Software Component Management Object (SCOMO)

Candidate Version 1.1 – 22 Nov 2011

Open Mobile Alliance

OMA-ER-SCOMO-V1_1-20111122-C

Contents

1. Scope

The document includes the Requirements, Architecture, and Technical Specification for SCOMO 1.1.

2. References

2.1 Normative References

[DMPRO]	"OMA Device Management Protocol, Version 1.2". Open Mobile Alliance™. OMA-TS-DM_Protocol-V1_2. URL:http://www.openmobilealliance.org/
[DMREPRO]	"OMA Device Management Representation Protocol, Version 1.2". Open Mobile Alliance™. OMA-TS-DM_RepPro-V1_2. URL:http://www.openmobilealliance.org/
[DMTND]	"OMA Device Management Tree and Description, Version 1.2". Open Mobile Alliance™. OMA-TS-DM-TND-V1_2 URL:http://www.openmobilealliance.org/
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt
[RFC3986]	T. Berners-Lee, R. Fielding and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", IETF RFC 3986, August 1998. URL:http://www.ietf.org/rfc/rfc3986.txt
[SCRRULES]	"SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

2.2 Informative References

[DLOTA]	"Generic Content Download Over The Air Specification Version 1.0", Open Mobile Alliance™, OMA-Download-OTA-v1_0, URL:http://www.openmobilealliance.org/
[DMDICT]	" OMA Device Management Dictionary", Draft Version 1.0, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/
[OMADICT]	"Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/
[RFC2616]	"Hypertext Transfer Protocol – HTTP/1.1". Network Working group. June 1999. URL:http://www.ietf.org/rfc/rfc2616.txt
[SSL 3.0]	The SSL Protocol, Version 3.0, <draft-freier-ssl-version3-02.txt>, Transport Layer Security Working Group, Alan O. Freier et al, November 18, 1996. URL:http://www.netscape.com/eng/ssl3/draft302.txt
[TLS 1.0]	"The TLS Protocol Version 1.0", T. Dierks, C. Allen, January 1999. URL:http://www.ietf.org/rfc/rfc2246.txt

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Kindly consult [DMDICT] and [OMADICT] for all definitions used in this document.

3.3 Abbreviations

Kindly consult [DMDICT] and [OMADICT] for all abbreviations used in this document.

4. Introduction

The Software Component Management Object (SCOMO) defined in this specification enables remote Software Component management within a Device based on OMA DM protocol [DMPRO]. SCOMO can be used, for example, to update antivirus software, browser plug-ins, etc.

4.1 Version 1.0

Management operations defined by SCOMO 1.0 for the purpose of Software Component management on the Device support delivery, download, installation, update, removal, activation, and de-activation. In addition, retrieval of inventory of Software Components on the Device is also supported. The inventory includes software components delivered via SCOMO and may include those that are installed outside of SCOMO (for example, at the factory or by the end user).

4.2 Version 1.1

SCOMO 1.1 supports all the management operations defined by SCOMO 1.0. In addition, it supports the following:

- Improved support in the inventory for components installed outside of SCOMO.
- Improved inventory control through supporting extensive metadata on Software Components, installed location, size, blacklisting, whitelisting, etc.
- Support for Firmware update following the same approach as Software Component update.
- User- and Device-initiated check and update of new version of Software Components or firmware.

5. Support for MO dependencies of Software Components Requirements (Normative)

5.1 High-Level Functional Requirements

Label	Description	Release
SCOMO1_1-HLF-01	The enabler SHALL support a capability to reboot the device before and after a Software Component is installed.	1.1
SCOMO1_1-HLF-02	The enabler SHALL support the reporting of all installed Software Components, regardless of whether the Software Components were installed with SCOMO or not.	1.1
SCOMO1_1-HLF-03	The enabler SHALL support reporting the size of installed Software Components.	1.1
SCOMO1_1-HLF-04	The enabler SHALL support reporting the installed location of the installed Software Components.	1.1
SCOMO1_1-HLF-05	The enabler SHALL support reporting the execution environment of the installed Software Components.	1.1
SCOMO1_1-HLF-06	The enabler SHALL support the installation of firmware.	1.1
SCOMO1_1-HLF-07	The enabler SHOULD support Software Component dependency lists.	Postponed to next SCOMO release
SCOMO1_1-HLF-08	The enabler SHALL support client-initiated check for new version for firmware or a specific Software Component.	1.1
SCOMO1_1-HLF-09	The enabler SHALL support client-initiated update for a specific Software Component.	1.1
SCOMO1_1-HLF-10	The enabler SHALL support client-initiated update for firmware.	1.1
SCOMO1_1-HLF-11	The enabler SHALL support a mechanism to configure the Software Component after it is installed.	Postponed to next SCOMO release
SCOMO1_1-HLF-12	The enabler SHALL support a mechanism to associate the Software Component with the corresponding MO if it is present.	1.1
SCOMO1_1-HLF-13	The enabler SHALL provide metadata of Software Components.	1.1
SCOMO1_1-HLF-14	The enabler SHALL support user interaction mechanism.	1.1
SCOMO1_1-HLF-15	The enabler SHALL support parameters used for the installation of a Software Component.	1.1
SCOMO1_1-HLF-16	The enabler SHALL support firmware update.	1.1
SCOMO1_1-HLF-17	The SCOMO enabler SHALL support a mechanism to query the inventory by metadata.	1.1
SCOMO1_1-HLF-18	The SCOMO enabler SHALL support a mechanism to verify Software Component dependencies.	Postponed to next SCOMO release
SCOMO1_1-HLF-19	The SCOMO enabler SHOULD support configuration data for Software Component.	1.1
SCOMO1_1-HLF-20	The SCOMO enabler SHALL support a mechanism for client-initiated check for new version of Software Component or firmware.	1.1
SCOMO1_1-HLF-21	The SCOMO enabler SHALL support a mechanism for configuring the frequency for client-initiated check for new version of Software Component or firmware.	1.1
SCOMO1_1-HLF-22	The SCOMO enabler SHALL support indicating whether a Software Component has been installed via SCOMO or via other means.	1.1
SCOMO1_1-HLF-23	The SCOMO enabler SHALL support the blacklisting of Software Components.	1.1
SCOMO1_1-HLF-24	The SCOMO enabler SHALL support the reporting of installed Software	1.1

	Components that are on the blacklist.	
SCOMO1_1-HLF-25	The SCOMO enabler SHALL support a mechanism for the client-initiated request of Software Components metadata.	1.1
SCOMO1_1-HLF-26	The SCOMO enabler SHALL support a mechanism for the client to initiate a request to download and/or install a specific Software Component.	1.1
SCOMO1_1-HLF-27	The SCOMO enabler SHALL support a mechanism for client-initiated check and update for new version of Software Component or firmware periodically.	1.1
SCOMO1_1-HLF-28	The SCOMO enabler SHALL support the whitelisting of Software Components.	1.1
SCOMO1_1-HLF-29	The SCOMO enabler SHALL support the reporting of installed Software Components that are on the whitelist.	1.1

5.2 Server Initiated Download Requirements

Label	Description	Release
SCOMO1_1- SID-01	The SCOMO enabler SHALL allow alternate download mechanisms to be used.	1.1

6. Architectural Model

6.1 Dependencies

The Software Component architecture diagram indicates dependencies on the OMA DM architecture. It also optionally depends upon other download technology, such as OMA Download [DLOTA] or HTTP.

6.2 Architectural Diagram

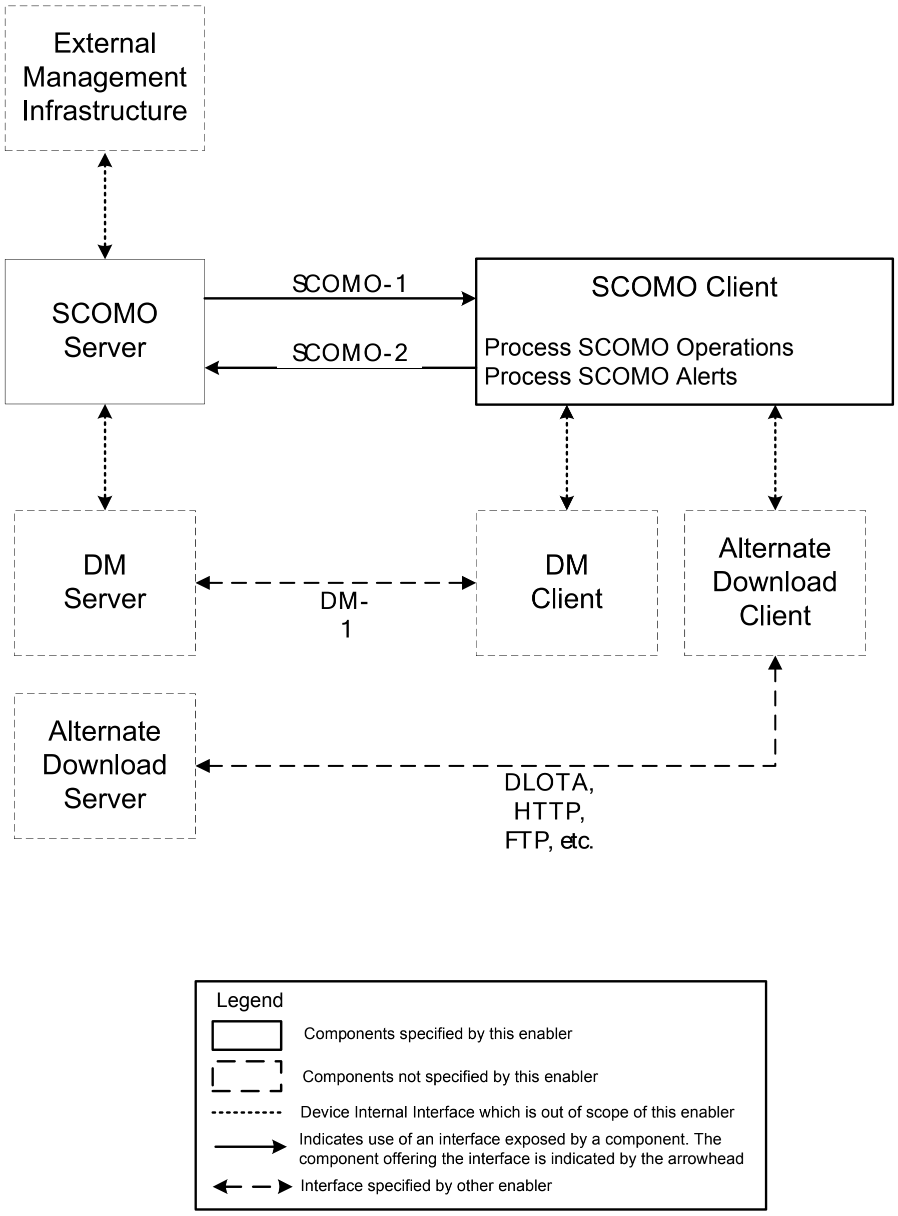
External Management Infrastructure
SCOMO Server
SCOMO-1
SCOMO-2
SCOMO Client
Process SCOMO Operations
Process SCOMO Alerts
DM Server
DM-1
DM Client
Alternate Download Client
Alternate Download Server
DLOTA, HTTP, FTP, etc.
Legend
Components specified by this enabler
Components not specified by this enabler
Device Internal Interface which is out of scope of this enabler
Indicates use of an interface exposed by a component. The component offering the interface is indicated by the arrowhead
Interface specified by other enabler

Figure 1: SCOMO Architecture Model

6.3 Functional Components and Interfaces/Reference Points Definition

6.3.1 Components

6.3.1.1 SCOMO Server

The SCOMO Server is a logical entity that is dedicated to issue SCOMO operations to the device or consume the SCOMO Alerts from the device.

6.3.1.2 SCOMO Client

The SCOMO Client is responsible for executing SCOMO operations. It consumes the Software Component delivered to the device and is expected to relay SCOMO Alerts conveying a success or failure result back to the SCOMO Server.

6.3.1.3 The Device Management Client

The DM Client component makes it possible to initiate Software Component management in the device from a DM Server. The DM Enabler provides support for device discovery and parameter setup by the DM Client component. The SCOMO enabler provides a management object for Software Components that the DM Client component provides access to, such that the DM Server can manipulate it. The DM Client interacts with a Software Component Agent in the device that is responsible for conducting the management activities using a delivered Software Component management object. The DM Client employs the Generic Alert [DMPRO] mechanism to communicate the final notification comprising the status of the management activity.

6.3.1.4 Alternate Download Client

The alternate download client is an optional feature that may exist on the device and is used for downloading Software Components to the device when the [DMPRO] is not being used for such downloads. The alternate download client may support DLOTA or some other mechanism. SCOMO does not define or specify the alternate download client.

6.3.1.5 The Device Management Server

The SCOMO architecture requires the DM Server component to support device discovery, determination of an appropriate Software Component and delivery of a Software Component to the device over Large Object downloads if the device can support that. It also facilitates receipt of a final notification from the device employing the Generic Alert mechanism.

6.3.1.6 The Alternate Download Server

The Alternate Download Server is an optional feature of the Device Management System that makes it possible to download Software Components using the alternate download mechanism, such as DLOTA. The Alternate Download Server is not defined or specified within the scope of SCOMO.

6.3.1.7 External Management Infrastructure

The Device Management System is comprised of a set of external management components over and above the DM Server that participate in the overall process of managing devices. The external management infrastructure is used but not defined or specified within the scope of SCOMO.

6.3.2 Interfaces

6.3.2.1 The SCOMO-1 Interface

The SCOMO-1 interface allows the SCOMO Server to invoke SCOMO operations on the device using the underlying DM-1 interface.

6.3.2.2 The SCOMO-2 Interface

The SCOMO-2 interface allows the SCOMO Client to send SCOMO Alerts to the SCOMO Server using the underlying DM-1 interface.

