# White Paper on Provisioning Objects

Approved – 24 Oct 2008

**Open Mobile Alliance**
OMA-WP-AC_MO-20081024-A

# Contents

1. SCOPE ..... 5
2. REFERENCES ..... 6
3. TERMINOLOGY AND CONVENTIONS ..... 7
3.1 CONVENTIONS ..... 7
3.2 DEFINITIONS ..... 7
3.3 ABBREVIATIONS ..... 7
4. INTRODUCTION ..... 8
4.1 INTRODUCTION TO DM ..... 8
4.2 INTRODUCTION TO ENABLERS IN DM WORKING GROUP ..... 9
5. WHICH TO CREATE? AC OR MO OR BOTH? ..... 11
5.1 AC CREATION ..... 11
5.2 MO CREATION ..... 11
5.2.1 Introduction to Management Objects ..... 11
5.2.2 MO General Recommendations ..... 11
5.2.3 Conversion from AC to MO ..... 11
5.3 AC AND MO CREATION ..... 11
6. AC CREATION FOR OMA CP ..... 12
6.1 OMNA REGISTRATION ITEMS FOR OMA AC FILES ..... 12
6.2 GUIDANCE FOR AC CREATION WITHIN OMA ..... 12
6.2.1 Naming of AC docs ..... 12
6.2.2 Handling of AC docs ..... 12
6.2.3 AC docs in Enabler Releases ..... 13
6.2.4 Getting AC docs into Public Directory ..... 13
6.2.5 Additional Guidance ..... 13
7. MO CREATION FOR OMA DM ..... 14
7.1 CONTENT GUIDANCE ..... 14
7.2 DEFINITION AND DESCRIPTION OF MANAGEMENT OBJECTS ..... 15
7.2.1 Definition and Description of Nodes ..... 16
7.3 ACCESS TYPE GUIDANCE ..... 17
7.4 STATUS AND OCCURRENCE GUIDANCE ..... 18
7.5 FORMAT GUIDANCE ..... 18
7.6 MO GENERATION ..... 19
7.7 OMNA REGISTRATION ITEMS FOR MO DDF FILES ..... 19
7.8 GUIDANCE FOR MO CREATION WITHIN OMA ..... 20
7.8.1 Naming of MO DDF docs ..... 20
7.8.2 Handling of MO DDF docs ..... 20
7.8.3 MO specifications and DDF Files in Enabler Releases ..... 20
8. OMNA REGISTRATION ..... 21
8.1 AC REGISTRATION FOR OMA CP ..... 21
8.2 MO REGISTRATION FOR OMA DM ..... 21
9. SMART CARDS ..... 23
9.2 DATA STORAGE FOR BOOTSTRAP INFORMATION ..... 23
9.3 INTEROPERABILITY AND BACKWARD COMPATIBILITY ..... 23
9.3.1 Smart Cards personalization ..... 23
APPENDIX A. CHANGE HISTORY (INFORMATIVE) ..... 24

# Figures

**Figure 1: Management View Exposed by a DM Client to a DM Server ..... 9**

**Figure 1: MO depicted in graphical notation ..... 16**

**Figure 2: MO depicted in enhanced graphical notation ..... 16**

# 1. Scope

This whitepaper provides best practices on the creation of Management Objects (MOs) and Application Characteristics (ACs). This whitepaper also provides information on how to work with Open Mobile Alliance (OMA) on the registration of names and identifiers of MOs and ACs.

# 2. References


**[ConnMO]** Connectivity Management Objects, Version 1.0, Open Mobile Alliance™, OMA-DM-ConnMO-V1_0,
URL: http://www.openmobilealliance.org/

**[CP]** "Client Provisioning ProvCont", OMA-WAP-TS-ProvCont-V1_1,
URL: http://www.openmobilealliance.org/

**[DMBOOT]** "OMA Device Management Bootstrap, Version 1.2". Open Mobile Alliance™. OMA-TS-DM_Bootstrap-V1_2.
URL: http://www.openmobilealliance.org

**[DMPROTO1.1.2]** "Device Management Protocol", OMA-TS-DM-Protocol-V1_1_2,
URL:http://www.openmobilealliance.org/

**[DMPROTO1.2]** "Device Management Protocol", OMA-TS-DM-Protocol-V1_2,
URL:http://www.openmobilealliance.org/

**[DMStdObj]** "Device Management Standardized Objects", OMA-TS-DM-StdObj-V1_2,
URL: http://www.openmobilealliance.org/

**[DMTND]** "Device Management Tree and Description", OMA-TS-DM-TND-V1_2,
URL:http://www.openmobilealliance.org/

**[PKCS#15]** "PKCS #15 v1.1: Cryptographic Token Information Syntax Standard", RSA Laboratories, June 6, 2000
URL: ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-15/pkcs-15v1_1.pdf

**[WPRHP]** "OMA Work Programme and Release Handling Processes", Approved Version 2.0, 04 Dec 2007
URL: http://www.openmobilealliance.org/

# 3. Terminology and Conventions

## 3.1 Conventions

This is an informative document, which is not intended to provide testable requirements to implementations.

