Device Management Architecture

Approved Version 2.0 – 09 Feb 2016

Open Mobile Alliance

OMA-AD-DM-V2_0-20160209-A

Contents

1. SCOPE (INFORMATIVE) 4

2. REFERENCES 5

2.1 NORMATIVE REFERENCES 5

2.2 INFORMATIVE REFERENCES 5

3. TERMINOLOGY AND CONVENTIONS 6

3.1 CONVENTIONS 6

3.2 DEFINITIONS 6

3.3 ABBREVIATIONS 6

4. INTRODUCTION (INFORMATIVE) 7

4.1 VERSION 1.2 7

4.2 VERSION 1.3 8

4.3 VERSION 2.0 8

5. ARCHITECTURAL MODEL 9

5.1 DEPENDENCIES 9

5.2 ARCHITECTURAL DIAGRAM 9

5.3 FUNCTIONAL COMPONENTS AND INTERFACES/REFERENCE POINTS DEFINITION 9

5.3.1 Protocol Endpoints 9

5.3.2 Interfaces 10

5.4 SECURITY CONSIDERATIONS 10

APPENDIX A. CHANGE HISTORY (INFORMATIVE) 12

A.1 APPROVED VERSION HISTORY 12

APPENDIX B. FLOWS (INFORMATIVE) 13

B.1 CLIENT INITIATED SESSION 13

B.2 SERVER INITIATED SESSION 14

B.3 UI INTERACTION USING WEB BROWSER COMPONENT 15

Figures

Figure 1: Device Management Architectural Diagram using interfaces 9

Figure 2: Client Initiated Management Call Flow 13

Figure 3: Server Initiated Management Call Flow 14

Figure 4: UI Interaction with Web Browser Call Flow 15

1. Scope (Informative)

The scope of the Device Management architecture document is to define the architecture for the Device Management v2.0 enabler. This document fulfils the functional capabilities and information flows needed to support this enabler as described in the Device Management requirements document [DM-RD].

2. References

2.1 Normative References

[DM-RD] “Device Management Requirements”, Open Mobile Alliance™, OMA-RD-DM-V2_0, 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

2.2 Informative References

[ARCH-PRINC] “OMA Architecture Principles”, Version 1.2, Open Mobile Alliance™, OMA-ArchitecturePrinciples-V1_2, URL:http://www.openmobilealliance.org/

[OMADICT] “Dictionary for OMA Specifications”, Version 2.9, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL:http://www.openmobilealliance.org/

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 [OMADICT] for all definitions used in this document.

3.3 Abbreviations

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

4. Introduction (Informative)

Device Management refers to the management of Device configuration and other managed objects of Devices from the point of view of the Management Authorities. Device Management includes, but is not restricted to setting initial configuration information in Devices, subsequent updates of persistent information in Devices, retrieval of management information from Devices, execute primitives on Devices, and processing events and alarms generated by Devices.

Device Management allows network operators, service providers or corporate information management departments to carry out the procedures of configuring devices on behalf of the end user (customer).

4.1 Version 1.2

Device management is the generic term used for technology that allows third parties to carry out the difficult procedures of configuring devices on behalf of the end user (customer). Third parties would typically be operators, service providers or corporate information management departments.

Through device management, an external party can remotely set parameters, conduct troubleshooting servicing of terminals, install or upgrade software. In broad terms, device management consists of three parts:

- Protocol and mechanism: The protocol used between a management server and a device
- Data model: The data made available for remote manipulation, for example browser and mail settings
- Policy: The policy decides who can manipulate a particular parameter, or update a particular object in the device

The specifications in the Device Management enabler Version 1.2 address the first part of device management above, the protocol and mechanism. More particularly, this enabler release addresses the management of devices by specifying a protocol and management mechanism that may be exposed by an OMA DM client and targeted by an OMA DM server.

The architecture of the Device Management enabler anticipates the needs of the market actors to differentiate their products through vendor-specific extensions while providing a core parameter set that can be relied upon in all terminals exposing this standardized interface.

The design of the architecture follows the OMA architecture principle [ARCH-PRINC] of Network Technology Independence by separating the bearer-neutral requirements from bearer-specific bindings. The described architecture also anticipates additional bearer and proxy types, as any are identified, without requiring a respecification of previously released documents. This preserves vendor and customer investment while supporting the scaling required by future innovations.

There are three parts to the object schema that provide break-points between more general and more specific parameters:

- A top level management object which is bearer-neutral;
- A set of bearer-specific parameters;
- Sub-tree(s) for exposing vendor-specific parameters.