6.3.2.3 The DM-1 Interface

The DM-1 interface is defined in the OMA DM Enabler and is the subject of those specifications. It provides a formal interface over which Servers may send Device Management commands to Clients and Clients may return status and Alerts to Servers.

7. Software Component Management Framework

7.1 Deployment Component

Deployment Component is a Software Component that is dedicated for management purposes. Deployment Component is execution environment neutral so that it MAY be used to describe the Software Components of different platforms.

Deployment Component MAY include any type of software asset as Software Components, such as executables, applications, libraries, UI-elements, certificates, licenses, firmware, etc.

Deployment Component consists of the Software Component and metadata. The metadata consists of multiple attributes (for example, component id, name, version, etc.) associated with Software Components. The metadata MUST be provided by the Device Management System and the same values MUST be placed in corresponding nodes under Deployed sub-tree by the SCOMO Client after installation. The metadata MUST be delivered within a Delivery Package.

7.1.1 Deployment Component State Machine and Primitives

Deployment Component management Primitives trigger transitions from one state to another. The Device internal operations and state transitions MUST appear to be atomic. If a state transition fails, the SCOMO Client MUST reverse the operation and make sure Deployment Components remain in the previous state.

Each Deployment Component MUST have one of the three states as follows:

a) Inactive state (reflected in the State node in the Deployed branch)

b) Active state (reflected in the State node in the Deployed branch)

c) Removed state (a logical state reflected by deleting the <x> sub-tree representing the Software Component from the Deployed branch)

The following figure depicts the state transition triggered by Deployment Component Primitives executed:

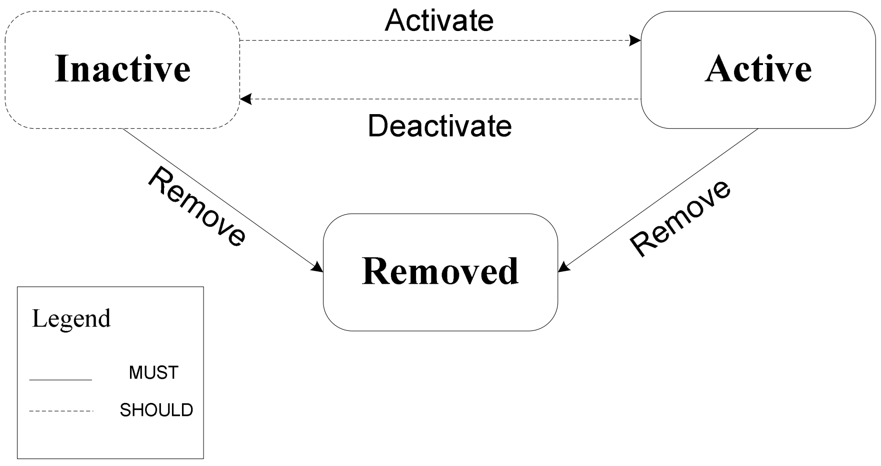

Figure 2: Deployment Component State Machine

7.1.1.1 Inactive state

In Inactive state, the Deployment Component has been installed into the execution environment but services or resources it embodies are not accessible to other entities or resources (including end-user).

The main goal of Inactive state is to minimize the downtime of Deployment Component management operations. Interference with external events (for example, some end-user actions) could disturb or even block some management tasks. Inactive state is a powerful concept when implementing fault tolerant systems since it enables controlled management operations (for example, safe removals of Deployment Components from runtime environment).

The SCOMO Client SHOULD support this state.

7.1.1.2 Active State

In Active state, the Deployment Component has been installed into the execution environment and services or resources it embodies are accessible to other entities or resources (including end user).

A service that consists of multiple Deployment Components is ready for launch after all the relevant Deployment Components have reached the Active state.

The SCOMO Client MUST support this state.

7.1.1.3 Removed State

In Removed state, the Deployment Component has been removed from the Device and services or resources it embodies are no longer accessible to other entities or resources (including end user). The SCOMO Client MUST remove the Software Component from the Device and remove corresponding nodes of the Deployment Component from Deployed tree.

The SCOMO Client MUST support this state.

7.1.1.4 Deployment Component Primitives

The Exec command semantics related to Primitives are described in Chapter 7.1.

Deployment Component Primitives are used to perform management process and triggers state transition of a Deployment Component.

Primitive	Description	Applicable states	Post-Primitive state	Primitive support
Remove	This Primitive is used to remove a Deployment Component from the Device.	Active Inactive	Removed	MUST be supported
Activate	This Primitive is used to move a Deployment Component from the Inactive state to the Active state.	Inactive	Active	SHOULD be supported
Deactivate	This Primitive is used to move a Deployment Component from the Active state to the Inactive state.	Active	Inactive	SHOULD be supported

Table 1 – Deployment Component Primitives

7.1.2 Deployment Component Delivery

Deployment Components MUST be delivered to the Device in a Delivery Package described in section 7.2.

7.2 Delivery Package

It could be desirable to deliver multiple Software Components in a single download operation. The Delivery Package abstraction represents platform-specific package formats used to deliver Software Components to the Device. A single

Delivery Package MAY contain one or more Deployment Components. The format and structure of Delivery Package are out of scope of this specification.

The Client MUST support at least one of the following methods for delivery of Delivery Packages:

- Direct delivery: Using OMA DM Replace command. The OMA DM Large Object mechanism MAY also be used.
- Indirect delivery: Using alternate download mechanism, such as [DLOTA].

The Delivery Package MAY be installed into the Device outside of SCOMO (for example, at the factory or by the end user). In this case, the Client SHALL include the information for Deployment Components installed outside of SCOMO in Inventory sub-tree.

7.2.1 Delivery Package State Machine and Primitives

Delivery Package management Primitives trigger transitions from one state to another. The Device internal operations and state transitions MUST appear to be atomic. If a state transition fails, the SCOMO Client MUST reverse the operation and make sure Delivery Package remains in the previous state.

Two kinds of Primitives are defined as below:

- Atomic Primitives: Download, Install, InstallInactive
- Composed Primitives: DownloadInstall, DownloadInstallInactive

When a Composed Primitive is executed, two state transitions happen in the Device. For example, if DownloadInstall is executed, a Deployment Component transits from Not Downloaded State to Delivered State after a successful download procedure. It transits to Active State after successful installation procedure. If the latter processing fails, it remains in previous state and the second state transition does not happen.

Each Delivery Package MUST have one of the four states as follows:

a) Not Downloaded state (a logical state, reflected by not having any sub-tree for the package or having it in the Download branch)

b) Delivered state (reflected in the State node in the Delivered branch)

c) Installed state (reflected in the State node in the Delivered branch)

d) Removed state (a logical state reflected by deleting the <x> sub-tree representing the Delivery Package from the Delivered branch)

The SCOMO Server MAY indicate to the SCOMO Client that it needs to reboot before install, after install, or before and after install.

The following figure depicts the state transition triggered by Delivery Package Primitives executed:

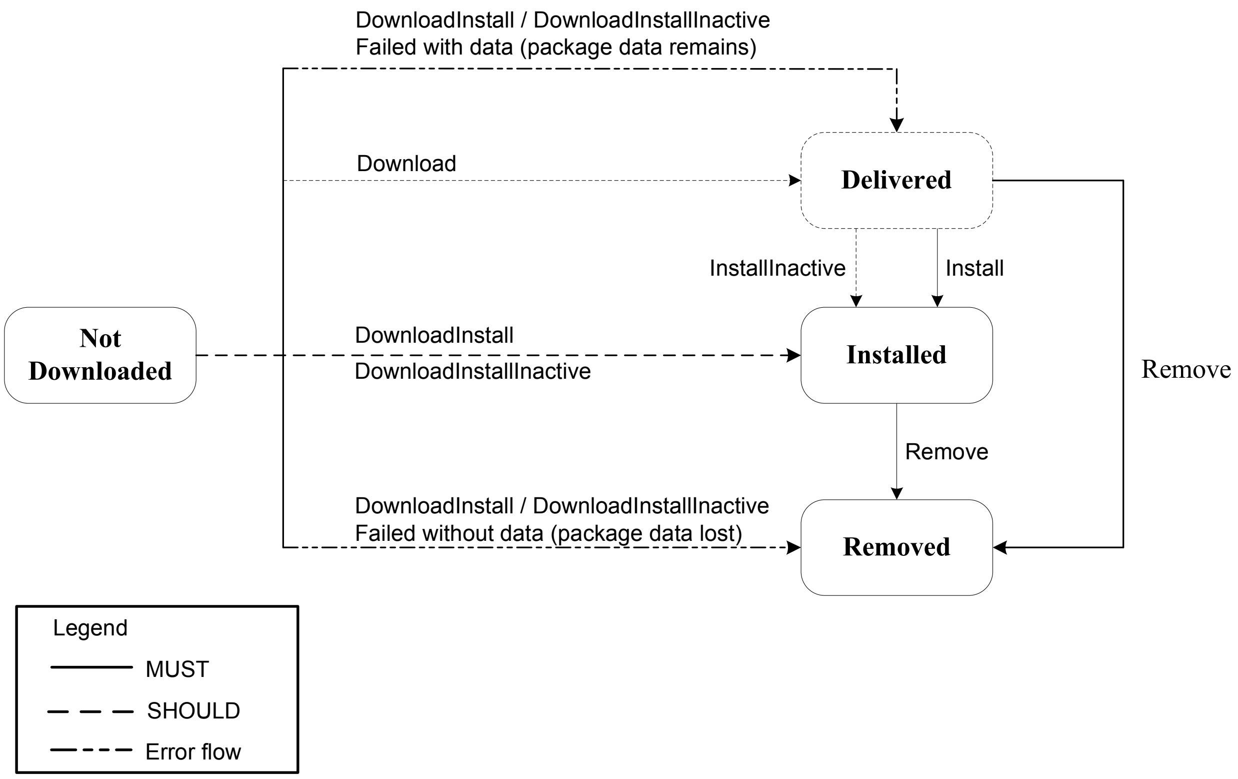
DownloadInstall / DownloadInstallInactive
Failed with data (package data remains)
Download
Delivered
InstallInactive
Install
Not Downloaded
DownloadInstall
DownloadInstallInactive
Installed
Remove
Remove
DownloadInstall / DownloadInstallInactive
Failed without data (package data lost)
Removed
Legend
MUST
SHOULD
Error flow

Figure 3: Delivery Package State Machine

7.2.1.1 Not Downloaded State

In Not Downloaded State, a Delivery Package has not been downloaded to the Device.

The SCOMO Client MUST support this state.

7.2.1.2 Delivered State

In Delivered State, a Delivery Package containing one or more Deployment Components has been delivered to the Device. The Deployment Component within the Delivery Package has not been installed and no corresponding nodes are added in the Deployed tree.

Delivered State also enables "deliver-first-install-later"-like use cases (for example, updates of mobile office solutions requiring all the Software Components to be activated immediately after back-end update). In this case the Delivery and Deployment are discrete.

The SCOMO Client SHOULD support this state.

7.2.1.3 Installed state

In Installed state, the Delivery Package has been installed, which means that Deployment Component(s) within it have been installed into the execution environment. The SCOMO Client MUST create corresponding nodes of each Deployment Component in the Deployed tree. The state of Deployment Components depends on the Delivery Package Primitive that has been executed. If the Primitive is 'Install' or 'DownloadInstall', then all Deployment Components have been installed and placed in Active state. If the Primitive is 'InstallInactive' or 'DownloadInstallInactive', then all Deployment Components have been installed and placed in Inactive state.

The SCOMO Client MUST support this state.

7.2.1.4 Removed State

In Removed state, the Delivery Package has been removed from the Device. The 'Remove' Primitive of Delivery Package MUST NOT affect the state of installed Deployment Components. The SCOMO Client MUST delete the Delivery Package from the Device and delete corresponding nodes that represented it from Delivered tree.

The SCOMO Client MUST support this state.

7.2.1.5 Delivery Package Primitives

Delivery Package Primitives are used to trigger management process and state transition of a Delivery Package.

NOTE: A Client that supports alternate download mechanism MUST support at least one of the operations: Download, DownloadInstall, or DownloadInstallInactive.

Primitive	Description	Applicable states	Post Primitive state	Primitive support dependency
Download	This Primitive is used to download a Delivery Package. The Delivery Package is placed in the Delivered state.	Not Downloaded	Delivered	MAY be supported See the Note above in this section
DownloadInstall	This Primitive is used to download and install a Delivery Package. The Deployment Component(s) contained in the Delivery Package will be installed to Active State.	Not Downloaded	Installed	MAY be supported See the Note above in this section
DownloadInstallInactive	This Primitive is used to download and install a Delivery Package. The Deployment Component(s) contained in the Delivery Package will be installed to Inactive State.	Not Downloaded	Installed	MAY be supported See the Note above in this section
Install	This Primitive is used to install a Delivery Package. The Deployment Component(s) contained in the Delivery Package will be installed to Active State.	Delivered	Installed	MUST be supported
InstallInactive	This Primitive is used to install a Delivery Package. The Deployment Component(s) contained in the Delivery Package will be installed to Inactive State.	Delivered	Installed	SHOULD be supported
Remove	This Primitive is used to remove Delivery Package from the Device.	Delivered Installed	Removed	MUST be supported

Table 2 – Delivery Package Primitives

Note that installation of a Delivery Package means installing the Deployment Components contained in the package; these MAY be either Deployment Components that were not previously installed on the Device (install from scratch) or updates to previously installed Deployment Components (install an update).

8. Standardized Management Objects

8.1 Introduction to Management Objects (Informative)

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

8.1.1 Definition and Description of Management Objects

OMA DM Management Objects are defined using the OMA DM Device Description Framework [DMTND], or DDF. 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.