## 3.2 Definitions

| | |
|---|---|
| **Management Authority** | An entity that has the right to perform a specific Device Management function on a Device or manipulate a given data element or parameter. For example, the Network Operator, handset manufacturer, enterprise, or Device owner may be the authority or share authority for managing the Device. One Management Authority may own all Device resources or may share or delegate all or parts of these with/to other Management Authorities. A DM Server with the appropriate authority to gain access to a DM Client. |
| **Management Object** | A data model for information which is a logical part of the interfaces exposed by DM components. |

## 3.3 Abbreviations

| | |
|---|---|
| **AC** | Application Characteristic |
| **CP** | Client Provisioning |
| **ConnMO** | Connectivity MO |
| **DCMO** | Device Capability MO |
| **DDF** | Device Description Framework |
| **DDS** | Data Definition Specification |
| **DiagMon** | Diagnostics and Monitoring |
| **DM** | Device Management |
| **DSO** | Document Support Officer |
| **DTD** | Document Type Definition |
| **ERELD** | Enabler Release Definition |
| **FUMO** | Firmware Update MO |
| **IANA** | Internet Assigned Numbers Authority |
| **LAWMO** | Lock and Wipe MO |
| **MO** | Management Object |
| **MOI** | Management Object Identifier |
| **OMA** | Open Mobile Alliance |
| **OMNA** | Open Mobile Naming Authority |
| **RRELD** | Reference Release Definition |
| **SC** | Smart Card |
| **SCOMO** | Software Component MO |
| **SUP** | Support Document |
| **URI** | Uniform Resource Identifier |
| **WG** | Working Group |
| **XML** | Extensible Markup Language |

# 4. Introduction

The OMA Device Management (DM) protocol [DMPROTO1.1.2, DMPROTO1.2] supports the notion of Management Objects (MOs). These are abstract representations of remote management capabilities supported by the device. All the available MOs pertaining to a device are organized in a hierarchical tree structure known as the Management Tree. The DM Server performs remote management actions by executing operations on the nodes of the various MOs in the Management Tree. Managing resources on a device entails a two-way communication between the DM Client and the DM Server.

Predating OMA-DM is an older protocol for device management which is known as the OMA Client Provisioning (CP) protocol [CP]. CP entails a one-way push of an XML file, containing the configuration parameters, from the Server to the Client. CP only allows the configuration parameters to be written once and does not allow modification or deletion of existing parameters.

In the CP configuration files, provisioning information is grouped into logical units called Characteristic Elements or Characteristics. The Characteristic Elements define different aspects of the protocol including protocol parameters, connectivity parameters and management capabilities. The management capabilities of a device are handled by the Application Characteristics (ACs) sections of the configuration files. ACs in CP are analogous to MOs in OMA-DM.

This whitepaper documents the best practices for creation of new Management Objects (MO) and Application Characteristics (AC). In addition to the best practices, this whitepaper gives some guidance regarding the processes and structures associated with the MOs and ACs, as well as the object registration process in OMA.

OMA DM specifications define the syntax and semantics of the OMA DM protocol, as well as the syntax for MOs. OMA CP [CP] specifications define the syntax for ACs. It is worth noting that the usefulness of ACs and MOs would be limited if different devices required different data formats and displayed different behaviours.

The creation of MOs and ACs allows Management Authorities to configure and update many settings in a device. To avoid the situation where each mobile device vendor defines a specialized and non-standard arrangement for managing device parameters, this whitepaper tries to present some guidance on the process of creating Management Objects and Application Characteristics to permit the standardized representation of device parameters in mobile devices.

Creating a standardized MO will allow the market to have a standardized way to present the information when using DM. This will avoid interoperability problems, since the information will be presented in a consistent way through the different devices.

## 4.1 Introduction to DM

OMA Device Management consists of two stages:

- Stage One - Bootstrap

  Bootstrap moves a device from an un-provisioned, empty state, to a state where it is able to initiate a management session to a DM Server. DM clients that have already been bootstrapped can be further bootstrapped to enable the device to initiate a management session to new DM servers. In addition to basic connectivity information, device and User Application settings can also be configured during the bootstrap process.

  This can be done in three primary ways:

  - Factory Provisioning: The manufacturer includes the device management Server configuration at the time the device is manufactured.
  - DM Profile: In this case the Server sends to the device directly a Management Object, or the device reads the bootstrap information from the DM message on the available smartcard.
  - OMA CP Profile: In this case the Server sends to the device an AC, which can later be mapped into a MO, or the device reads the bootstrap information from the AC on the available smartcard.

- Stage Two – Continuous Provisioning and Management

  Continuous provisioning and management is the process by which the device is provisioned, via the OMA DM Server, with further information or management after the device is bootstrapped.

Currently, DM technology allows a device to present the information stored on the device to an external Server, in case the external Server has sufficient rights to access the information.

This can be seen in the following picture:

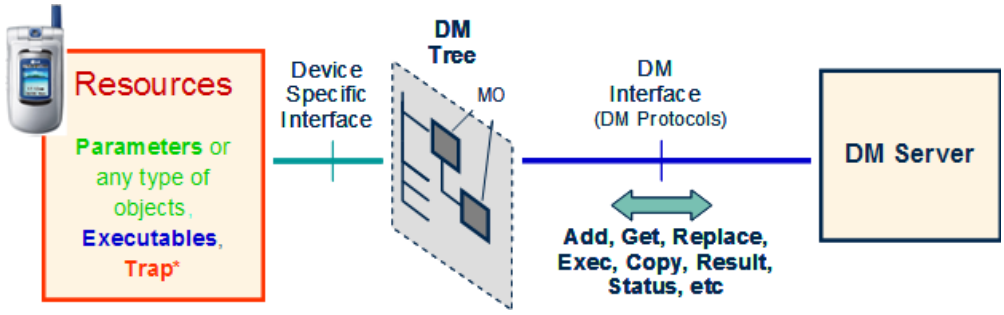


**Figure 1: Management View Exposed by a DM Client to a DM Server**

At the left side of the picture we can find the device, with its internal resources. The device presents part of this information to the Server that is on the right part. The way to do this is through Management Objects at the DM Tree.