By composing the management objects in this way, it becomes possible for a device management authority to:

- Target generic requirements that span all implementations;
- Focus on bearer-specific idiosyncrasies of a given networking environment;
- Activate terminal-specific behaviour by adjusting vendor-specific parameters.

In a wireless environment, the crucial element for device management protocol is the need to efficiently and effectively address the characteristics of devices including low bandwidth and high latency and to provide for support of these management operations remotely, over-the-air.

4.2 Version 1.3

OMA DM Version 1.3 reused the architecture from OMA DM Version 1.2. It does introduce new notification, transport protocols and a new DM Server to DM Server interface for delegation.

4.3 Version 2.0

OMA DM Version 2.0 reuses the Management Objects which is designed for DM Version 1.3 or earlier DM Protocols.OMA DM Version 2.0 introduces new Client-Server DM protocol based on HTTP following RESTful architectural design patterns.

OMA DM Version 2.0 also introduces new user interaction method on Device Management using Web Browser Component.

5. Architectural Model

5.1 Dependencies

None.

5.2 Architectural Diagram

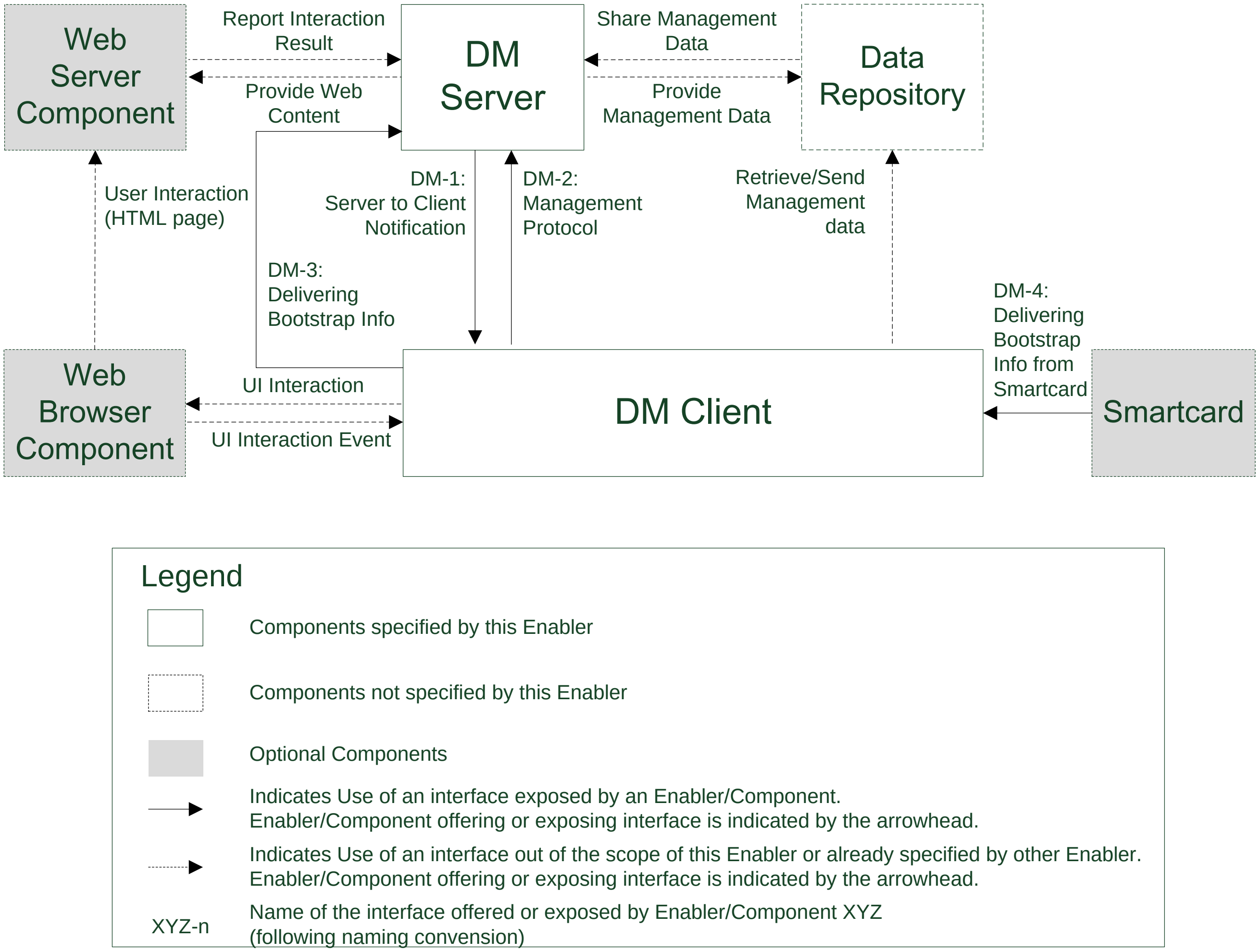
Web Server Component
Report Interaction Result
Provide Web Content
DM Server
Share Management Data
Provide Management Data
Data Repository
User Interaction (HTML page)
DM-1: Server to Client Notification
DM-2: Management Protocol
Retrieve/Send Management data
DM-3: Delivering Bootstrap Info
DM-4: Delivering Bootstrap Info from Smartcard
Web Browser Component
UI Interaction
UI Interaction Event
DM Client
Smartcard
Legend
Components specified by this Enabler
Components not specified by this Enabler
Optional Components
Indicates Use of an interface exposed by an Enabler/Component.
Enabler/Component offering or exposing interface is indicated by the arrowhead.
Indicates Use of an interface out of the scope of this Enabler or already specified by other Enabler.
Enabler/Component offering or exposing interface is indicated by the arrowhead.
XYZ-n
Name of the interface offered or exposed by Enabler/Component XYZ
(following naming convention)

