either manually - through the mPRM GUI console, or by custom applications – through the mPRM APIs.
Retrieving and Storing of the Device Configuration

mPRM collects, retrieves and stores in the database configuration and status information about each device it manages.

The configuration data retrieved and stored by the mPRM include:

- **Device capabilities** – the characteristics of the device hardware (e.g. CPU model, RAM, Flash/HDD size, etc.) and software platform (OS name and version, JVM vendor and version, OSGi framework vendor, supported specification version, available software component/modules, etc).

- **OSGi framework and JVM configuration settings and status**

- **Available OSGi bundles and applications, their status and configuration, registered OSGi services and bundle security permissions.**

- **OS packages, processes, and platform resource utilization statistics.**

- **Fault alerts, error/warning log messages** - the type and content pattern of the log messages that are sent automatically to the server can be configured, but the operator is able to explicitly retrieve any type of log messages on demand.

- **Home network devices configuration and status** – this includes the data from the devices in the home network accessible through the OSGi framework acting as residential gateway or home device controller.

- **Configuration of the additional custom resources.**

- **Custom monitorable data**

The information is collected and transferred to the management server by the mPRM Management Agent – either automatically (implicitly) or on-demand (upon explicit request). The exact type of data which is subject of automatic synchronization can be configured, while the information that is not synchronized automatically - is delivered to the backend on demand.

Device configuration synchronization is an optimized process - it does not lead to transferring of full information about the configuration and status every time (unless the device connects for the first time to the mPRM), but only checksums and changes. The mechanism can be described briefly as follows:

1) **When the device connects to the backend (mPRM server) the changes in the device configuration are transferred to the server and updated in the mPRM database. If the device connects for the first time, this step effectively retrieves and stores full device configuration data (which is subject of automatic synchronization).**
2) Once the session is established the device (mPRM Management Agent) instantly sends update events for changes occurred in the device configuration and state (the data which is subject to automatic synchronization - e.g. application started/stopped, changed properties, fault alerts and application alarms, error/warning log messages, etc). This data is updated in the mPRM database and appropriate notifications are sent to the interested applications.

The device configuration data is available to system operators - through the mPRM GUI management console, and to the custom backend applications - through the mPRM API.

**OSGi Framework Settings Management**

mPRM supports retrieval and remote manipulation of the global OSGi framework parameters:

- Framework start level - compatible with OSGi Start Level specification
- OSGi/JVM system properties
- Framework security policy – includes management of default bundle permissions and conditional permissions, compatible with the OSGi Permission Admin and Conditional Permission Admin specifications

**OSGi Bundle Management**

mPRM supports management of OSGi based software on two alternative levels of granularity:

- Low-level bundle management – provides management of separate OSGi bundles on the target device:
  - collects information about installed OSGi bundles for each device – installed through either the mPRM or via any other source (or preinstalled in the OSGi framework). Provides information about static properties and runtime status for each bundle – manifest headers, ID, location, start level, state (STARTED, RESOLVED, INSTALLED, etc.) as well as registered OSGi services at runtime.
  - provides means for remote life-cycle management of OSGi bundles – installing of new bundles and starting/stopping/updating/uninstalling of the existing bundles.

- High-level service application management – supports logical grouping of a set of (one or more) software components (e.g. OSGi bundles, Deployment Packages) and possibly additional resources and content files and provides ability to provision and manage their lifecycle (update, start/stop, uninstall) together – as a whole application, handling issues like dependency and compatibility resolving, sharing of components between applications, etc.

OSGi bundle management features described in this point operate directly over the standard API of the OSGi framework.
Generic Software Management

mPRM supports remote software components deployment and lifecycle management based on the Software Admin service of the ProSyst mBS OSGi Runtime and mBS SDK.

The Software Admin service provides universal façade for installation and life-cycle management of software components of different types – e.g. OSGi bundles, deployment packages, midlets, widgets, OS modules (Debian Packages, IPKGs, etc), Android applications (APK). The Software Admin employs a pluggable model which allows adding support for new software component types to be achieved by developing and installing of corresponding plugins (in the form of OSGi bundles) on the OSGi framework. The model can be used for manipulation of the software not only on the local device running the OSGi framework / Software Admin service, but also to other home devices connected to the OSGi device.

OS-Level Management and Monitoring

- System monitoring - provides information about the current hardware configuration and current utilization of the platform resources – CPU load, free/occupied RAM, network interfaces – utilized bandwidth and traffic statistics, storage (HDD, flash) – free occupied space.