A Management Object is a subtree of the management tree which is intended to be a (possibly singleton) collection of nodes which are related in some way, e.g. all the possible connectivity information for one application.

## 4.2 Introduction to Enablers in DM Working Group

As per the OMA Release Handling Processes, OMA releases its work as Enabler Release or Reference Release Packages [WPRHP]. The DM Working Group develops both Enablers and Reference Releases that use fundamental functions provided by the DM Enabler and also provide additional functions to address specific needs in the market. Many of these Enablers and Reference Release Packages define standardized MOs.

The following is a brief description of some of the Enabler Release and Reference Release Packages that have been developed by the DM WG. This description will give the reader some ideas about the kind of management functionality that can be realized through MOs.

- Connectivity MO Reference Release (ConnMO):
    - A Reference Release that provides a set of MOs representing variety connectivity settings that will be used in devices to connect to the different network (GSM, GPRS, UMTS, WLAN, CDMA, etc.).
- Device Capability Enabler (DCMO):
    - An enabler that provides mechanisms to manage device capability components in the device.
    - Allows the Management Authority to enable or disable hardware-related device capabilities in devices.
    - Allows the Management Authority to discover removable hardware in devices.
- Diagnostic and Monitoring Enabler (DiagMon):
    - An enabler that provides framework and mechanisms for Management Authority to run diagnostics and monitoring of devices. The specific diagnostic and monitoring functions are not specified in 1.0 release of this MO enabler.
    - Allows the Management Authority to collect diagnostic data of devices in order to avoid or solve possible problems in the device.
    - Allows the Management Authority to set traps to monitor the device.
- Firmware Update Enabler (FUMO):
    - An enabler that provides mechanism for Management Authority to update the firmware in the device.

- Lock and Wipe Enabler (LAWMO):
    - An enabler that provides mechanism for Management Authority to lock/unlock the device or wipe user data in the device.
- Scheduling Enabler:
    - An enabler that provides mechanisms to schedule management operations in the device for offline processing, when certain time or event is satisfied.

- Smartcard Enabler (SC):
    - An enabler that provides secure dynamic provisioning of MOs available from a smart card
- Software Component Enabler (SCOMO):
    - An enabler that provides mechanisms for Management Authority to manage software components in the device.
    - Allow the Management Authority to manage the software components and their status in devices.
    - Allow the Management Authority to get inventory of software components in the device.

The above MO Enablers and Reference Releases can be classified as follows:

- Data Model - MO enablers that only provide one or more MOs but do not specify associated functions, e.g. ConnMO Reference Release.
- Functional - MO enablers that provide one or more MOs and specify associated functions with well defined behaviour, e.g. SCOMO Enabler, DCMO Enabler.

In addition to MO Enablers and Reference Releases, DM WG has also developed some other enablers. Examples include:

- Device Management Enabler (DM):
    - An enabler that provides management of device configuration and other managed objects of devices.
    - Setting of initial provisioning information in devices.
    - Subsequent updates of persistent information in devices.
    - Retrieval of management information from devices.
    - Processing of events and alerts generated by devices.
- Client Provisioning Enabler (CP):
    - An enabler that allows for an initial setting of provisioning information in devices by downloading XML formatted configuration data files from the Server to the Client

# 5. Which to create? AC or MO or both?

## 5.1 AC Creation

AC files provide a mechanism to declare configuration elements associated with an enabler or other control entity as supported by the OMA -CP enabler. The AC information is made available through the OMNA registry (where the APPID values are registered).

The AC registry can be found at:

http://www.openmobilealliance.org/tech

## 5.2 MO Creation

MOs allow a device to present the configuration of the device in a standardized way, allowing a Server to be able to retrieve and manage the configuration of a device (the parameters included in the MO).

### 5.2.1 Introduction to Management Objects

Management Objects are the data model for information that can be manipulated by management operations carried over the OMA DM protocol. A Management Object node can represent information as small as an integer or as large and complex as a background picture, screen saver or security certificate. The OMA DM protocol is neutral about the contents of the Management Objects and treats the node values as opaque data.

### 5.2.2 MO General Recommendations

Management Objects are processed by the Device Management enabler, and are useful for devices with DM Clients. MOs can be used for initial settings (as with ACs). Additionally, MOs can be used for a Server to be able to proceed with later management of the existing device settings.

When creating a MO for an enabler that requires the use of Generic Alert or any new feature in OMA DM 1.2, the DM version to be used must be OMA DM 1.2 [DMPROTO1.2] or later compatible version. For all other MOs, the DM version to be used must be OMA DM 1.1.2 [DMPROTO1.1.2] or later compatible version.

### 5.2.3 Conversion from AC to MO

While converting from AC to MO, it is recommended to not simply copy the format and data. For example, there is no need to include the AC's APPIDin a MO since this information will be included in the MO Type.

The conversion from AC to MO is a good time to reorganize the data (i.e. ACs do have nesting capability, although it is rarely taken advantage of).