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 (for example, 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, 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 (for example, *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 name is not known in the description and that it will be assigned at run-time. 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.

For a further introduction to this graphical notation, please refer to [DMStdObj].

8.2 DDF Compliance

The Management Object descriptions in this document are normative. However, the descriptions also contain a number of informative aspects that could be included to enhance readability or serve as examples. Other informative aspects are, for instance, the ZeroOrMore and OneOrMore elements, where implementations may introduce restrictions. All these exceptions are listed here:

- All XML comments (for example, "<!--some text '-->'"), are informative.
- The descriptions do not contain an RTProperties element, or any of its child elements, but a description of an actual implementation of this object MAY include these.

- If a default value for a leaf node is specified in a description by the DefaultValue element, an implementation MUST supply its own appropriate value for this element. If the DefaultValue element is present in the description of a node, it MUST be present in the implementation, but MAY have a different value.
- The value of all Man, Mod, Description, and DFTitle elements are informative and included only as examples.
- Below the interior nodes Ext and BearerParams, an implementation may add further nodes at will.
- The contents of the AccessType element MAY be extended by an implementation.
- If any of the following AccessType values are specified, they MUST NOT be removed in an implementation: Copy, Delete, Exec, Get, and Replace.
- If the AccessType value Add is specified, it MAY be removed in an implementation if the implementation only supports a fixed number of child nodes.
- An implementation MAY replace the ZeroOrMore or OneOrMore elements with ZeroOrN or OneOrN respectively. An appropriate value for *N* must also be given with the …*OrN* elements.

8.2.1 Conformance Definitions

The status definition in the node definitions indicates if the client supports that node or not. If the status is “Required”, then the client MUST support that node in the case the client supports the parent node.

9. Software Component Management Object

The Management Objects associated with Software Component management are assembled under an unnamed interior node *x*, dynamically or statically created.

Protocol Compatibility: This object is compatible with OMA Device Management protocol specifications, version 1.2 [DMPRO].

9.1 Tree Structure

SCOMO tree has a well-defined structure, with designated Ext nodes to allow non-standard extension nodes. The general structure of the tree is as follows:

1. Download: A sub-tree containing pre-delivery information and actions that are used for triggering the delivery and installation of Delivery Packages using indirect delivery mechanism, specified in section 7.2.
2. Inventory: A sub-tree containing post-delivery information and actions.

    2.1. Inventory/Delivered: A sub-tree containing post-delivery (but pre-installation) information of Delivery Packages. This sub-tree is created either by the Device after using indirect delivery mechanism or by the server before using direct delivery mechanism specified in section 7.2. This sub-tree contains actions for installation and removal.

    2.2. Inventory/Deployed: A sub-tree containing post-installation information about SCOMO Deployment Components and other installed components, as well as actions for activation, deactivation, and removal.

9.2 State Information in the Tree

SCOMO specifies two state machines: one for Delivery Package (see 7.2.1) and one for Deployment Component (see 7.1.1).

The state of a given Delivery Package can be known by examining the location of its <X> branch (within the Download or Delivery sub-tree) and querying the value of Delivered/<X>/State node.

The state of a given Deployment Component can be known by the presence (or absence) of its <X> branch and by querying the value of Deployed/<X>/State node (note that the Not Downloaded state and Removed states are logically different but the differentiation between them is mostly abstract).

9.3 Figure of the Management Object (Informative)

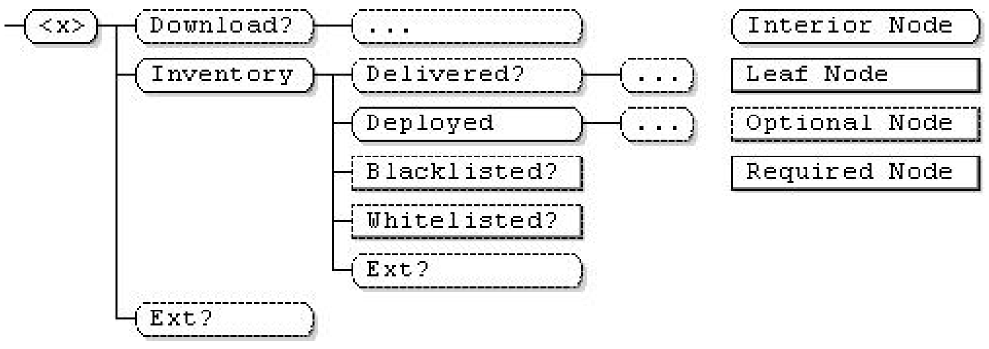

Figure 4: Software Component Management Object

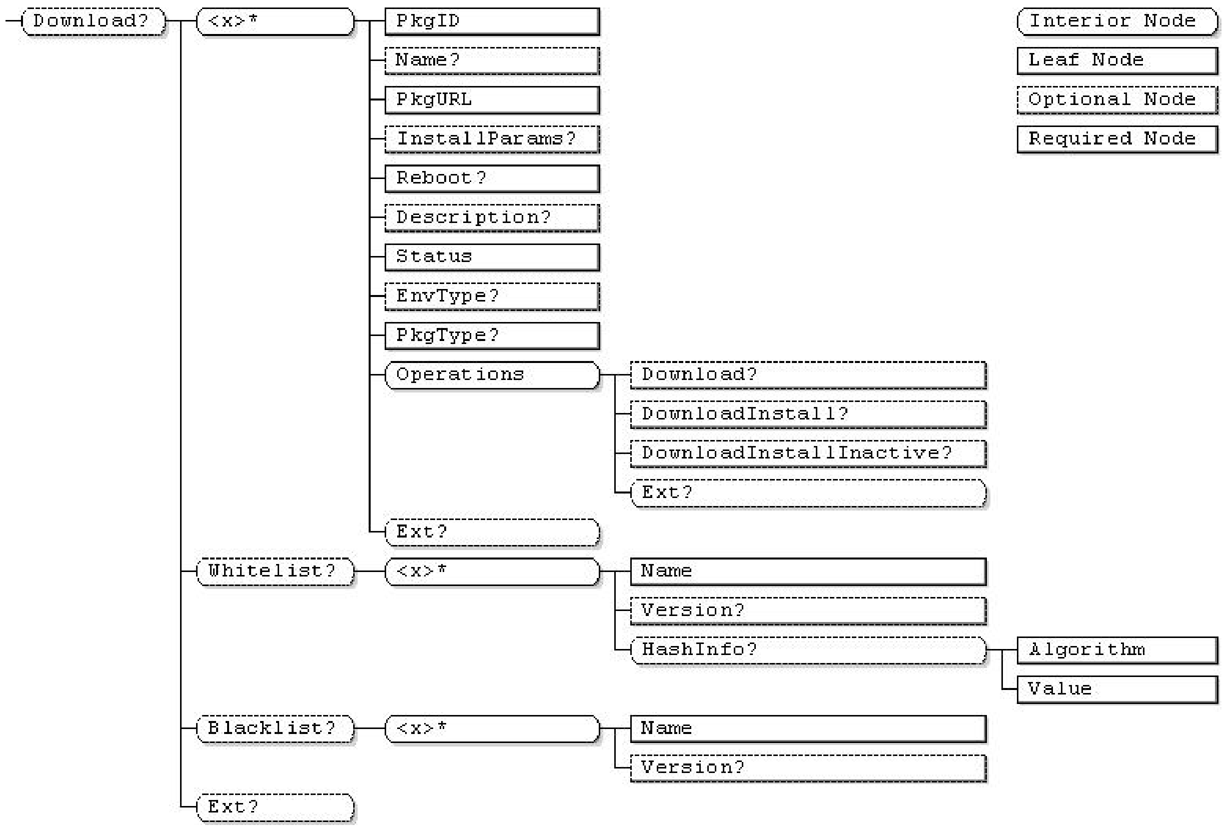

Figure 5: Software Component Management Object – Download Sub-tree

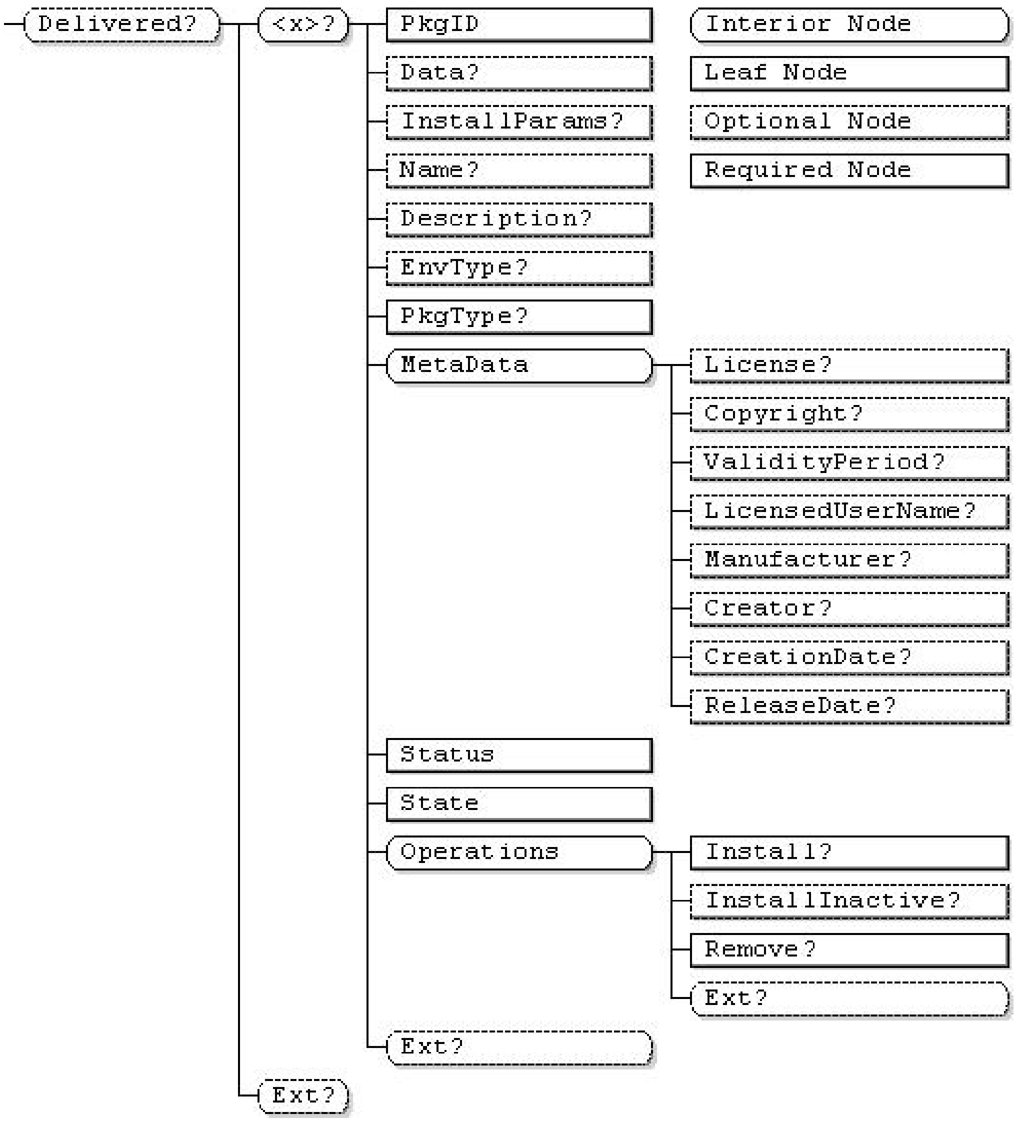

Figure 6: Software Component Management Object – Delivered Sub-tree

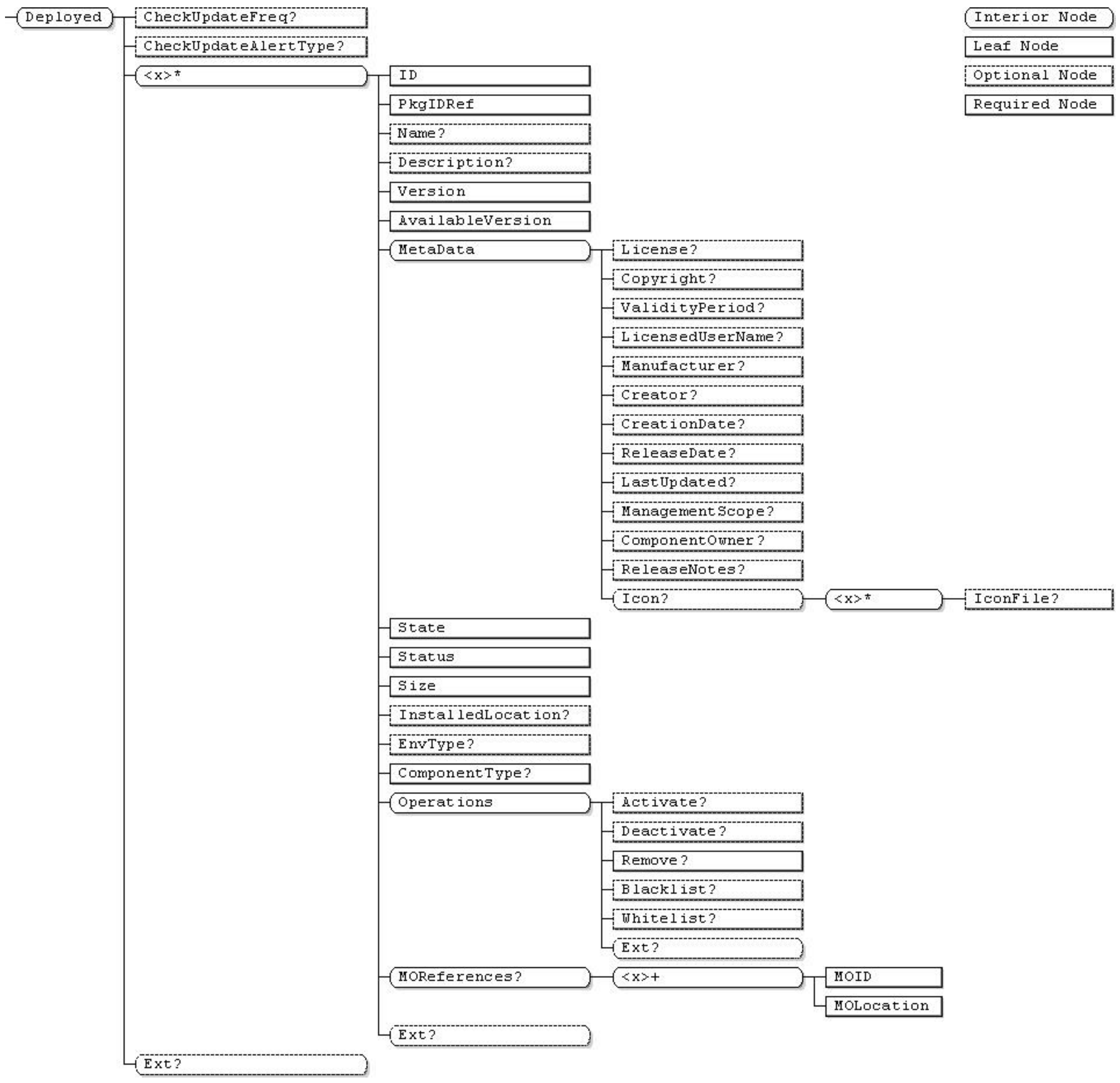

Figure 7: Software Component Management Object – Deployed Sub-tree

9.4 Software Component Management Object Parameters

Software Component Management Object consists of following parameters:

<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This interior node groups together the parameters of a Software Component Management Object. The ancestor elements of this node define the position in the Management Tree of this Management Object. But the structure of the DM tree and hence positions in the tree of Management Objects are out of scope of this specification.