- OS process management – provides information about the currently running OS processes on the target device and status information about each process – PID, command line, status, used memory, priority. Provides the means to start new and stop the running OS processes.

- Native software deployment and update - provides support to retrieve information about installed native packages on devices as well as ability to uninstall/update/uninstall these packages remotely. mPRM supports management of Debian and iPKG packages, as well as a ZIP based package format - for devices that do not have build-in package management capabilities.

- Distribution of software as ZIP Packages provides generic way for software distribution. The ZIP package may contain arbitrary file/dir content, which is extracted to a specified directory on the target device. The ZIP package may optionally contain two OS shell scripts - install.sh and uninstall.sh, which are executed by the mPRM Management Agent upon ZIP package installation and uninstallation respectively.

Bundle Configuration (Settings) Management

mPRM supports retrieving and remote manipulation of the bundle configuration settings though the standard OSGi Configuration Admin service. Configuration settings management is applicable to both system level and application bundles installed on the target OSGi framework.

Preferences Data Management and Data Synchronization

mPRM Preferences service is a data storage service compatible with OSGi Preference Service
specification. It allows storing of application data associated it particular device or user of the mPRM system. The data is maintained persistently in the mPRM backend database, but can be also replicated locally on the target devices, where it can be accessed and possibly changed through the OSGi Preferences service APIs. In this way the preferences data can be accessed/manipulated either by backend-side applications (including mPRM GUI console) or by device-side applications. mPRM Preferences service provides efficient bi-directional data synchronization mechanism for replicating changes in the preferences data between the device-side preferences storage and the central backend DB.

Providing of Custom Management Interfaces

mPRM and mBS SDK provide a powerful mechanism for extending the management capabilities of the system, by exporting management interface to custom resources. The mechanism is based on the so called Control Unit model. Control Unit model allows unified representation of diverse type of resources (devices, software or hardware components, etc.) along with a common way to manage and monitor these resources.

The key characteristics of the control unit data model are:

- state variables – represent the status and configuration information of the modeled resource
- actions – determine the operations which can be performed over the modeled resource
- type and identifier of control units – identifies uniquely the control unit instances.
- hierarchy of control units – control units can be organized hierarchically, thus allowing modeling of more complex systems (by decomposing them in sub-subsystem modeled as separate sub-control units)
- Control Unit Meta-data – provide information about the state variables and actions supported by particular control unit as well as information to the user.

Custom bundles may provide control unit compliant interface to a particular resource by implementing a dedicated java interface and registering it as OSGi service on the target platform. The mPRM agent automatically discovers the exported control units and made them available for management through the mPRM server, including:

- Retrieving of the control unit state (control unit instances and their state variables)
- Storing control unit data in the mPRM database
- Viewing and manipulating control units through the mPRM GUI console and through the mPRM API
• Invoking of the control unit actions – either on a single concrete device or in the scope of mass management operations.

Home Network Devices Management

The Control Unit model described in the previous paragraph can also be used for providing remote monitoring and control interface to the home network devices (e.g. ZigBee devices) connected to the OSGi device (when it acts as a residential gateway / home controller).

Providing a control unit compliant interface to such devices in the OSGi framework makes them accessible and manageable through the mPRM.

Retrieving Device Logs

mPRM provides remote interface for retrieving of log messages generated on the managed OSGi devices. The interface is compatible with standard OSGi Log Service and supports both automatic and on-demand retrieval of the device logs.

• Automatic log transfer – the mPRM Log Agent can be configured to automatically send log messages satisfying particular criteria – based on severity level, source and/or log message text template. Such messages and instantly delivered to the mPRM server, where they are stored and made available to the operator - through mPRM GUI or for custom applications – through the mPRM APIs. Enabling/disabling of automatic log transfer feature and the criteria for automatically transferred log messages can be configured at runtime through the mPRM – as appropriate.

• Explicit log retrieval – the operator can request explicit retrieval of the log files generated on particular OSGi device.

Fault Alerts

mPRM provides a mechanism through which OSGi device applications can rise notifications about some temporal or permanent malfunctions, specific for the concrete application. The alert notifications are transferred to the mPRM and made available to the system operators though the mPRM GUI console. They can be also received by custom application modules (through the mPRM API) which can generate appropriate notifications, e.g. SMS, emails, etc.

Similarly to the log messages, the alerts have severity level, source, and text description. The main difference between alerts and log messages is that the alarms reflect the current status of the system malfunctions, i.e. the alert is automatically closed when the fault situation is resolved.

Application Monitoring

mPRM supports retrieving of monitoring data provided through the standard OSGi Monitor Admin service. The Monitor Admin service can be used by any application running in the OSGi framework to
export arbitrary runtime status information, which can change dynamically.