Other recommendations include:

- making sure that the data in the AC does not conflict with the MO as this will be a cause of concern during deployment
- determining what data will be useful for writing only, and what data will be useful for read/write and trying to segregate write-only from read-write data

## 5.3 AC and MO Creation

The need to create both an AC and an MO arises only if it is unclear if the device will run DM, CP or both DM and CP. When sending both an AC and a MO, the decision of which information to use will be left to the implementation, therefore Management Authorities and developers should make sure that the information included in both AC and MO is not contradictory.

# 6. AC Creation for OMA CP

## 6.1 OMNA Registration Items for OMA AC Files

The OMNA registration for AC files is based on the assigned APPIDvalue and includes a derived namespace. The namespace value is structured on the APPIDand enabler names.

Due to the lack of versioning information in the AC file itself, it is not possible to differentiate different versions with a single assigned APPIDvalue. Therefore, new APPIDvalues need to be registered when changes to the AC are needed to support a new version. This will permit the elements to have awareness of the specific configuration set being used. Thus, though the namespace value includes version - that is not intended to differentiate but merely assist with the version awareness.

The format of these namespaces is

**urn:oma:ac:***{APPID}*_*{EnablerName}*:*{VersionText}*

Example namespaces are:

**urn:oma:ac:ap0094_foo:1.0**
**urn:oma:ac:ap0121_bar:1.0**

## 6.2 Guidance for AC Creation within OMA

### 6.2.1 Naming of AC docs

The public reachable AC file names are generally shorter and simpler than the names of the OMA permanent documents used for change management. There is generally no need to have superfluous text in the public file names and it is quite expected to avoid constructs like date strings and status values as well. As a consequence, the following file name construction scheme is used for the 'SUP' files in the permanent document area and public file names.

Perm Doc - **OMA-SUP-AC_***{APPID}*_*{enablerName}*-**V***{versionNumbers}*-**{DateStr}-{State}.txt**
Public - **ac_***{APPID}*-*{enablerName}*-**v***{versionNumbers}*.**txt**

Examples:

**OMA-SUP-AC_ap0094_foo-V1_0-20060123-C.txt ==> ac_ap0094_foo-v1_0.txt**

**OMA-SUP-AC_ap0121_bar-V1_0-20060321-C.txt ==> ac_ap0121_bar-v1_0.txt**

### 6.2.2 Handling of AC docs

The AC SUP file is maintained in the OMA Portal Permanent Document area as with the specifications in the enabler. It is subject to revision using the normal CR process. Application of CRs to the AC SUP file will result in an updated file with the appropriate date associated.

The AC SUP file should be treated as if it has the ".txt" file extension.

References to OMA defined ACs in specifications should be done to the developed SUP file. This will permit easy reference and retrieval of current or past versions using the normal PD file mechanisms. An example reference would look like:

| | |
|---|---|
| [FOO_AC] | "AC for FOO Enabler, Version 1.0", Open Mobile Alliance™, OMA-SUP-AC_ap0094_foo-V1_0, URL:http://www.openmobilealliance.org/ |

### 6.2.3 AC docs in Enabler Releases

As with specifications, ACs may be deliverable of an Enabler Release or a Reference Release. So, similar to how specifications are released, ACs are packaged in the Enabler Release Package or Reference Release Package to be made available. This packaging also includes appropriate linkages in the ERELD or RRELD.

The following is an example AC entry in the ERELD:

| Supporting Files | | |
|---|---|---|
| [FOO_AC] | OMA-SUP-AC_ap0094_foo-V1_0-20060123-C | This AC describes the configuration for the FOO Enabler.<br>Working file in OMNA AC directory:<br>file: ac_ap0094_foo-v1_0.txt<br>path: http://www.openmobilealliance.org/tech/omna/dm_ac/ |

### 6.2.4 Getting AC docs into Public Directory

The primary means of publishing the AC file into the public directory is the advancement of the enabler release to Candidate. At that time, the DSO will review the ERELD and if an AC file is listed, copy the material in the listed SUP file to the named version of the public AC file. This will also result in the OMNA AC registry getting a link to this file if it had not been present in a previous release.

### 6.2.5 Additional Guidance

Do try to make the node names readable and understandable. Shorten the names only when absolutely needed.

# 7. MO Creation for OMA DM

## 7.1 Content Guidance

When creating a MO, the creators should take into account that the aim is to have a standardized way to manage the information present in the device.

It should also be noted that not all the parameters of a standardized MO have to be mandatory. Some information may be relevant for a substantial number of actors, and irrelevant for others. For this case, allowing optional elements lets both groups use the MO without undue burden.

It should also be noted, that in order to decrease interoperability problems, it is better to make sure all the relevant parameters are in the lowest versions of the MO (if there is more than one of this), even if they are optional, since incrementing the number of parameters in a later version could break backward compatibility. When in doubt about a node, add it in.

When creating a MO there are some fields that are recommended.

- Parameters to be used by an application on the device, but that may be changed by the Management Authority.
- Date or Version of the data
    - Information of when the data was last set (this will allow the Management Authority and the device to know which ones are the most updated parameters).
- Connectivity parameters.
    - Connectivity parameters should reuse, if possible, other already existing ConnMOs that may be on the device. This means that in order to allow more than one application the use of the same connectivity parameters (for example in case of using the same Server), the MO should point, as an example, to existing instances of [ConnMO].
    - QoS Parameters
        - These parameters are typically bearer specific. The MO should reference already existing bearer specific QoS parameters that are specified by [ConnMO]. If insufficient, the additional information specific QoS parameters may have to be specified as well.
- Application information.
    - Information of the application that may later be interesting for trouble shooting (by diagnosis and monitoring, customer care, etc).
    - Information of the application that may be interesting to be accessed remotely (by customer care).
- Behaviour: It is also possible to indicate in the MO description part of the behaviour associated to the MO.
- Nodes to be targeted with commands.
    - It is possible to relate a node that would be used to start an action in the device, e.g.: start a firmware download, install a software component on the device.
    - When possible, it is recommended to group such related nodes under (one or more) Operations parent node(s).
- Proprietary Extensions:
    - All MO should have a sub-node allowing a vendor to include any proprietary sub-nodes in order to include any vendor specific extensions. The name of this node should be “Ext”.