The type of this node MUST be the SCOMO Management Object ID “urn:oma:mo:oma-scomo:1.1”.

Download

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This node is a parent node of the downloadable Delivery Packages that are ready for download or are in the process of downloading. This node MUST be used with the Alternate Download Mechanism described in section 11.1.2 (as opposed to usage of OMA DM delivery, as described in section 11.1.1).

Download/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrMore	node	Get

This node groups pre-delivery information and actions that are used for triggering the delivery and installation of Delivery Packages using the indirect delivery mechanism specified in section 7.1.2. Nodes in this sub-tree are used with Exec commands to start the Download operation to download the Delivery Package; to optionally install the Delivery Package; and to optionally move the Deployment Components into either the Active or the Inactive state.

Download/<x>/PkgID

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get, Replace

This leaf node specifies the identifier for a Delivery Package. The PkgID MUST be provided by the Device Management System and uniquely identify the Delivery Package within the SCOMO tree.

Download/<x>/Name

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get, Replace

This leaf node specifies the Name for the Delivery Package. The Package Name MUST be provided by the Device Management System.

Download/<x>/PkgURL

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the downloadable URL of the Delivery Package or its download descriptor. This URL is used for Alternative Download Mechanisms (such as HTTP Get [RFC2616] or Descriptor Based Download [DLOTA]).

Download/<x>/InstallParams

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node is used with DownloadInstall or DownloadInstallInactive Primitives to specify the installation parameters when installing the Delivery Package to the device. The format of the installation parameters is platform-specific.

Download/<x>/Reboot

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	int	Get

This leaf node is used with DownloadInstall or DownloadInstallInactive Primitives to specify when to reboot the device -- before and after installation. If this node is missing, then no reboot is needed.

Integer Value	Description
0	No reboot needed.
10	Reboot device before installation takes place.
20	Reboot device after installation has taken place.
30	Reboot device before installation and after installation.

Download/<x>/Description

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides a description of the Delivery Package.

Download/<x>/Status

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the Status of the Download and/or Install operation. This value is set by the SCOMO Client; the possible values are:

Integer Value	**Status**	**Description**
10	Idle / Start	There is no data available and download is about to start.
20	Download Failed	Download failed and there is No data received.
30	Download Progressing	Denotes that a download has started and that 0 or more bytes of data have been downloaded.
40	Download Complete	Have data after Download has been completed successfully.
50	Install Progressing	Denotes that an installation has started.
60	Install Failed with data	Install failed and the downloaded package is still in the Device.
70	Install Failed without data	Install failed and the downloaded package is deleted.

Download/<x>/EnvType

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get, Replace

This leaf node specifies the environment type of a Delivery Package. If a value is available for the EnvType, then it MUST be provided by the Device Management System. The value of this leaf node is an URI [RFC3986], which unambiguously identifies the target environment for the package. Standardized values for the URI value MUST be registered with OMNA.

As SCOMO applies to Software Components and unlike complete programs and applications, these components might not be functional without an environment that is installed in the device to allow Software Components to be used on the device (in general, this environment can consist of the OS, frameworks, libraries, or other software elements). In such a case, the EnvType can be used to address the environment that Software Component is applicable to.

In cases that the Software Component is targeted for a device that does not support the environment, the delivery package MUST be rejected -- meaning that it cannot be downloaded, installed, or activated in any form.

Download/<x>/PkgType

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	chr	Get, Replace

This leaf node specifies the Content Type of a Delivery Package. The value MUST be a MIME Content Type. The PkgType MUST be provided by the Device Management System.

Download/<x>/Operations

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This node is a parent node for Primitives that can be executed to start a download of a Delivery Package into the Device and possibly its subsequent installation.

Each of the direct child nodes of this parent are marked Optional, but a Client that supports Alternate Download Mechanism MUST support at least one of these child nodes.

Download/<x>/Operations/Download

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Download operation -- to download the Delivery Package into the Device and in the Delivered state.

Download/<x>/Operations/DownloadInstall

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the download operation to download the Delivery Package into the Device and to install the Deployment Component(s) (extracted from the Delivery Package) in the Active state.

Download/<x>/Operations/DownloadInstallInactive

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the download operation to download the Delivery Package into the Device and to install the Deployment Component(s) (extracted from the Delivery Package) in the Inactive state.

Download/<x>/Operations/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the download operations sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Download/Whitelist

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node is a parent node for the management of whitelisted Software Components.

Download/Whitelist/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrMore	node	Get

This node groups the information for one whitelisted Software Component. The device execution environment MAY use this information to determine whether a given downloaded package (from DM Server or by any other means) is allowed to be installed and used by the user.

Download/Whitelist/<x>/Name

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the name of the whitelisted Software Component . The name MUST be provided by the DM Server.

Download/Whitelist/<x>/Version

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the version information of the whitelisted Software Component. The version MAY be provided by the DM Server.

Download/Whitelist/<x>/HashInfo

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This interior node groups hash information.

Download/Whitelist/<x>/HashInfo/Algorithm

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the hashing algorithm used to calculate the hash value. The supported algorithm type information MUST be provided by the DM Server for the following values:

Value	**Algorithm**
1	MD5
2	SHA1
3	SHA256

Download/Whitelist/<x>/HashInfo/Value

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the hash value for the data of a Delivery Package. When an attempt is made to install and/or use a given Software Component, the Device execution environment MAY use the <x>/Download/Whitelist/<x>/HashInfo/Algorithm node and the value of this node to determine whether this package is acceptable or not.

Download/Blacklist

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node is a parent node for the management of blacklisted Software Components.

Download/Blacklist/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrMore	node	Get

This node groups the information for one blacklisted Software Component. The device execution environment may use this information to determine whether a given downloaded package (from DM Server or by any other means) is inhibited from installation and use standpoint by the user. When an attempt is made to install and/or use a given Software Component , the Device execution environment MAY use the <x>/Download/Blacklist/<x>/Name node and the <x>/Download/Blacklist/<x>/Version node to determine whether this package is inhibited or not.

Download/Blacklist/<x>/Name

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the name for the blacklisted Software Component . The name MUST be provided by the DM Server.

Download/Blacklist/<x>/Version

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the version information of the blacklisted Software Component. The version MAY be provided by the DM Server.

Download/<x>/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the Download Delivery Package sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Download/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the download sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees

Inventory

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This node is a parent node for all of the Delivery Packages and Deployment Components in the Device.

Inventory/Delivered

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This node is a parent node of the delivered Delivery Packages in the Device.

Inventory/Delivered/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	node	Get

This node acts as a placeholder for a Delivery Package in the Device.

The DM System provides the values for supported child nodes of this parent in either of two ways:

- When using OMA DM Replace as the delivery mechanism: These values are set by the SCOMO Server using DM commands.
- When using Alternate Download Mechanism: The SCOMO Client MUST copy these values from the corresponding branch of the Download sub-tree.

Inventory/Delivered/<x>/PkgID

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get, Replace

This leaf node specifies the identifier of a Delivery Package in the Device.

The PkgID is provided by the Device Management System and uniquely identifies the Delivery Package within the SCOMO tree. A Delivery Package MUST preserve the same identifier after a state transition. For example, the same value of Download/<X>/PkgID that was used for the Delivery Package while it was in the Download sub-tree MUST also be used as the value for Inventory/Delivered/<X>/PkgID in the Delivered sub-tree.

Inventory/Delivered/<x>/Data

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	bin	No Get, Replace

This leaf node specifies the actual binary data of a Delivery Package. When using OMA DM for delivery, this node is targeted with a Replace command containing the data of the Delivery Package to be delivered.

Inventory/Delivered/<x>/InstallParams

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node is used with Install or InstallInactive Primitives to specify the installation parameters when installing the Delivery Package to the device. The format of the installation parameters is platform-specific.

Inventory/Delivered/<x>/Name

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get, Replace

This leaf node specifies the Name of a Delivery Package in the Device. The Package Name is provided by the Device Management System. A Delivery Package MUST preserve the same name after a state transition. For example, the same value of Download/<X>/Name that was used for the Delivery Package while it was in the Download sub-tree MUST also be used as the value for Inventory/Delivered/<X>/Name node.

Inventory/Delivered/<x>/Description

Status	Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides a description of the Delivery Package. A Delivery Package MUST preserve the same description after a state transition. For example, the same value of Download/<X>/Description that was used for the Delivery Package while it was in the Download sub-tree MUST also be used as the value for Inventory/Delivered/<X>/Description node.

Inventory/Delivered/<x>/EnvType

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get, Replace

This leaf node specifies the environment type of a Delivery Package in the Device. The value of EnvType is provided by the Device Management System. The value is an URI [RFC3986] that unambiguously identifies the target environment for the package. Standardized values for the URI value MUST be registered with OMNA.

A Delivery Package MUST preserve the same EnvType after a state transition. For example, the same value of Download/<X>/EnvType that was used for the Delivery Package while it was in the Download sub-tree MUST also be used as the value for Inventory/Delivered/<X>/EnvType in the Delivered sub-tree.

As SCOMO applies to Software Components and unlike complete programs and applications, these components might not be functional without an environment that is installed in the device to allow Software Components to be used on the device (in general, this environment can consist of the OS, frameworks, libraries, or other software elements). In such a case, the EnvType can be used to address the environment that Software Component is applicable to.

In cases that the Software Component is targeted for a device that does not support the environment, the delivery package MUST be rejected -- meaning that it cannot be installed or activated in any form.

Inventory/Delivered/<x>/PkgType

Status	Occurrence	Format	Min. Access Types
Required	ZeroOrOne	chr	Get, Replace

This leaf node specifies the Content Type of a Delivery Package in the Device. The value MUST be a MIME content type. The PkgType MUST be provided by the Device Management System. A Delivery Package MUST preserve the same PkgType after a state transition. For example, the same value of Download/<X>/PkgType that was used for the Delivery Package while it was in the Download sub-tree MUST also be used as the value for Inventory/Delivered/<X>/PkgType in the Delivered sub-tree.

Inventory/Delivered/<x>/MetaData

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This interior node specifies Metadata for the Delivered Package, providing information such as license, manufacturer, copyright, etc.

Inventory/Delivered/<x>/MetaData/License

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node is used to provide license information for the Delivery Package. If used, it MUST provide the license information or a link to it. The license information describes the terms and conditions for using the Package. The specific text formatting and value of this node is left to implementation.

Inventory/Delivered/<x>/MetaData/Copyright

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides copyright information for the Delivered Package. If used, it MUST provide the copyright information or a link to it.

Inventory/Delivered/<x>/MetaData/ValidityPeriod

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the validity period of the license -- the date up to which the license is valid for the Delivered Component. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. It is up to the implementation to use this information (for example, to notify the user of expiry and the need for update to renew or purchase a new license). If the validity period is not available, this value MUST be an empty string.

Inventory/Delivered/<x>/MetaData/LicensedUserName

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies authorized licensee(s) of the Delivered Package. The specific text formatting and value of this node is left to implementation.

Inventory/Delivered/<x>/MetaData/Manufacturer

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides a name of the manufacturer of the Delivered Package. The specific text formatting and value of this node is left to implementation.

Inventory/Delivered/<x>/MetaData/Creator

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a creator of the Delivered Package. In many cases, the creator can be different from the manufacturer (which actually releases and distributes the package).

Inventory/Delivered/<x>/MetaData/CreationDate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a creation date of the Delivered Package. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. If the creation date is not available, this value MUST be an empty string.

Inventory/Delivered/<x>/MetaData/ReleaseDate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a release date of the Delivered Package. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. If the release date is not available, this value MUST be an empty string.

Inventory/Delivered/<x>/Status

Status	Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the status of Remove or Install Primitive. This value is set by the SCOMO Client and the possible values are:

Integer Value	**Status**	**Description**
10	Idle / Start	The Device has not started the Remove or Install Primitive.
20	Remove Failed	Remove failed and the Deployment Component is still in the Device
30	Remove Progressing	Denotes that Remove has started.
40	Install Progressing	Denotes that an installation has started.
50	Install Failed with data	Install failed and the downloaded package is still in the Device.
60	Install Failed without data	Install failed and the downloaded package is deleted.