**mPRM Device Driver Locator**

mPRM Device Driver Locator provides driver locator service compliant with OSGi Device Access Specification, which searches for available device drivers in the mPRM Software Repository. This mechanism supports automatic download and installation of the suitable drivers when a new home device or peripheral device is detected.

**TR-069 Device Management**

mPRM supports configuration and firmware update of customer devices (CPEs) over TR-069 protocol standardized by the Broadband Forum. The basic features of the mPRM TR-069 based device management are:

- Support for TR-69 Amendment 3 (including Annexes F and G)
- Support for Software Module Management as specified in TR-069 Amendment 3 and TR-157 Amendment 3.
- Support for TR-069 based management of OSGi devices compliant with OSGi Residential Management Tree specification.
- Remote monitoring and control of home network appliances (e.g. ZigBee, Z-Wave devices) through an OSGi based gateway (based on the Home Device Management layer of the mBS Smart Home)
- Support of custom data models and vendor specific methods - sophisticated device modeling framework, allowing representation of the manageable elements of the devices as domain-oriented objects, e.g. OSGi bundles, software modules, processes, etc. This allows convenient and intuitive monitoring and manipulation of the devices by the operators. mPRM provides built-in object data models for OSGi devices, as well as for some of the standard (Broadband Forum defined) profiles, and it supports adding of additional custom provided object models for concrete device classes/models.
- Secure communication - connections over HTTP/HTTPS with Digest and Basic Authentication
- Support downloading of firmware updates and software modules from the mPRM Software Repository
- Configurable (data model based) retrieval/synchronization of device parameters - controllable (rule-based) way for initial retrieving and subsequent refreshing of the device parameters, based on the knowledge provided by system integrator/operator about data model (profiles) supported by concrete classes of devices. This provides significant improvement in the performance and scalability of the system and reduces the network bandwidth utilization.
ProSyst also provides OSGi-based device-side implementation of TR-069 stack – as an optional component of the mBS Smart Home SDK and OSGi Runtime. It is compliant with OSGi Residential Management Tree specification, which defines a TR-069 compliant management interface to the OSGi framework and its resources.

The server-side stack of the mPRM and the client-side stack of the mBS are not strictly bound each other – i.e. due to the standard and the general implementation, it is possible to use mPRM for management of any TR-069 device (not just based on mBS/OSGi/Java), and it is possible to use other TR-069 compliant DM systems for remote management of the mBS.

**Mobile (OMA DM-based) Device Management**

mPRM supports management of devices over the OMA DM protocol. OMA DM protocol is defined by Open Mobile Alliance (http://www.openmobilealliance.org) and is widely used in mobile devices.

With the means of the flexible and plugable architecture of the mPRM it is quite easy to implement any kind of management object available on the device to be managed from the backend. Yet the mPRM also provides built in management of some of the common management objects on an OMA DM enabled devices.

**Generic OMA DM Support**

OMA DM Specification defines several common management objects for every OMA DM enabled device. These include:

- Dev Info Management Object with information about the device
- Dev Detail Management Object with details on the device software
- DM Accounts – device preconfigured OMA DM account
- Software Component Management Object (SCOMO)
- Lock and Wipe Management Object (LAWMO)

mPRM provides a build in support for these common managements objects

**OSGi Mobile Management Objects**

OSGi Alliance has released OSGi Mobile Specification R4 that intends to unify the management objects on OSGi enabled devices to be managed via OMA DM protocol. This specification includes the following management objects for OSGi devices that are supported by the mPRM:

- Deployment – for deploying and managing software components on OSGi enabled devices
Policy – for managing device security policies

Configuration – for configuring devices

Log – for extracting device logs

Monitor – monitoring different monitorable indicators on the device.

Application – for lifecycle management of installed on the device applications

Mass Management Operations

Mass management operation is a means for executions of a set of management actions over multiple possibly all devices registered in the mPRM system. Management actions can be e.g. software deployment/update, changing of configuration setting, control unit action invocation, etc.

Some of the key elements of the management operations are:

• Operation target:
  ✓ single concrete device
  ✓ all devices in a particular device group
  ✓ devices, which satisfy a given criteria (filter)
  ✓ newly registered devices (can be also combined with device criteria) – the operation will be executed on devices which connects for the first time to the mPRM. Suitable for provisioning of initial configuration settings and or software applications

• Execution time:
  ✓ instant execution – the operation starts execution instantly
  ✓ scheduled for an exact date-time
  ✓ periodic execution – the operation is executed repeatedly on a specified time period
  ✓ execution on every device connection – the operation is executed on the target device upon every connection with the server

• Operation status – mPRM supports monitoring of the status, progress and outcome of the scheduled, running and finished operations. It maintains the following status data for each operation:
  ✓ number of devices which are target to this operation
number of devices which has finished operation execution

the devices which are still pending

errors and warning, etc.