If a collection of related nodes within an MO definition is expected to be repeated a certain number of times, it is highly recommended to introduce an unnamed node in the MO definition and have all the nodes subtend from this unnamed node. The name of such nodes is not known in the MO description but it is assigned at run-time. Traditionally unnamed nodes are represented by the letter “*x*”. Needless to say, “*x*” should never be chosen as the name of any node in the MO definition. Sibling instances of “*x*” cannot have the same name and it is also recommended not to have any named nodes as siblings of an “*x*” node, to reduce chances of clashes with instances of “*x*” that will be generated by the client.

The DM Account Management Object [DMStdObj], has three unnamed nodes. These are the child nodes of the ToConRef, AppAddr and AppAuth nodes.

## 7.2 Definition and description of Management Objects

OMA DM Management Objects are defined using the OMA DM Device Description Framework [DMTND]. The use of this description framework produces detailed information about the device in question. However, due to the high level of detail in these descriptions, they are sometimes hard for humans to digest and it can be a time consuming task to get an overview of a particular object's structure.

In order to make it easier to quickly get an overview of how a Management Object is organized and its intended use, a simplified graphical notation in the shape of a block diagram is used in this document. Even though the notation is graphical, it still uses some printable characters, e.g. to denote the number of occurrences of a node. These are mainly borrowed from the syntax of DTDs for XML. The characters and their meaning are defined in the following table.

| Character | Meaning |
|---|---|
| + | one or many occurrences |
| * | zero or more occurrences |
| ? | zero or one occurrences |

If none of these characters is used, then the default occurrence is exactly once.

There is one more feature of the DDF that needs to have a corresponding graphical notation, the un-named block. These are blocks that act as placeholders in the description and are instantiated with information when the nodes are used at run-time. Un-named blocks in the description are represented by a lower case character in italics, e.g. *x*.

Each block in the graphical notation corresponds to a described node, and the text is the name of the node. If a block contains an *x,* it means that the corresponding node is unnamed. The names of all ancestral nodes are used to construct the URI for each node in the Management Object. It is not possible to see the actual parameters, or data, stored in the nodes by looking at the graphical notation of a Management Object.

Figure 1 depicts an illustrative management object in graphical notation. There is one unnamed node under the node labelled *DiagMonData* which has one or more occurrences. The node labelled *Description* has zero or one occurrence. The placeholder node for proprietary extensions, labelled *Ext*, also has zero or one occurrence. All other nodes have an occurrence of one.

Details regarding the graphical notation can be found in [DMStdObj].

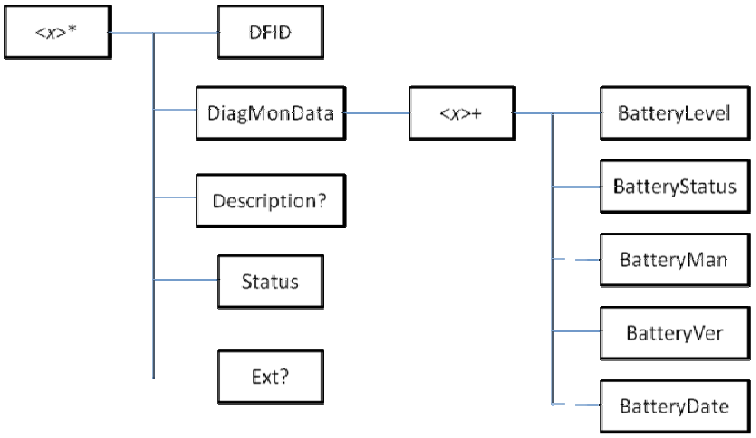


**Figure 2: MO depicted in graphical notation**

An enhanced graphical notation can distinguish between interior and leaf nodes in the MO, as well as between Required and Optional nodes. Figure New-2 <FIXME!> depicts the illustrative management object in enhanced graphical notation.