Inventory/Delivered/<x>/State

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the state of a Delivery Package in the Device. The value of this node is one of the following:

Integer Value	State	Description
10	Delivered	The Delivery Package is in the Delivered State.
20	Installed	The Delivery Package is in the Installed State.

Inventory/Delivered/<x>/Operations

Status	Occurrence	Format	Min. Access Types
Required	One	node	Get

This node is a parent node for Primitives that can be executed for Delivery Package in the Device.

Inventory/Delivered/<x>/Operations/Install

Status	Occurrence	Format	Min. Access Types
Required	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Install operation. Deployment Component(s) from a single Delivery Package are installed in the Active state.

Inventory/Delivered/<x>/Operations/InstallInactive

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the InstallInactive operation. Deployment Component(s) from a single Delivery Package are installed in the Inactive state.

Inventory/Delivered/<x>/Operations/Remove

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Remove operation to remove the Delivery Package from the Device.

Inventory/Delivered/<x>/Operations/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the Delivered operations sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Delivered/<x>/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the Delivery Package parameters sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Delivered/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the delivered sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Deployed

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This node is a parent node of the Deployment Components in the Device. The Client MUST include the information of Deployment Components delivered and deployed via SCOMO in the Inventory/Deployed sub-tree. The Client SHALL also include information of Deployment Components installed outside of SCOMO in the Inventory/Deployed sub-tree (for example, at the factory or by the end user).

Inventory/CheckUpdateFreq

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	int	Get

This leaf node specifies the frequency (numbered by days) of client-initiated check for new version of the Deployed Components.

Inventory/Deployed/CheckUpdateAlertType

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	int	Get

This leaf node specifies the Alert Type of client-initiated check for new version of the Deployed Components. The definition of the Alert Type for the client-initiated update alert is specified in the section 13. The value of this node MUST be one of the following values:

Integer Value	**Alert Type**
10	UpdateDeviceRequest
20	CheckVersionDeviceRequest

Inventory/Deployed/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrMore	node	Get

This node acts as a placeholder for a Deployment Component in the Device. The node name is assigned by the SCOMO Client.

Inventory/Deployed/<x>/ID

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get, Replace

This leaf node specifies the identifier of a deployed Deployment Component in the Device. The ID MUST be provided by the Device Management System and uniquely identify the Deployment Component within the SCOMO tree. The implementation MAY generate execution environment-specific ID for the Deployment Component during installation in the Device and maintain the mapping with Component ID provided by the Device Management System.

Inventory/Deployed/<x>/PkgIDRef

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node provides the reference to the Delivery Package ID as specified in Download and/or Delivered tree in the Device. The value of this node is the same as the value of PkgID for corresponding Delivery Package if the Software Component is deployed via SCOMO mechanism. If the Software Component is deployed using other mechanisms (such as the user installs an application), the value of PkgIDRef MUST be '-1'.

Inventory/Deployed/<x>/Name

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the Name of a Deployment Component in the Device.

Inventory/Deployed/<x>/Description

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides a description of the Deployment Component in the Device.

Inventory/Deployed/<x>/Version

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the Version of a Deployment Component in the Device.

The format of the version string is implementation-specific and MAY depend on the characteristics of the Deployment Component; however, this opaque string SHOULD encapsulate information that can differentiate between different versions (for example, different editions) of the same Deployment Component.

A common and widely accepted version string contains a sequence of two or more positive integers, separated by '.' Character, for example: "2.1.1".

A Deployment Component that does not have any version information at the time of installation MAY use an empty string to denote that it is version-less.

Inventory/Deployed/<x>/AvailableVersion

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This leaf node specifies the available version of the Deployment Component in the Server when device- or user-initiated check for new version of the Deployment component is completed.

The format of the version string is implementation-specific and MAY depend on the characteristics of the Deployment Component; however, this opaque string SHOULD encapsulate information that can differentiate between different versions (for example, different editions) of the same Deployment Component.

A common and widely accepted version string contains a sequence of two or more positive integers, separated by '.' Character, for example: "2.1.1".

When a Software Component is installed for the first time or updated later, the value of the corresponding AvailableVersion node SHOULD be the same as the value of the Version node.

Inventory/Deployed/<x>/MetaData

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This interior node specifies the metadata on the Deployed Component, providing information such as date and time of creation, manufacturer, creator of the software, copyright, etc.

Inventory/Deployed/<x>/MetaData/License

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides license information of the Deployed Component license. If used, it MUST provide the license information or a link to it. The license information describes the terms and conditions for using the Software Component. The specific text formatting and value of this node is left to implementation.

Inventory/Deployed/<x>/MetaData/Copyright

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides the copyright information on the Deployed Component. If used, it MUST provide the copyright information or a link to it.

Inventory/Deployed/<x>/MetaData/ValidityPeriod

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the validity period of the license -- the date up to which the license is valid for the Deployed Component. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. It is up to the implementation to use this information (for example, to notify the user of expiry and the need for update to renew or purchase a new license). If the validity period is not available, this value MUST be an empty string.

Inventory/Deployed/<x>/MetaData/LicensedUserName

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the authorized licensee(s) of the Deployed Component. The specific text formatting and value of this node is left to implementation.

Inventory/Deployed/<x>/MetaData/Manufacturer

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node provides a name of the manufacturer of the Deployed Component. The specific text formatting and value of this node is left to implementation.

Inventory/Deployed/<x>/MetaData/Creator

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a creator of the Deployed Component. In many cases, the creator can be different from the manufacturer (which actually releases and distributes the component).

Inventory/Deployed/<x>/MetaData/CreationDate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a creation date of the Deployed Component. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. If the creation date is not available, this value MUST be an empty string.

Inventory/Deployed/<x>/MetaData/ReleaseDate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies a release date of the Deployed Component. The numeric representation of the date should follow ISO 8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. If the release date is not available, this value MUST be an empty string.

Inventory/Deployed/<x>/MetaData/LastUpdated

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the date and time when the Deployed Component was last updated. The numeric representation of the date and time should follow IS8601 basic format, complete representation of a date and time value, as specified in the [DMTND]. If the last updated date and time is not available, this value MUST be an empty string.

Inventory/Deployed/<x>/MetaData/ManagementScope

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	int	Get

This leaf node specifies the management scope for this Software Component, whether the Software Component is managed through SCOMO or it is an inventory-only Software Component. The value of this node is informative. The device implementation decides how this value is updated. Possible values are:

Integer Value	**Description**
0	Software Component installed using SCOMO (which implies managed via SCOMO).
10	Software Component installed using a non-SCOMO mechanism. The Software Component is inventory only.
20	Software Component installed using a non-SCOMO mechanism, but the Software Component MAY be managed through SCOMO (for example, factory installed Software Components that may be later managed through SCOMO).

Inventory/Deployed/<x>/MetaData/ComponentOwner

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the owner of the Software Component. Possible values are: "Operator", "Device Manufacturer", "Third Party", and "User".

Inventory/Deployed/<x>/MetaData/ReleaseNotes

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the release notes for this Software Component.

Inventory/Deployed/<x>/MetaData/Icon

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This interior node is a parent node of the Icons specified for this Software Component.

Inventory/Deployed/<x>/MetaData/Icon/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrMore	node	Get

This interior node acts as a place holder for an icon used with the graphical user interface (GUI) of this Software Component.

Inventory/Deployed/<x>/MetaData/Icon/<x>/IconFile

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	bin	Get, Replace

This leaf node specifies a single icon used as the Graphical User Interface (GUI) element for this Software Component. The value of this node is the Icon file in binary format. The Device Management System MAY create and update icons used with a Software Component. Actual format, size, and usage of icons are execution environment-dependent and it is left to the implementation.

Inventory/Deployed/<x>/State

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the State of a Deployment Component in the Device. The value of this node is one of the following:

Integer Value	**State**	**Description**
10	Inactive	The Deployment Component is in the Inactive State.
20	Active	The Deployment Component is in the Active State.

Inventory/Deployed/<x>/Status

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the Status of Remove, Activate, and Deactivate Primitives. This value is set by the SCOMO Client and the possible values are:

Integer Value	**Status**	**Description**
10	Idle / Start	The Device has not started the Remove operation.
20	Remove Failed	Remove failed and the component is still in the Device.
30	Remove Progressing	Denotes that Remove has started.
40	Activate Failed	Activate failed and the Software Component is still in the Inactive state.
50	Activate Progressing	Denotes that the Activate operation has started.
60	Deactivate Failed	Deactivate failed and the Software Component is still in the Active state.
70	Deactivate Progressing	Denotes that the Deactivate operation has started.

Inventory/Deployed/<x>/Size

Status	Tree Occurrence	Format	Min. Access Types
Required	One	int	Get

This leaf node specifies the actual installed size of the Software Components installed.

Inventory/Deployed/<x>/InstalledLocation

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the actual installed location of the Software Component in the device. This node MUST exist only if the location of Software Component in the device is unique.

Inventory/Deployed/<x>/EnvType

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This leaf node specifies the environment type of a Deployment Component in the Device. The value of this leaf node is an URI [RFC3986], which unambiguously identifies the target environment that was used for installation of the Deployment Component. Standardized values for the URI value MUST be registered with OMNA.

Inventory/Deployed/<x>/ComponentType

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	chr	Get, Replace

This leaf node specifies the Content Type of a deployed Software Component in the Device. The value MUST be a MIME content type. The Type MUST be provided by the Device Management System. A Software Component MUST preserve the same Type after a state transition.

Inventory/Deployed/<x>/Operations

Status	Tree Occurrence	Format	Min. Access Types
Required	One	node	Get

This node is a parent node for Primitives that can be executed for a Deployment Component in the Device.

Inventory/Deployed/<x>/Operations/Activate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Activate operation to transfer the Deployment Component from the Inactive state to the Active state.

Inventory/Deployed/<x>/Operations/Deactivate

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Deactivate operation to transfer the Deployment Component from the Active state to the Inactive state.

Inventory/Deployed/<x>/Operations/Remove

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Remove operation to Remove the Deployment Component from the Device. The presence of this node means the Deployment Component supports the Remove operation.

Inventory/Deployed/<x>/Operations/Blacklist

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Blacklist operation. Successful execution of the Exec command on this node results in updating the ‘Deployed/Blacklisted’ and ‘Deployed/Whitelisted’ nodes as specified. The presence of this node means the Deployment Component supports the Blacklist operation. This operation SHOULD be applicable irrespective of the Management Scope in the Metadata.

Inventory/Deployed/<x>/Operations/Whitelist

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	null	Get, Exec

This node is used with the Exec command to start the Whitelist operation. Successful execution of the Exec command on this node results in updating the ‘Deployed/Blacklisted’ and ‘Deployed/Whitelisted’ nodes as specified. The presence of this node means the Deployment Component supports the Whitelist operation. This operation SHOULD be applicable irrespective of the Management Scope in the Metadata.

Inventory/Deployed/<x>/Operations/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the Deployment Component operations sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Deployed/<x>/MOReferences

Status	Tree Occurrence	Format	Min. Access Types
Required	ZeroOrOne	node	Get

This node is a parent node to the MO references for this deployed Software Component. MOs associated with this deployed Software Component MUST be specified under this node.

Inventory/Deployed/<x>/MOReferences/<x>

Status	Tree Occurrence	Format	Min. Access Types
Required	OneOrMore	node	Get

This node is a placeholder for one instance of a MO reference.

Inventory/Deployed/<x>/MOReferences/<x>/MOID

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

This value of leaf node MUST contain the Management Object ID of the associated Management Object for this instance.

Inventory/Deployed/<x>/MOReferences/<x>/MOLocation

Status	Tree Occurrence	Format	Min. Access Types
Required	One	chr	Get

The value of this leaf node specifies the location of the root node of the corresponding MO in the form of Uniform Resource Identifier (URI).

Inventory/Deployed/<x>/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the Deployment Component parameters sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Deployed/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the deployed sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Inventory/Blacklisted

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This node provides a list of Software Components that are blacklisted in the device. The value of this node is a simple list of component IDs separated by ';'. For example, 'id1;id2;id3'. This node is updated to include a blacklisted Software Component when the Blacklist operation is successful. When a previously blacklisted Software Component is whitelisted, the Software Component is removed from the list. When a previously whitelisted Software Component is blacklisted, the component ID is added to the list. When a previously blacklisted Software Component is removed from the device, the Software Component MAY be removed from the list. The environment MAY use this information to prevent a user from accessing the Software Component.

Inventory/Whitelisted

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	chr	Get

This node provides a list of Software Components that are whitelisted in the device. The value of this node is a simple list of component IDs separated by ';'. For example, 'id1;id2;id3'. This node is updated to include a whitelisted Software Component when the Whitelist operation is successful. When a previously whitelisted Software Component is blacklisted, the Software Component is removed from the list. When a previously blacklisted Software Component is whitelisted, the Software Component is added to the list. When a previously whitelisted Software Component is removed from the device, the Software Component MAY be removed from the list. The environment MAY use this information to allow a user to access the blacklisted Software Component.

Inventory/Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the inventory sub-tree into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

Ext

Status	Tree Occurrence	Format	Min. Access Types
Optional	ZeroOrOne	node	Get

This optional interior node designates a branch of the SCOMO root into which platform or vendor extensions MAY be added, permanently or dynamically. Ext sub-trees, such as this one, are included at various places in the Software Component Management Object to provide flexible points of extension for platform or implementation-specific parameters. However, vendor extensions MUST NOT be defined outside of one of these Ext sub-trees.