The Management operations can be executed either through the mPRM GUI console tool or through the mPRM API. mPRM also provides scripting language for defining more complex and reusable management procedures.

mPRM maintains a persistent queue for management/control commands sent to each device. If the device is not currently connected to the mPRM – these commands are sent when the device becomes connected again.

Software Repository

mPRM Software Repository is a backend software and content database compliant with J2EE Client Provisioning specification (JSR - 124), which maintains the meta-information and content files of the software components (applications, system modules/drivers, firmware update files, etc.) which can be remotely provisioned on the target devices through the mPRM.

The main features of the Software Repository are:

- Support for multiple software/content types and distribution formats - OSGi Bundles, OSGi Deployment Packages, MIDlet Suites, Widgets, Android Applications (APK), Debian Packages, IPKG, software images (zip archives), content files (text, audio, video), Provisioning Archives (JSR-124 PAR format). The extensible model allows adding of support for additional software types.

- Common persistent storage for maintaining executable code and content intended for delivery to the devices, including software content files and additional properties facilitating the proper installation/delivery to the devices.

- Software dependency management – automatic and/or manual defining of dependency and compatibility relations between software components and their different versions.

- Software requirement management – automatic and/or manual defining of device capability requirements of the software and content.

- Dependency resolving and capability matching – determines the components and versions which are suitable for every unique device.
mPRM GUI – Management Console

mPRM provides system operators with rich and interactive GUI tools for performing system administration and remote device management and monitoring tasks. The Management Console is available in two forms:

- Stand-alone (windows-like) GUI applications, which connects remotely to the mPRM server.
- Web-based interface – provides GUI to the mPRM accessible through a web browser.

Both versions of the mPRM Management Console provide access to the full functionality of the mPRM. The Management Console is based on the Eclipse RPC and RAP technologies and is open for customization and extension with custom modules and functionality.
API and Integration Interfaces

mPRM provides a extensive and open API to its full functionality allowing extending, customization and integration of the system with custom applications and backend systems.

The mPRM API is available in the following forms:

- mPRM OSGi API – mPRM servers are based on OSGi technology, which allows deployment of custom applications inside the mPRM OSGi framework. These custom bundles may use the mPRM API in the form of OSGi services.

- mPRM Remote Java API – API in the form of Java library providing any external Java based application/system with remote interface to the mPRM.

- mPRM J2EE Resource Adapter – Java Connector Architecture compliant API providing J2EE applications with standard interface to the mPRM.

- Web Service API – provides SOAP 1.2 and REST-based API to either non-Java (.NET, C++, others) or Java-based applications.

Security

User Authentication and Authorization

mPRM User Management package provides rich functionality for creating of user accounts and user groups and for assigning the access rights on a user and/or user group level. The user accounts are used for getting access to the mPRM through the mPRM GUI console applications and also for accessing (by external systems and applications) the remote backend APIs of the mPRM described in the previous paragraph.

User Authentication

mPRM supports basic (username/password) based and secure (based on personal user certificates) authentication methods.

User Authorization

mPRM utilizes flexible and fine-grained access control model based on roles. The control can be done either on specific user level, as well as on a defined group of users, such as system administrators, service providers, end users. The access control restrictions to the protected resources and functionality maintained by the mPRM is enforced on all backend interfaces provided by the mPRM –
GUI tools and remote APIs provided for integration with external systems and applications.

Below are some of the build-in mPRM roles, which can be assigned or not to the users/users groups (thus allowing or denying user access to particular resources):

- **Bundle-Import** – grants rights to import new bundles into the mPRM repository

- **Bundle-View(<bundle_id>)** – parameterized role granting rights to view/use the client bundle available in the mPRM Software Repository, with the specified `<bundle_id>`. This ID may correspond either to single OSGi bundle or composition (application/service).

- **Bundle-Manager(<bundle_id>)** – parameterized role granting rights to edit (e.g. rules, properties, etc) the client bundle available in the mPRM Software Repository, with the specified `<bundle_id>`. This ID may correspond either to single OSGi bundle or composition (application/service).

- **Device-View(<device_node_path>)** – parameterized role granting rights to view/monitor a specified device registered in the mPRM. The `<device_node_path>` may refer either to concrete device or to a group of devices (in the mPRM device tree).