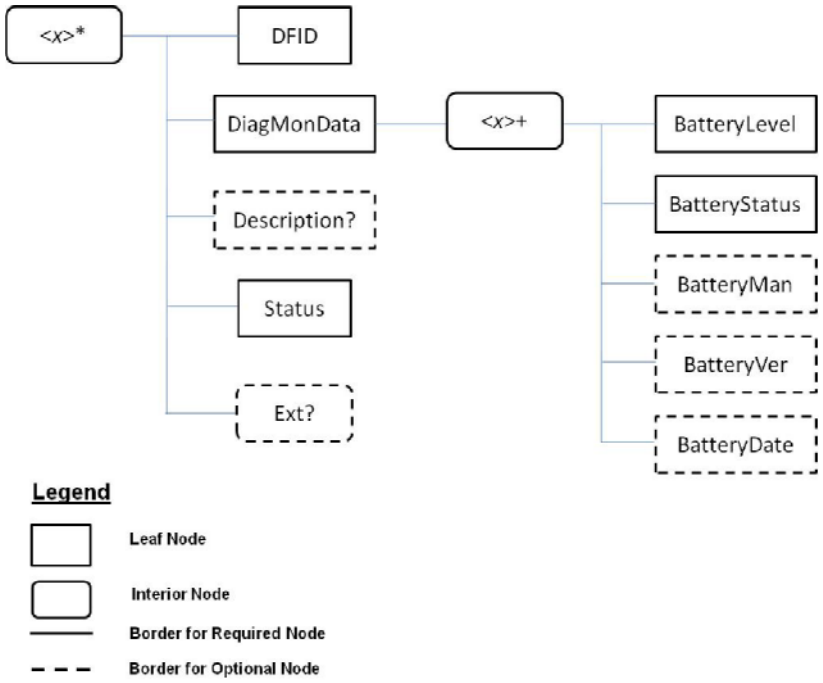


**Figure 3: MO depicted in enhanced graphical notation**

Interior nodes are shown with rounded rectangles while leaf nodes are shown with normal rectangles. Additionally, all the Required nodes have a solid border while the Optional nodes have a dotted border.

## 7.2.1 Definition and Description of Nodes

Nodes can be described in any number of ways. OMA DM has experimented with various layouts, and the current preference for describing a node looks like this (examples are based on the ConnMO Reference Release):

**ID**

| Status | Occurrence | Format | Min. Access Types |
|---|---|---|---|
| Required | One | chr | Get |

This leaf node defines the identity of the one specific network access point which an instance of this management object represents.

**CallType**

| Status | Occurrence | Format | Min. Access Types |
|---|---|---|---|
| Optional | ZeroOrOne | chr | Get |

The CallType parameter is used to define protocol to be used for data exchange or call type. If the parameter is not present or if no value is given, the default value ANALOG-MODEM SHOULD be assumed.

| CallType | **Description** |
|---|---|
| ANALOG-MODEM | ANALOG-MODEM (default) in use [3GPP22.002-AsynchNT]. |
| V120 | V.120 in use [3GPP22.002-AsynchNTD]. |
| V110 | V.110 in use [3GPP22.002-AsynchNTD]. |
| X31 | X.31 in use [3GPP22.002-FMT]. |
| BIT-TRANSPARENT | BIT-TRANSPARENT in use [3GPP22.002-360-BT]. |

Each node in the MO definition is characterized by four node description characteristics: *Status*, *Occurrence*, *Format* and *Minimum Set of Access Types*. These node description characteristics are described further on in this whitepaper.

## 7.3 Access Type Guidance

When deciding upon access type guidance, the recommendation from OMA DM is as a general rule to grant only *Get* access type on the various nodes within the standardized MO definition, except in some special cases, where other access types may be required.

The allowed values for access type are any combination of the following:

- *Get*
- *Replace*
- *Exec*
- *Add*
- *Delete*
- *Copy*

In some instances there may be a need to explicitly disallow a certain type of access for a given node. This is indicated by prepending the access type with the keyword "*No*". For example, it is strongly recommended that any node that deals with security related information, such as user-id and password, should not be granted the *Get* or *Copy* access type. The *Min. Access Types* for such a node should include *No Get* and *No Copy*.

The *Add* and *Delete* access types are generally provided for unnamed nodes for which the Occurrence is one of the following: *ZeroOrOne*, *ZeroOrMore* or *ZeroOrN*.

It is a good practice to grant an optional node the *Get* access type, in addition to other access types that it may support. For example, an optional node that supports the *Exec* access type should have its Min. Access Types set to "*Get, Exec*" whereas a required node that supports the *Exec* access type should have its Min. Access Types set to "*Exec*" only.

It needs to be noted that specifying only Get access type for a node within the MO does not restrict an implementation from supporting additional access types for that node since the standardized MO only specifies the minimum set of access types that the node is required to support.

## 7.4 Status and Occurrence Guidance

When creating a MO, the creators need to decide on the value of the Status and Occurrence of each node. For that purpose, this section describes the definitions and some examples on how Status and Occurrence work:

- Status:
    - Definition:
        - The Status definition in the node definitions indicates if the Client must support that node or not.
        - If the Status is “Required” even though the node may not be present at that time, the Server can expect the Client to be able to support it.
        - If the Status is “Required” then the Client must support that node in the case the Client supports the parent node. If the Status is “Optional” then this should be reflected in DDF file to show whether the node is supported. If the Status is “Required” then the Client must support that node in the case the Client support the parent node.
        - When creating the Status of an MO, the child may be Required, while the parent node may be Optional. This would mean that all those elements would be Optional, but in case the parent node is present, then those child nodes would be Required.
        - The Status of the top (or root) node in an MO definition is always “Required”.
    - Possible Values: The value of this parameter can be "Required" or “Optional”.
- Occurrence:
    - Definition: The Occurrence element specifies the allowed number of instances for a node within the subtree that is rooted at its parent node.
    - Possible Values:
        - ZeroOrOne
        - ZeroOrMore
        - ZeroOrN (N is a character sting that represents a positive integer value between 2 and 65536)
        - One
        - OneOrN (N is a character sting that represents a positive integer value between 2 and 65536)
        - OneOrMore

The DM TND specification [DMTND] requires that each node in a management tree must have a unique URI and so it is not possible for two nodes with the same parent node to have the same name. Consequently, the Occurrence values ZeroOrMore, OneOrMore, ZeroOrN and OneOrN cannot be used for named nodes.