10.Behavior Associated with the Management Object

In the SCOMO tree, the Exec command is only allowed to target Primitives under Operations nodes. After an Exec command to one of the Primitives under Operations node, the SCOMO Client MUST send a response to the SCOMO Server in either of the following ways:

1. Asynchronously: If the Exec command is acceptable and executed asynchronously.

   - The SCOMO Client MUST return status code 202 ("Accepted for processing") for the Exec command as defined in [DMREPRO].

   - Upon completion of the asynchronous operation, the SCOMO Client MUST send an alert back to the SCOMO Server by using the Generic Alert defined in [DMPRO].

2. Synchronously: If the Exec command is acceptable and executed synchronously.

   - If the Exec command is performed successfully, the SCOMO Client MUST return SCOMO Result Code 1200 ("Command completed successfully") for the Exec command.

In SCOMO Client's response message for reporting purposes, the SCOMO Client MUST include the Delivery Package or Deployment Component information in addition to the status code described above:

- URI(s) of newly created dynamic node(s) <x> for Delivery Package or Deployment Component as a result of the Primitives executed in <Item>/<Target> element.

- URI of the Primitive node on which the Exec command was invoked in <Item>/<Source> element.

- For asynchronous reporting, Delivery Package PkgID or Deployment Component ID in <Item>/<Data> element which format is 'xml'.

- For asynchronous reporting, SCOMO Result Code of the Primitive executed in <Item>/<Data> element which format is 'xml'.

After an Exec command to one of the Primitives under Operations node it is the SCOMO Client's responsibility to reflect the changes in the Inventory sub-tree as follows:

- After successful installation of Deployment Components, the SCOMO Client MUST create their nodes in Deployed sub-tree.

- The SCOMO Client MUST create nodes for Delivery Package in the Delivered sub-tree, in case a Download Primitive was executed successfully.

- The SCOMO Client MUST remove nodes for Delivery Package in the Delivered sub-tree, in case a Remove Primitive was executed successfully.

- The SCOMO Client MUST remove nodes for Deployment Component in the Deployed sub-tree, in case a Remove Primitive was executed successfully.

- The SCOMO Client MAY create nodes for Delivery Package in the Delivered sub-tree, in case a DownloadInstall or DownloadInstallInactive Primitive was executed successfully. In this case, it MAY also remove these nodes after Deployment Components were installed successfully.

The detailed Exec command semantics and SCOMO Client responsibility are described below.

10.1 Structure Elements of <Item>/<Data> for Reporting

As described above, the Delivery Package PkgID or Deployment Component ID will be included in XML structure in <Item>/<Data> element. The XML structure is defined below.

10.1.1 ResultCode

Usage: Specify the SCOMO Result Code of the Primitive executed

Parent Elements: none

Restrictions: The Format of this element MUST be int.

Content Model: (#PCDATA)

10.1.2 Identifier

Usage: Specify the Delivery Package ID or Deployment Component ID

Parent Elements: none

Restrictions: The Format of this element MUST be chr.

Content Model: (#PCDATA)

10.2 Exec Command Semantics on Download Sub-tree

Before performing the Exec command, the following pre-conditions need to be satisfied:

- The dynamic node Download/<X> MUST be created.
- Optionally the value of the Download/<X>/Reboot node is set.
- The following nodes MUST be set with appropriate values:
    - Download/<X>/PkgID
    - Download/<X>/PkgURL
- At least one Operation sub-node MUST exist under Download/<X>/Operations.

In case the Download/<X>/EnvType is specified and is incompatible with the environment on the device, the SCOMO Client MUST reject the Exec command with a SCOMO Result Code 1413 (Operation rejected – unsupported environment).

The Exec command targeting a child node of an Operations node starts the execution of a chosen Primitive. For example:

```
<Exec>
   <CmdID>3</CmdID>
   <Item>
      <Target>
         <LocURI>./SCM/Download/Package123/Operations/DownloadInstall</LocURI>
      </Target>
   </Item>
</Exec>
```

If the Exec command is targeting any Primitives under the Download sub-tree, the SCOMO Client MUST start the download of the Delivery Package. If the Exec command is targeting the DownloadInstall or the DownloadInstallInactive Primitive, after successful download the SCOMO Client MUST start the installation of the Deployment Components.

The SCOMO Client responsibilities are described in the following table. Gray cells describe the case and white cells show the resulting responsibilities:

Primitive	Download	DownloadInstall or DownloadInstallInactive	DownloadInstall or DownloadInstallInactive
download procedure	Succeeds	Succeeds	Succeeds
installation procedure	N/A	Fails	Succeeds
Create nodes in Delivered sub-tree and send Delivery Package information to the SCOMO Server	MUST	MAY	MAY
Create nodes in the Deployed sub-tree and send Deployment Component information to the SCOMO Server	N/A (no installation)	N/A (installation failed)	MUST
Remove nodes from the Download sub-tree	SHOULD	MAY	SHOULD

Table 3 – Responsibilities Associated with Exec on Download Sub-tree

After the nodes are created in the Delivered or the Deployed sub-tree as described above, the SCOMO Client MUST assign ACL permissions to the new <x> node and its sub-nodes to ensure the originally authorized DM Servers have the same permission to the Delivery Package or Deployment Component as before download or installation.

In addition to applying ACL inheritance rules as defined in [DMTND], the SCOMO Client copies permissions into the ACL property value of new dynamic node <x> and its sub-nodes as described below:

- The ACL permissions of parent node <x> of the Primitive node on which the Exec command was invoked are copied to the new <x> node in Delivered or Deployed sub-tree. These copied permissions are added to those inherited as per [DMTND].
- The ACL permissions of Primitive node on which the Exec command was invoked are copied to new Primitive nodes under new <x>/Operations node in Delivered or Deployed sub-tree. These copied permissions are added to those inherited as per [DMTND].
- The ACL permissions of other sub-nodes under new <x> node inherit the ACL permissions from its parent node according to ACL inheritance rules as defined in [DMTND].

When trying to process an installation request – if the Deployment Component ID already exists, the SCOMO Client MUST either replace the value of existing nodes in the Deployed sub-tree to avoid ID clashes within SCOMO tree (which implies an update of an existing Deployment Component) or respond with an appropriate SCOMO Result code to indicate an error (which implies that an update is not possible).

10.3 Exec Command Semantics on Delivered

Before performing the Exec command, the following pre-conditions need to be satisfied:

- The dynamic node Delivered/<X> MUST be created.
- The following nodes MUST be set with appropriate values:

  - Delivered/<X>/PkgID
  - Delivered/<X>/Data

- At least one Operation sub-node MUST exist under Delivered/<X>/Operations.

In case that Delivered/EnvType is specified and is incompatible with the environment on the device, the SCOMO Client MUST reject the Exec command with a SCOMO Result Code 1407 (failed package validation).

The Exec command targeting a child node of an Operations node starts the execution of a chosen Primitive. For example:

```
<Exec>
   <CmdID>3</CmdID>
   <Item>
      <Target>

   <LocURI>./SCM/Inventory/Delivered/Package456/Operations/Install</LocURI>
      </Target>
   </Item>
</Exec>
```

If the Exec command is targeting Install Primitives, the SCOMO Client MUST start the installation of the Deployment Components.

The SCOMO Client responsibilities in that case are described in the following table. Gray cells describe the case and white cells show the resulting responsibilities:

Installation procedure	succeeds	failed with data (Delivery Package data remains in the Device)	failed without data (Delivery Package data lost)
Create nodes in the Deployed sub-tree and send Deployment Component information to the SCOMO Server	MUST	MUST NOT	MUST NOT
Remove nodes from the Delivered sub-tree	MAY	MUST NOT	MUST

Table 4 – Responsibilities Associated with Exec on Delivered Sub-tree

If the Exec command is targeting Remove Primitives under the Delivered sub-tree, the SCOMO Client MUST remove the Delivery Package from the Device. In this case the SCOMO Client MUST remove nodes of the Delivery Package from the Delivered sub-tree.

After the nodes are created in the Deployed sub-tree as described above, the SCOMO Client MUST assign ACL permissions to new <x> node and its sub-nodes to ensure the originally authorized DM Servers have the same permission to the Delivery Package or Deployment Component as before installation.

In addition to applying ACL inheritance rules as defined in [DMTND], the SCOMO Client copies permissions into the ACL property value of the new dynamic node <x> and its sub-nodes as described below:

- The ACL permissions of the parent node <x> of the Primitive node on which the Exec command was invoked are copied to the new <x> node in the Deployed sub-tree.

- The ACL permissions of the Primitive node on which the Exec command was invoked are copied to the new Primitive sub nodes under new <x>/Operations node in the Deployed sub-tree.

- The ACL permissions of other sub nodes of the new <x> node inherit the ACL property value from its parent node according to ACL inheritance rules as defined in [DMTND].

When trying to process an installation request, if the Deployment Component ID already exists, the SCOMO Client MUST either replace the value of existing nodes in the Deployed sub-tree to avoid ID clashes within SCOMO tree (which implies an update of an existing Deployment Component) or respond with an appropriate SCOMO Result code to indicate an error (which implies that an update is not possible).

10.4 Exec Command Semantics on Deployed

Before performing the Exec command, the following pre-conditions need to be satisfied:

- The dynamic node Deployed/<X> MUST be created.
- The following node MUST be set with an appropriate value:
  - Deployed/<X>/PkgIDRef
  - At least one Operation sub-node MUST exist under Deployed/<X>/Operations

The Exec command targeting a child node of an Operations node starts the execution of a chosen Primitive. For example:

```
<Exec>
   <CmdID>3</CmdID>
   <Item>
      <Target>

   <LocURI>./SCM/Inventory/Deployed/Component789/Operations/Activate</LocURI>
      </Target>

   </Item>
</Exec>
```

If the Exec command is targeting Remove Primitive under the Deployed sub-tree, the SCOMO Client MUST remove the Deployment Component from the Device and remove all the nodes of the Deployment Component from the Deployed sub-tree.

10.5 Use of Synchronous Reporting

If the Exec command is executed synchronously and succeeds, the SCOMO Client MUST send Delivery Package or Deployment Component information in a <Status> command for Exec command as described below:

- The <Data> element MUST contain a valid SCOMO Result Code (Sec 10.7).
- The URI of the Primitive node on which the Exec command was invoked – used to identify the source – MUST be included in the <Source>/<LocURI> of <Status>/<Item> element.
- The URI(s) of the newly created dynamic node(s) <x> under the Delivered or Deployed tree for the Delivery Package or Deployment Component used to identify the result – MUST be included in the <Target>/<LocURI> of <Status>/<Item> element.

In case nodes are created for multiple Deployment Components after package installation, they MUST be individually represented in multiple Item elements in a <Status>. In each Item element, the Target/LocURI element contains the URI of the newly created dynamic node(s) <x> under the Deployed sub-tree, and the Source/LocURI contains the URI of the executable Primitive node on which the Exec command was invoked.

Following is the example message:

```
<Status>
```

```
    <MsgRef>1</MsgRef>
    <CmdRef>2</CmdRef>
    <Cmd>Exec</Cmd>
    <Data>1200</Data>       <!-- SCOMO Result Code -->
    <Item>
      <Source>
        <LocURI>./SCOMO/Download/Package1/Operations/DownloadInstall</LocURI>
      </Source>
      <Target>
        <LocURI>./SCOMO/Inventory/Deployed/Component1</LocURI>
      </Target>
    </Item>
    <Item>
      <Source>
        <LocURI>./SCOMO/Download/Package1/Operations/DownloadInstall</LocURI>
      </Source>
      <Target>
        <LocURI>./SCOMO/Inventory/Deployed/Component2</LocURI>
      </Target>
    </Item>
  </Status>
```

10.6 Use of the Generic Alert

If the Exec command is executed asynchronously, the SCOMO Client MUST send a notification to the SCOMO Server about the outcome of the operation via a Generic Alert [DMPRO] message. The SCOMO Client MAY send the alert message during the same or subsequent DM session. The alert message includes the following data:

- Delivery Package PkgID or Deployment Component ID in <Item>/<Data> element; format is 'xml' (see section 10.1).
- An integer Result Code, which is used to report status of the operation. The appropriate SCOMO Result Code defined in section 10.7 MUST be included in an Item/Data element which format is 'xml' (see section 10.1).
- The URI of the Primitive node on which the Exec command was invoked – used to identify the source – MUST be included in the Source/LocURI element within the Generic Alert.
- The URI(s) of the newly created dynamic node(s) <x> under the Delivered or Deployed tree for the Delivery Package or Deployment Component used to identify the result – MUST be included in the Target/LocURI within the Generic Alert.
- An alert type – used to identify the operation MUST be included.

Alerts that are reporting an error or failure condition SHOULD report an importance level defined in [DMPRO] higher than Informational in the Mark field of the Meta information. If the SCOMO Server needs to retrieve additional information, such as download or install status described in Download/<X>/Status node, then the SCOMO Server MAY query the Device for those specific nodes.

In case nodes are created for multiple Deployment Components after package installation, they MUST be individually represented in multiple Item elements in a single Generic Alert message. In each Item element, the Target/LocURI element contains the URI of the newly created dynamic node(s) <x> under the Deployed sub-tree, and the Source/LocURI contains the URI of the executable Primitive node on which the Exec command was invoked.