- **Device-Manager(<node_path>)** – parameterized role granting rights to manipulate (e.g. install/upgrade software, change settings, etc.) a specified device registered in the mPRM. The `<device_node_path>` may refer either to concrete device or to a group of devices (in the mPRM device tree).

There are also other roles supported by the mPRM – e.g. related to management operations and procedures, certificates, user management, etc. Additional custom roles can also be defined in the mPRM and used by applications to guard access to their protected resources.

The roles can be defined and assigned to the users and user groups through the mConsole GUI or through the mPRM API.

**Secure Network Communication**

mPRM supports secure (SSL or HTTPS based) communication over its remote interfaces – with the devices and the backend interface – with the mPRM GUI console applications and external systems.

**Certificate Management**

mPRM provides common certificate management module, which maintain mPRM own private key and certificates, as well as trusted certificates. There can be different certificates (own and trusted) specified for the different interfaces of the mPRM - e.g. for mPRM Message service, for HTTP server, for backend API access, for user authentication.
mPRM Certificate Manager supports all of the most widely used formats for importing and exporting of the certificate and private key data. It also supports certificate revocation lists (CRL).

**mPRM Scalability and High Availability**

mPRM supports both vertical (achieved by using more powerful server hardware - number of CPUs/CPU cores and amount of RAM) and horizontal (using a cluster of multiple mPRM servers, which number can be increased with increasing load) scalability approaches.

mPRM Cluster ensures high scalability and availability of the mPRM system, by allowing a number of backend servers to act as a single management server. mPRM supports different clustering models:

- **Static load distribution** - every management server manages a different sub-set of explicitly specified devices
- **Dynamic** - more than one backend server instance is managing the same subset of devices, distributing the load automatically and dynamically.
- **Mixed** - the system may contain multiple clustered servers, each of them responsible for a different sub-set of devices, and every cluster contains multiple backend servers, which distribute the corresponding device sub-set dynamically.

The diagrams below show three mPRM example deployment configurations:

**Single-server configuration**

The figure above depicts the most simple mPRM configuration, suitable for development and evaluation purposes. It consists of only one backend server, which hosts mPRM Control Center (CC), Management Server (MS) and Remote Access Server (RAS).
The figure above shows an mPRM multi-server configuration with a single clustered Management Server and a single RDBMS (which in turn can be clustered). The Clustered Management Server manages all registered devices and distributes the load between the server nodes. RAS servers are deployed on every mPRM host and distribute requests from the other external backend systems.
This figure shows a multi-host configuration, with several clustered Management Servers, each of them working with a separate database server for storing of the device configuration data. This multi-host configuration is suitable for distributed system deployments. Hosts of one MS cluster and the corresponding DB server are connected in a local area network (fast speed). Different MS clusters, CC and RAS cluster communicate via Internet.
Platform Requirements

Software Requirements

mPRM is 100% pure Java implementation and can run on any OS supporting Java SE 1.5+. It is tested with all major OS systems: Linux SuSe/RedHat/Ubuntu, Windows 2000/XP/Vista, Windows Server 2003/2008, Solaris OS.

Hardware Requirements

- mPRM Backend Server

<table>
<thead>
<tr>
<th></th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>Dual-Core Intel Xeon or AMD Operton CPU or other with similar performance.</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>2048 MB minimum</td>
</tr>
<tr>
<td></td>
<td>4096 MB or more recommended</td>
</tr>
<tr>
<td><strong>Disk space, required for Installation</strong></td>
<td>The space depends on selected components. Recommended free space – at least 500 MB.</td>
</tr>
</tbody>
</table>

- mPRM Management Console

<table>
<thead>
<tr>
<th></th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>x86 CPU - 1800+ MHz or comparable CPU</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>1024 MB</td>
</tr>
<tr>
<td><strong>Disk space, required for Installation</strong></td>
<td>50 MB</td>
</tr>
</tbody>
</table>

Database Requirements

mPRM requires an external (3rd party) SQL RDBMS. The supported DB servers are:

- Oracle Database 9i, 10g, 11g (recommended for large scale deployments)
- MySQL 5.0.*
- H2 – an embedded Java RDBMS, which is included in the mPRM distribution for development purposes only

**DB Storage Space**

The following general guideline can be used for determine needed DB storage space for Oracle DB:

- 500 MB data space + 100 MB transactional space + 100 MB of additional data space for every 1000 devices registered in the system.
Appendix A: Resources

You can find more resources about the mPRM at:

- http://dz.prosyst.com – ProSyst Developer’s Zone