## 7.5 Format Guidance

When creating a MO, the creators need to decide on the Format of each node.

The Format property always maintains information about the data format of the current node value. The possible Format values are listed as below: `b64 | bin | bool | chr | int | node | null | xml | date |time |float.`

Note that interior nodes must have “`node`” as the Format value.

If a leaf node has no content information and it will never have any value, the Format value can be set as “null”. For example, when a leaf node serves only as the target of an Exec command, the Format value can be set as “null”. That is, it is never intended to have a value.

## 7.6 MO Generation

Management Objects are created on the device in one of three ways:

- Fully static MOs — the Server cannot create or delete these MOs. The MOs are fully specified in static form by the Client vendor. Node values within the MOs can be replaced, but nodes within the MOs cannot be added or removed.
- Fully dynamic MOs are created node-by-node typically by the Server (but could sometimes be created by the Client, with notification to the Server) at run-time.
- Hybrid MOs— after the dynamic creation of a parent node by the Server, the device automatically creates a factory-provisioned set of sub-nodes, and may fill in some default data as well. The static parts of these MOs are the automatically created sub-nodes and the default data (which is up to the implementation). This model may also be used within an MO to provide for repeating groups of related nodes.

Devices will likely employ all three models.

Device manufacturers will decide if they implement nodes as static or dynamic. They will publish which method they have used in the corresponding DDF file for all nodes. It is common to specify the minimum access type to only GET since the commands REPLACE, ADD and DELETE are dependent of the device manufacturers’ implementation choice.

Some examples of these models:

- DevInfo/DevDetail is probably fully static for most vendors.
- DataSync is an example of a fully dynamic sub-tree — servers create each node and set the proper values in order for the device to make use of the configuration.
- E-Mail and DMAcc are examples where the hybrid model makes most sense: A Server adds a new e-mail account dynamic node and the device automatically creates all sub-nodes with appropriate default values. Server then uses Replace commands to adjust the values as needed.

## 7.7 OMNA Registration Items for MO DDF Files

The OMNA registration for MO DDF files records the name for the MO in two of three name schemes. The three defined name schemes are:

| MO Label | Description |
|---|---|
| oma_label | Assignments of values to MOs defined by OMA Work Groups<br>These will have namespaces assigned by OMNA (see below) |
| ext_label | Assignments of values to MOs defined by external entities<br>These will have namespaces assigned by the external group |
| x_label | Values that are used for testing or private use<br>These will not be recorded by OMNA |

The format of the namespaces registered by OMA Work Groups is:

**urn:oma:mo:oma**_{Label}:{VersionText}

Example namespaces are:

**urn:oma:mo**:oma_foo:1.0
**urn:oma:mo**:oma_bar:1.0

# 7.8 Guidance for MO Creation within OMA

## 7.8.1 Naming of MO DDF docs

The public reachable MO DDF file names are generally shorter and simpler than the names of the OMA permanent documents used for change management. There is generally no need to have superfluous text in the public file names and it is quite expected to avoid constructs like date strings and status values as well. As a consequence, the following file name construction scheme is used for the 'SUP' files in the permanent document area and public file names.

Perm Doc - OMA-SUP-MO_oma_{Label}-V{versionNumbers}-{DateStr}-{State}.txt
Public - oma_{Label}-v{versionNumbers}.ddf

Examples:

OMA-SUP-MO_oma_foo-V1_0-20070123-C.txt ==> oma_foo-v1_0.ddf
OMA-SUP-MO_oma_bar-V1_0-20070321-C.txt ==> oma_bar-v1_0.ddf

## 7.8.2 Handling of MO DDF docs

The DDF file is maintained in the OMA Portal Permanent Document area as with the specifications in the enabler. It is subject to revision using the normal CR process. Application of CRs to the DDF file will result in an updated file with the appropriate date associated.

The DDF permanent document file should be treated as if it has the ".txt" file extension. When the public reachable file is loaded, it will be loaded with the ".ddf" extension.

References to OMA defined DDF files in specifications should be done to the developed supporting file. An example reference would look like:

| | |
|---|---|
| [FOO_MO_DDF] | "MO DDF for FOO Enabler", Version 1.0, Open Mobile Alliance™, OMA-SUP-MO_oma_foo-V1_0, URL:http://www.openmobilealliance.org/ |

## 7.8.3 MO specifications and DDF Files in Enabler Releases

As with specifications, MO specifications and DDF files may be deliverable of an Enabler Release or a Reference Release. So, similar to how specifications are released, MO specifications and DDF files are packaged in the Enabler Release Package or Reference Release Package to be made available. This packaging also includes appropriate linkages in the ERELD or RRELD.

The reference release procedure will be used to develop Data Model MO enablers. The enabler release procedure, defined in [WPRHP], will be used to develop Functional MO Enablers.

The following is an example MO DDF file entry in the ERELD:

| Supporting Files | | |
|---|---|---|
| [FOO_MO_DDF] | OMA-SUP-MO_oma_foo-V1_0-20070123-C | This MO DDF describes the configuration for the FOO Enabler.<br>Working file in OMNA MO directory:<br>file: oma_foo-v1_0.ddf<br>path: http://www.openmobilealliance.org/tech/omna/dm_mo/ |

# 8. OMNA Registration

## 8.1 AC Registration for OMA CP

OMNA maintains a registry of values used for APPID which is used in the Application Characteristic (AC) descriptions. These APPID registrations may be made by OMA Working Groups or External entities. In all cases, the registry provides for allocation of the needed 'APPID' value and serves as a repository for the AC descriptions. These AC descriptions are for information.