Following is the example message:

```
<Alert>
   <CmdID>2</CmdID>
   <Data>1226</Data>                           <!-- Generic Alert -->
   <Correlator>correlator1</Correlator>
   <Item>

<Source><LocURI>./SCM/Inventory/Delivered/Package456/Operations/Install</LocUR
I></Source>
      <Target><LocURI>./SCM/Inventory/Deployed/Component1</LocURI></Target>
      <Meta>
         <Type xmlns='syncml:metinf'>
              urn:oma:at:scomo:1.0:OperationComplete
         </Type>
         <Format xmlns='syncml:metinf'>text/plain</Format>
         <Mark xmlns='syncml:metinf'>warning</Mark>
      </Meta>
      <Data>
        <![CDATA[
         <ResultCode>1200</ResultCode>          <!-- SCOMO Result Code -->
         <Identifier>Component1ID</Identifier>
        ]]>
      </Data>
   </Item>

   <Item>

<Source><LocURI>./SCM/Inventory/Delivered/Package456/Operations/Install</LocUR
I></Source>
      <Target><LocURI>./SCM/Inventory/Deployed/Component2</LocURI></Target>
      <Meta>
         <Type xmlns='syncml:metinf'>
                 urn:oma:at:scomo:1.0:OperationComplete
         </Type>
         <Format xmlns='syncml:metinf'>text/plain</Format>
         <Mark xmlns='syncml:metinf'>warning</Mark>
      </Meta>
      <Data>
        <![CDATA[
         <ResultCode>1200</ResultCode>          <!-- SCOMO Result Code -->
         <Identifier>Component2ID</Identifier>
        ]]></Data>
   </Item>
</Alert>
```

10.7 SCOMO Result Code

The Result Code of the operation MUST be sent as an integer value in Item/Data element of the Generic Alert [DMPRO] message or in response to an Exec command in case of synchronous execution. The Result Code MUST be one of the values defined below:

Result Code	Result Message	Informative Description of Status Code Usage
1200	Successful	Successful - the Request has succeeded.
1250-1299	Successful – Vendor Specified	Successful operation with vendor specified Result Code.
1400	Client Error	Client error – based on User or Device behavior.
1401	User Cancelled	User chose not to accept the operation when prompted.
1402	Download Failed	The Software Component download failed.
1403	Alternate Download Authentication Failure	Authentication was Required but Authentication Failure was encountered when downloading Software Component.

1404	Download Failed Due to Device being Out of Memory	The download failed due to insufficient memory in the Device to save the Delivery Package.
1405	Install Failed	Software Component installation failed in the Device.
1406	Install Failed Due to Device being Out of Memory	The install failed because there was not sufficient memory to install the Software Component in the Device.
1407	Failed Package Validation	Failed to validate digital signature of the Delivery Package.
1408	Remove Failed	The Software Component Remove operation failed.
1409	Activate Failed	The Software Component Activate operation failed.
1410	Deactivate Failed	The Software Component Deactivate operation failed.
1411	Not Implemented	The Device does not support the requested operation.
1412	Undefined Error	Indicates failure not defined by any other error code.
1413	Operation Rejected – Unsupported Environment	The operation was rejected because the device does not support the target environment type.
1414	Corrupted Package	A Delivery Package corruption was detected by, for example, a failed CRC check.
1415	Component Not Acceptable	The Software Component was not acceptable.
1416	Firmware Update In Progress	Operation rejected because a firmware update is in progress.
1450-1499	Client Error – Vendor Specified	Client Error encountered for operation with vendor specified Result Code.
1500	Alternate Download Server Error	Alternate Download Server error encountered.
1501	Alternate Download Server Unavailable	The Alternate Download Server was unavailable or did not respond.
1502	Alternate Download Request Time-Out	Client encountered a timeout when downloading the Delivery Package.
1503	Malformed or Bad URL for Alternate Download	The Alternate Download URL provided for alternate download is either malformed or the response from the server did not match the expected format.
1504	Alternate Download Fails Due To Network Issues	The Alternate Download failed due to network or transport level errors.
1505	Alternate Download Fails Due To Unknown Reason	The Alternate Download failed due to unknown reason.
1506	Blacklist failed	The Software Component Blacklist operation failed.
1507	Whitelist failed	The Software Component Whitelist operation failed.
1550-1599	Alternate Download Server Error – Vendor Specified	Alternate Download Server Error encountered with vendor-specified Result Code.

Table 5 – SCOMO Result Codes

10.7.1 Alert Types for the Software Component Management Object

The following alert type MUST be used in a Generic Alert [DMPRO] message originating from a Software Component Management Object. The alert type is used to identify the operation that was performed on the Device.

- urn:oma:at:scomo::1.1:OperationComplete

10.8 User Interaction Commands

This section deals with the SCOMO Client behavior associated with user interaction commands sent to the Device.

The SCOMO Server SHALL support DISPLAY Alert, CONFIRM OR REJECT Alert, and USER CHOICE Alert as described in [DMPRO].

The SCOMO Client, when installed on a device that supports user interaction, SHALL support DISPLAY Alert, CONFIRM OR REJECT Alert, and USER CHOICE Alert as described in [DMPRO].

10.8.1 Requesting User Confirmation

The SCOMO Server MAY request user confirmation before performing SCOMO operations within the Device. However, it is important to ensure that the SCOMO Client supports the ability to handle the user confirmation request.

In particular, the SCOMO Client:

- When installed on a device that supports user interaction, SHALL support DISPLAY Alert as described in [DMPRO].
- When installed on a device that supports user interaction, SHALL support CONFIRM OR REJECT Alert as described in [DMPRO].
- SHOULD support USER INPUT Alert as described in [DMPRO].
- When installed on a device that supports user interaction, SHALL support USER CHOICE Alert as described in [DMPRO].

In particular, the SCOMO Server:

- SHALL support triggering of DISPLAY Alert to DM Client.
- SHALL support triggering of CONFIRM OR REJECT Alert to DM Client.
- SHOULD support triggering of USER INPUT Alert to DM Client.
- SHALL support triggering of USER CHOICE Alert to DM Client.

10.9 Check Update

The SCOMO enabler MUST support a mechanism for configuring the frequency for client-initiated check for new version of a Software Component or Firmware. This mechanism is implemented by the configuration of Inventory/Deployed/CheckUpdateFreq node. If this node is present, the Client MUST execute the automatic check according to the value of this node, unless there is no other restriction based on user choice or policy. The Client MUST use Generic Alert and the Alert Type specified by Inventory/Deployed/CheckUpdateAlertType node.

10.10 Application Blacklisting and Whitelisting

The SCOMO enabler MAY support the blacklisting and whitelisting of software components. The mechanism for blacklisting and whitelisting of deployed components is specified in the description of Operations/Blacklist and Operations/Whitelist nodes of the Deployed subtree. Blacklisting and whitelising of downloadable delivery packages are specified in the description of Blacklist and Whitelist nodes of the Download subtree.

11.SCOMO Usage (Informative)

This section describes some common flows for the purpose of Software Component management.

11.1 Delivery of Deployment Components

SCOMO allows a Deployment Component to be delivered either via OMA DM delivery mechanism or using an alternate download mechanism such as [DLOTA].

11.1.1 OMA DM Delivery

OMA DM delivery is achieved by the DMS sending a Replace command containing the actual Deployment Component data to the /Inventory/Delivered/<X>/Data node. The DM Large Object mechanism MAY be used.

11.1.2 Alternate Download

Step 1: DMS sets the appropriate values in Download/<X>/ sub-tree as specified in section 10.2.

Step 2: DMS performs an Exec on one of the nodes under Download/<X>/Operations.

Step 3: The Device returns an appropriate Generic Alert indicating status of the requested operation as specified in section 10.6 Use of the Generic Alert.

11.2 Installation of Deployment Components

Step 1: The DMS performs an Exec on one of the supported installation nodes, such as Inventory/Delivered/<X>/Operations/Install or Inventory/Delivered/<X>/Operations/InstallInactive

Step 2: The Deployment Component installation process is initiated on the Device; upon completion, an appropriate status is returned to the DMS using a Generic Alert as specified in section 10.6 Use of the Generic Alert.

11.3 Activation/De-activation of Deployment Components

Step 1: The DMS performs an Exec on Inventory/Deployed /<X>/Operations/Activate or Inventory/Deployed /<X>/Operations/Deactivate depending on whether activation or deactivation is desired.

Step 2: The Deployment Component activation or deactivation process is initiated on the Device; upon completion, an appropriate status is returned to the DMS using a Generic Alert as specified in section 10.6 Use of the Generic Alert.

Alternatively, the activation/deactivation process MAY be performed synchronously; upon completion, a <Status> command for Exec command indicates the result of the operation.

11.4 Removal of Software Components

Step 1: The DMS performs an Exec on Inventory/Deployed /<X>/Operations/Remove or Inventory/Delivered /<X>/Operations/Remove.

Step 2: The Deployment Component removal process is initiated on the Device; upon completion, an appropriate status is returned to the DMS using a Generic Alert as specified in section 10.6 Use of the Generic Alert.

11.5 Retrieving Inventory of Deployment Components

Step 1: The DMS performs a GET on the Inventory/node (or the Inventory/Delivered or Inventory/Deployed nodes directly) to find out what Software Components are delivered or deployed on the Device.

Step 2: The Device returns the inventory to the DMS.

11.6 Blacklisting/Whitelisting of Deployment Components

Step 1: The DMS performs an Exec on Inventory/Deployed /<X>/Operations/Blacklist or Inventory/Deployed /<X>/Operations/Whitelist.

Step 2: The Deployment Component blacklisting or whitelisting is initiated on the Device; upon completion, an appropriate status is returned to the DMS using a Generic Alert as specified in section 10.6 Use of the Generic Alert.

12.Security Considerations (Informative)

Security for delivery and management of Software Components is of paramount importance. It is envisioned that authentication mechanisms supported by OMA DM protocol [DMPRO] will be used to ensure that authenticated entities can deliver/perform management operations on Software Components on the Device. Similarly, authorization for management operations on Software Components under the purview of this enabler will be based on the ACL mechanisms defined by OMA DM TND [DMTND].

The SCOMO enabler does not mandate nor restrict any mechanism to guarantee authenticity, confidentiality, and integrity of Software Components delivered to the Device. It is envisioned that existing security mechanisms for this purpose, such as Digital Signatures, SSL [SSL3.0], TLS [TLS1.0], etc., can easily work in conjunction with SCOMO.

13.Client-Initiated Update

13.1 General

This section defines the format of the client-initiated update message and specifies in which circumstances the device sends it. The message uses Generic Alert and acts as a notification to the server so that the server should investigate if an update is available. The client and server initiate appropriate action based on the Alert Type definitions below.

13.1.1 Generic Alert

The message MUST follow the Generic Alert format.

13.1.2 Alert Type

The message MUST include an Alert Type identifying the purpose of the request.

The following Alert Types are defined:

Alert Type	Definition
urn:oma:at:scomo:1.1:UpdateDeviceRequest	Device-initiated request for update of a Software Component or firmware in the device. The server SHOULD initiate the update if there is a new version of the software or firmware identified by the URI. The server MAY query the contents of the node specified by the URI.
urn:oma:at:scomo:1.1:UpdateUserRequest	User-initiated request for update of a Software Component or firmware in the device. The server SHOULD initiate the update if there is a new version of the Software Component identified by the URI. The server MAY query the contents of the node specified by the URI. The server MAY send User Interaction Commands in the same session to inform the user how the server will handle the SCOMO/Firmware update request. The Server MAY investigate if update is needed in the same session, but may also inform the user that the server will investigate this later on.
urn:oma:at:scomo:1.1:CheckVersionDeviceRequest	Device-initiated check for new version of a Software Component or firmware. The server SHOULD update the AvailableVersion in the Deployed sub-tree if a new version of the Software Component or firmware identified by the URI is available. The device MAY initiate an update.
urn:oma:at:scomo:1.1:CheckVersionUserRequest	User-initiated check for new version of a Software Component. The server SHOULD update the AvailableVersion in the Deployed sub-tree if a new version of the Software Component or firmware identified by the URI is available. The server MAY send User Interaction Commands in the same session to inform the user about the available version. The device MAY initiate an update.

13.1.3 URI

The URI in the alert, if specified, MUST point to the location of a Deployed Software Component in the Deployment sub-tree of the SCOMO management tree.

13.1.4 Data

The Data element MUST be included. A client vendor MAY use the data field to supply implementation-specific data. If there is no implementation-specific data, the value needs to be left empty.