The APPID registry is divided into four sets of labels: three named collections which are managed by OMNA and one in support of IANA ports. These labels are described in the following table:

| Range | Description |
|---|---|
| ap0001-ap1999 | Assignments of APPIDs for Enablers defined in OMA |
| ap2001-ap5999 | Assignments of APPIDs for Externally defined Application entities |
| w1-wA | Legacy assignments of APPID from the former WAP Forum (based on earlier assignment rules) |
| *numeric* | Assignments associated with registered ports (IANA) |

For registered APPID, link to the AC descriptions and other information about the AC descriptions are provided on the OMNA webpage.

To request an APPID registration, it is necessary to make a request and provide a copy of the AC that would be associated to the APPID. The description will be reviewed by the OMA Device Management (DM) Working Group for conformance with the specifications. . If this passes, it will be assigned an APPID and registered on the OMNA page. The form for submitting a request for registration is available through the OMNA pages of the OMA website.

## 8.2 MO Registration for OMA DM

OMNA also maintains a registry of MOs. The MO registry can be found at:

http://www.openmobilealliance.org/tech

Registering a MO is an important task. It allows a party to associate a Management Object identifier (MOI) to a specific MO and to let the rest of the world know about that MO. The MOI registrations may be made by OMA Working Groups or external organizations.

Registering consists in gathering all information relevant to a MO in a document that will later be referenced that will later be posted in the OMNA website.

In order to register a MO there is a process that needs to be followed. Please note that vendors registering MOs need only to do step 6:

1. OMA Working Groups must use the latest support document template as the baseline for the MO DDF (OMA-Template-SUPgeneric-YYYYMMDD-I.txt). Non OMA parties are encouraged to use this template as well.
2. Create MO specification. OMA Working Groups must use a standalone document for the MO using the latest TS template or DDS template Typically the Data Model MO will use the DDS template. However, one enabler may contain two functional MOs and one data model MO, then all documents could be based on the TS template. Non OMA parties are encouraged to use these templates as well.
3. Other Standard bodies are highly recommended to send the MO for review to the OMA DM WG. Please note that the MO will not be agreed by the OMA WG, just reviewed for compliance with DM protocol. This review will not have any official status, but will help determine potential problems with the MO.
4. Wait for the OMA review of the document.

5. Modify the MO and the associated DDF according to the comments from the OMA DM WG (if any comments have been received).
6. Complete the electronic form on the OMNA MO Request page for submission. The linkage for this page is http://www.openmobilealliance.org/tech/omna. The required information is for any party registering an MO is:
    a. Name of the submitter.
    b. Email address of the submitter.
    c. URL of MO DDF file (optional).
    d. URL of MO specification that defines the Management Object.
    e. Requested MO registration identifier.
    f. Short description of the MO.

A copy of the MO DDF file needs to be sent to the OMNA Secretary. After some review, the OMNA Secretary will revise the documentation and most likely assign the requested MOI to the MO.

At this moment, the registration will be considered complete and all the relevant information (all parameter listed on top plus the assigned MOI) will be posted at the OMA-OMNA website.

OMNA maintains a registry of values used for Managed Object (MO) descriptions. In all cases, the registry provides for allocation of the needed MO URN value and serves as a repository for the MO descriptions. The linkage for the registry is http://www.openmobilealliance.org/tech/omna/omna-dm_mo-registry.htm

The MO registry is divided into three segments: two named collections which are managed by OMNA (one for OMA WG defined objects and one for external) and one undefined set for testing or private use. These labels are described in the following table:

| Range | Description |
|---|---|
| oma-*label* | Assignments of values to MOs defined by OMA Work Groups |
| ext-*label* | Assignments of values to MOs defined by external organizations |
| x-*label* | Values that are used for testing or private use - will not be recorded |

Here are some examples for URN based MOI for information:

| MO Value | MO Identifier |
|---|---|
| oma-dm-devinfo | urn:oma:mo:oma-dm-devinfo:1.0 |
| ext-3gpp-vcc | urn:oma:mo:ext-3gpp-vcc:1.0 |
| x-private-test | urn:oma:mo:x-private-test:1.0 |

# 9. Smart Cards

Smart cards provide a way to initialize devices independently of the bootstrap profile supported as described in [DMBOOT]. The following sections provide recommendations on the different matters to be considered during the creation of Application Characteristics and/or Management Objects when smart cards are involved.

## 9.2 Data storage for bootstrap information

The parameters exposed to a remote Server by the means of a Management Object can be initialized using smart cards. The initialization through a smart card can be accomplished in different ways:

a) Including bootstrap information (i.e. MOs) in the DM bootstrap file as defined in [DMBOOT].

b) Including bootstrap information (i.e. ACs) in the CP bootstrap file as defined in [DMBOOT].

## 9.3 Interoperability and backward compatibility

### 9.3.1 Smart Cards personalization

If smart cards are expected to work on devices independently of the bootstrap profile (i.e. CP or DM) it is recommended to personalize the smart cards with a single PKCS#15 [PKCS#15] structure able to support both profiles.

# Appendix A. Change History (Informative)

| Document Identifier | Date | Sections | Description |
|---|---|---|---|
| OMA-WP-AC_MO-20081024-A | 24 Oct 2008 | n/a | Approved by OMA TP<br>OMA-TP-2008-0406R01-<br>INP_AC_MO_V1_0_RRP_for_Notification_and_Final_Approval.doc |