14.Release Information

14.1 Supporting File Document Listing

Doc Ref	Permanent Document Reference	Description
Supporting Files		
[SCOMODDF]	OMA-SUP-MO_SCOMO-V1_1-20111122-C	SCOMO 1.1 Device Description File. Working file in DM_MO directory: http://www.openmobilealliance.org/tech/omna/omna-dm_mo-registry.aspx

Table 6 – Listing of Supporting Documents in SCOMO 1.1 Release

14.2 OMNA Considerations

The OMNA registry needs to add and maintain the following in the MO registry:

MO Identifier	Description	Owner	Version	MO DDF	MO Spec
urn:oma:mo:oma-scomo:1.1	SCOMO v1.1	DM WG	1.1	mo_scomo-v1.1.ddf	OMA-ER-SCOMO-V1_1

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior 1.1 version

A.2 Draft/Candidate Version 1.1 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-ER-SCOMO-V1_1	10 Jun 2010	All	First draft baseline as agreed in OMA-DM-SCOMO-2010-0008-INP_SCOMO_1_1_baseline
	03 Aug 2010	5.1	Incorporated CRs: OMA-DM-SCOMO-2010-0012R01-CR_HLR OMA-DM-SCOMO-2010-0013R01-CR_HLF_Client_Initiated_Operation OMA-DM-SCOMO-2010-0014R01-CR_Config_HLR
	17 Aug 2010	6, 6.1, 6.2, 6.3	Section 6 updated incorporating OMA-DM-SCOMO-2010-0015R01-CR_Arch_Section
	06 Jan 2011	All	Incorporated CRs: OMA-DM-SCOMO-2010-0018-CR_Import_MO OMA-DM-SCOMO-2010-0019R02-CR_Additional_Requirements OMA-DM-SCOMO-2010-0020R01-CR_Secure_Server_Initiated_Download_Requirements OMA-DM-SCOMO-2010-0021-CR_additional_requirement OMA-DM-SCOMO-2010-0023R02-CR_CI_Operation_Details
	20 Jan 2011	All	Editorial clean-up of headers and section cross-references in the document Language set to English UK
	30 Mar 2011	All	Incorporated CRs: OMA-DM-SCOMO-2011-0001R01-CR_Reboot_and_Cleanup OMA-DM-SCOMO-2011-0002-CR_Available_Version OMA-DM-SCOMO-2011-0003-CR_Client_Initiated_Operations OMA-DM-SCOMO-2011-0006-CR_Inconsistency_UI_Commands
	31 Mar 2011	9.3, 9.4	Incorporated CR: OMA-DM-SCOMO-2011-0004R01-CR_Deployed_Size
	19 May 2011	5.1, 7.1, 9.3, 9.4, 10.7, 10.2, 10.3, 10.4, 10.6	Incorporated CRs: OMA-DM-SCOMO-2011-0005R01-CR_Firmware_Support OMA-DM-SCOMO-2011-0008R04-CR_Installed_Location OMA-DM-SCOMO-2011-0011R01-CR_ER_More_Bug_Fixes OMA-DM-SCOMO-2011-0014R01-CR_Allowed_Blocked_Software OMA-DM-SCOMO-2011-0016R01-CR_Client_Initiated_Retrieving_Available_Software_Component
	09 Jun 2011	9.4	Incorporated CRs: OMA-DM-SCOMO-2011-0019R01-CR_MO_Dependencies OMA-DM-SCOMO-2011-0020R01-CR_Component_Type
	04 Aug 2011	5.1, 9.3, 9.4, 10.7, 10.9 (new)	Incorporated CRs: OMA-DM-SCOMO-2011-0009R02-CR_Check_Update_Frequ OMA-DM-SCOMO-2011-0010R05-CR_Check_Update_Frequ OMA-DM-SCOMO-2011-0013R03-CR_MetaData OMA-DM-SCOMO-2011-0021-CR_Client_Initiated_Upda OMA-DM-SCOMO-2011-0023R01-CR_Download_Failure_U MO diagram updated.
	22 Aug 2011	7.2.1, 14.1, 14.2, 14.3, Appendix C, Appendix D,	Incorporated CRs: OMA-DM-SCOMO-2011-0033-CR_Reboot_Clarification OMA-DM-SCOMO-2011-0030-CR_Release_Information OMA-DM-SCOMO-2011-0031-CR_SCR

Document Identifier	Date	Sections	Description
	08 Sep 2011	9.4, C.4, C.5, C.6, 9.3,	Incorporated CRs: OMA-DM-SCOMO-2011-0027R02-CR_Component_Blacklisting_Whitelisting OMA-DM-SCOMO-2011-0029R01-CR_Additional_Metadata OMA-DM-SCOMO-2011-0034R03-CR_SCR_Updates OMA-DM-SCOMO-2011-0036R01-CR_MO_Diagram OMA-DM-SCOMO-2011-0035-CR_Updates_from_Closure_Review
	12 Sep 2011	9.3, 9.4	Incorporated CRs: OMA-DM-SCOMO-2011-0038R03-CR_Download_Blacklist_Whitelist_Support
	15 Sep 2011	All	Incorporated CRs: OMA-DM-SCOMO-2011-0025R01-CR_HLF_Update OMA-DM-SCOMO-2011-0028R02-CR_Postponing_Requirements OMA-DM-SCOMO-2011-0040-CR_Edit_section_reference OMA-DM-SCOMO-2011-0041R01-CR_Closure_Review_Comments_Addressed
	17 Oct 2011	All	Incorporated CR: OMA-DM-SCOMO-2011-0042-CR_TechWriter_Review_Update
	31 Oct 2011	9.4, 10.2, 10.7, 10.9, 11	Incorporated CR: OMA-DM-SCOMO-2011-0044R01-CR_Addressing_CONR_Comments_on_ER
Draft Version OMA-ER-SCOMO-V1_1	09 Nov 2011	14.1	Updated the SUP file to its latest version in the list of documents
Candidate Version OMA-ER-SCOMO-V1_1	22 Nov 2011	N/A	Status changed to Candidate by TP Ref # OMA-TP-2011-0410-INP_SCOMO_V1.1_ERP_and_ETR_for_Candidate_approval

Appendix B. Use Cases (Informative)

B.1 Check for New Version of a Software

A user has installed several applications available through an application store. The user desires to use the latest version of the application. As such he wants to check if a new version of the application is available and if so, he can initiate the download at a convenient time.

B.1.1 Short Description

Arnold is a device user and very interested in latest versions of mobile applications. He is subscribed to the application store of “SmartApps Co.” After using an application for few weeks, he is unsure if he is using the latest version. Arnold wants to check if his application store offers the latest version of the application. If so, he plans to download it at a convenient time.

Arnold initiates check for new version of the application through the application user interface. The application interacts with the SCOMO client to initiate appropriate Generic Alert to the SCOMO Server. The SCOMO Server processes the request and provides the latest version of the application. The application displays this information to the user.

B.1.2 Market benefits

Improved consumer experience.

B.2 Application Blacklisting

A Management Authority (Enterprise, Operator or Service Provider) wants to ensure that the users are not using harmful applications in their Devices.

B.2.1 Short Description

ABC Company allows its employees to download any applications they want. The company learns about the availability of new free software application that is harmful to users and the company. The company wants to ensure that users are not using this application in the Devices.

At any time, the Management Authority can perform remote inventory query through the SCOMO Server and retrieve the list of Software Components installed in a device. The Management Authority can initiate remote blacklist operation on a specific component in the device through the SCOMO Server. In addition, the Management Authority can provide a list of all Blacklisted applications to the Device in case the user tries to download and install additional blacklisted applications.

B.2.2 Market benefits

Improved safety and security of consumers and businesses and cost reduction.

Appendix C. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

C.1 SCR for SCOMO 1.1 Tree Structure

Item	Function	Reference	Requirement
SCOMO-T-001-M	Use of appropriate Management Object identifier for the SCOMO node	Section 9.4	
SCOMO-T-002-M	Support for Required nodes under root node	Section 9.4	
SCOMO-T-003-O	Support for Optional nodes	Section 9.4	
SCOMO-T-004-M	Support for Required nodes under an Optional node if the Optional node is supported	Section 9.4	

C.2 SCR for SCOMO 1.1 Client

Item	Function	Reference	Requirement
SCOMO-C-001-M	Support for Download	Section 7.2	SCOMO-C-002-O OR SCOMO-C-005-O
SCOMO-C-002-O	Support for Alternate Download	Section 7.1.2	SCOMO-C-026-O AND SCOMO-C-027-O
SCOMO-C-003-O	Support for Add of <x>/Download/<x>/Pkg URL	Section 9.4	
SCOMO-C-004-O	Support for Replace of <x>/Download/<x>/Pkg URL	Section 9.4	
SCOMO-C-005-O	Support for OMA DM Download based on Replace mechanism	Section 7.1.2	SCOMO-C-028-O AND SCOMO-C-029-O
SCOMO-C-006-O	Support for Add of <x>/ Inventory/Delivered/<X >/Data	Section 9.4	
SCOMO-C-007-O	Support for Replace of <x>/ Inventory/Delivered/<X >/Data	Section 9.4	
SCOMO-C-008-O	Support Inactive State for Deployment Component	Section 7.1.1.1	
SCOMO-C-009-M	Support Active State for Deployment Component	Section 7.1.1.2	
SCOMO-C-010-M	Support Removed State for Deployment Component	Section 7.1.1.3	
SCOMO-C-011-O	Support Activate Primitive for	Section 7.1.1.4	

Item	Function	Reference	Requirement
	Deployment Component		
SCOMO-C-012-O	Support Deactivate Primitive for Deployment Component	Section 7.1.1.4	
SCOMO-C-013-M	Support Remove Primitive for Deployment Component	Section 7.1.1.4	
SCOMO-C-014-M	Support Not Downloaded State for Delivery Package	Section 7.2.1.1	
SCOMO-C-015-O	Support Delivered State for Delivery Package	Section 7.2.1.2	
SCOMO-C-016-M	Support Installed State for Delivery Package	Section 7.2.1.3	
SCOMO-C-017-M	Support Removed State for Delivery Package	Section 7.2.1.4	
SCOMO-C-018-O	Support Download Primitive for Delivery Package	Section 7.2.1.5	
SCOMO-C-019-O	Support DownloadInstall Primitive for Delivery Package	Section 7.2.1.5	
SCOMO-C-020-O	Support DownloadInstallInactive Primitive for Delivery Package	Section 7.2.1.5	
SCOMO-C-021-O	Support Install Primitive for Delivery Package which is in the Delivered sub-tree	Section 7.2.1.5	
SCOMO-C-022-O	Support InstallInactive Primitive for Delivery Package	Section 7.2.1.5	
SCOMO-C-023-M	Support Remove Primitive for Delivery Package	Section 7.2.1.5	
SCOMO-C-024-O	Support synchronous result reporting	Section 10.5	
SCOMO-C-025-O	Support asynchronous result reporting	Section 10.6	
SCOMO-C-026-O	Support for OMA-DM Download using PkgURL node	Section 10.6	SCOMO-C-003-O OR SCOMO-C-004-O
SCOMO-C-027-O	Support for Primitives on Download sub-tree	Section 9.4	SCOMO-C-018-M OR SCOMO-C-019-O OR SCOMO-C-020-O
SCOMO-C-028-O	Support for OMA-DM Download using Data node	Section 11.1.1	SCOMO-C-006-O OR SCOMO-C-007-O
SCOMO-C-029-O	Support installation Primitives on Delivered sub-tree	Section 7.1.2	SCOMO-C-021-O OR SCOMO-C-022-O

Item	Function	Reference	Requirement
SCOMO-C-030-O	After installation reflect the Deployment Component metadata (provided by the DM System) in corresponding nodes under the Deployed sub-tree	Section 7.1	
SCOMO-C-031-M	Support state transitions that appear to be atomic (for Deployment Component). If a state transition fails the - reverse the operation and make sure the Deployment Components remain in the previous state	Section 7.1.1	
SCOMO-C-032-M	Support state transitions that appear to be atomic (for Delivery Package). If a state transition fails the - reverse the operation and make sure the Delivery Package remains in the previous state	Section 7.2.1	
SCOMO-C-033-M	Create corresponding nodes of each Deployment Component in the Deployed tree	Section 7.2.1.3	
SCOMO-C-034-M	When executing Remove Primitive of Delivery Package - delete the delivery Package from the Device and delete corresponding nodes which represented it in the Delivered tree	Section 7.2.1.4	
SCOMO-C-035-O	Support EnvType node	Section 9.4	SCOMO-C-036
SCOMO-C-036-O	Reject a Software Component if its indicated EnvType is not supported	Section 9.4	
SCOMO-C-037-M	When using alternate download mechanism – copy the values of leaf nodes from the Download sub-tree to the corresponding leaf nodes in the Delivered tree	Section 9.4	
SCOMO-C-038-M	Support result reporting	Section 10	SCOMO-C-024 OR SCOMO-C-025
SCOMO-C-039-M	After successful	Section 10	

Item	Function	Reference	Requirement
	installation of Deployment Components, create their nodes in the Deployed sub-tree		
SCOMO_C-040-O	Support for reboot before install, after install or before and after install.	Section 7.2.1	
SCOMO-C-041-M	Support result reporting for Firmware Update.	Section 10.7	
SCOMO-C-042-M	Support for User Interaction Commands – Display Alert, Confirm or Reject Alert, User choice Alert.	Section 10.8	
SCOMO-C-043-M	Creation of <x> nodes and sub nodes under Download Sub-tree before performing Exec command.	Section 10.2	
SCOMO-C-044-M	Creation of <x> nodes and sub nodes under Delivered Sub-tree before performing Exec command.	Section 10.3	
SCOMO-C-045-M	Creation of <x> nodes and sub nodes under Deployed Sub-tree before performing Exec command.	Section 10.4	
SCOMO-C-046-M	Support the mechanism for configuring the frequency for client-initiated check for new version of Software Component or firmware.	Section 10.9	
SCOMO_C-047-M	Support for firmware update considering firmware as a software.	Section 13	SCOMO-C-048-M
SCOMO-C-048-M	Support for Generic Alerts for Client-Initiated Update	Section 13.1	

C.3 SCR for SCOMO 1.1 Server

Item	Function	Reference	Requirement
SCOMO-S-001-M	Support for the Software Component Management Object	Section 9.4	
SCOMO-S-002-M	Support triggering of DISPLAY Alert	Section 10.8, Section 10.8.1	

Item	Function	Reference	Requirement
SCOMO-S-003-M	Support triggering of CONFIRM OR REJECT Alert	Section 10.8.1	
SCOMO-S-004-O	Support triggering of USER INPUT Alert	Section 10.8.1	
SCOMO-S-005-M	Support triggering of USER CHOICE Alert	Section 10.8.1	
SCOMO-S-006-O	Support for handling of Client-Initiated Updates.	Section 13.1	
SCOMO-S-007-O	Support for indicating to the SCOMO Client that it needs to reboot either before install, after install or before and after install.	Section 7.2.1	

C.4 SCR for DM System

Item	Function	Reference	Requirement
SCOMO-DMS-001-M	Provide metadata with the Deployment Component	Section 7.1, section 9.4	
SCOMO-DMS-002-M	Provide value for PkgID to uniquely identify the Delivery Package within the SCOMO tree	Section 9.4	
SCOMO-DMS-003-M	Provide Package Name	Section 9.4	
SCOMO-DMS-004-O	Provide value for EnvType	Section 9.4	
SCOMO-DMS-005-O	Provide value for PkgType	Section 9.4	
SCOMO-DMS-006-M	Provide value for ID to uniquely identify the Deployment Component within the SCOMO tree	Section 9.4	