Figure 1: Device Management Architectural Diagram using interfaces

5.3 Functional Components and Interfaces/reference points definition

5.3.1 Protocol Endpoints

5.3.1.1 DM Client

The DM Client is the abstract software component that conforms to the requirements for DM Clients specified in this enabler.

5.3.1.2 DM Server

The DM Server is the abstract software component that conforms to the requirements for DM Servers specified in this enabler.

5.3.1.3 Web Server Component

The Web Server Component is a optional logical server responsible to deliver web content for UI interaction with a Web Browser Component on the Device. The DM Server MAY request DM Client to access to this component using Web Browser Component. The DM Server also MAY receive the result of UI interaction from this component.This component is not specified in this enabler.

5.3.1.4 Web Browser Component

The Web Browser Component is an optional logical component responsible to provide UI interaction functionality for the DM Client. DM Client MAY trigger the Web Browser to process web contents provided by the Web Server Component. This component is not specified in this enabler

Informative Note: This component could be implemented as using a standalone web browser application or the browser window could be implemented as web browser component as part of the DM Client application.

5.3.1.5 Data Repository

The data repository is a logical Server, and the DM Client can retrieve and send management data to and from this component by using HTTP Methods or other transport protocols.The DM Server can exchange the management data with this entity

5.3.2 Interfaces

5.3.2.1 DM-1 Server-to-Client Notification

This provides an interface over which Servers may send device management notification to Clients to initiate a Device Management session.

5.3.2.2 DM-2 Device Management Protocol

This provides an interface over which Servers may send device management commands to Clients and Clients may return status and Alerts to Servers. This is an interface that is bearer neutral and offers many standardized bindings including HTTP and HTTPS.

This provides an interface to receive the device management commands from the DM Server for DM Client, over HTTP/HTTPS communication established from the DM Client to the DM Server.

The DM Client MAY report the execution status of the device management command and/or event notifications (Alerts in DM1.x) through this interface.

5.3.2.3 DM-3 Retrieving Bootstrap Information

This provides an interface to retrieve the bootstrap information.

5.3.2.4 DM-4 Delivering Bootstrap Information from Smartcard

The DM Client may be bootstrapped from data stored on a Smartcard. This data will contain the information needed by the DM Client to be bootstrapped. This is a one-way interface.

5.4 Security Considerations

DM 2.0 enabler requires a high level of security, due to the data that is being handled. If a DM Client were to be configured by a rogue DM Server, it is possible for the device to be ruined. If a rogue DM Client were to be configured by a DM Server, it is possible for the data from that DM Client to propagate into the network (if the DM Client were masquerading as another device).

In the end, the service provider:

- provides mutual authentication between DM Server and DM Client.
- Supports mutual authentication between Data Repository and DM Client.
- does not allow un-authorized access from DM Server to DM Client.
- does not allow un-authorized access from DM Client to DM Server.
- provides secure communication channel between DM Server and DM Client.
- supports secure communication channel between Data Repository and DM Client.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-AD-DM-V2_0-20160209-A	09 Feb 2016	Status changed to Approved by TP TP Ref # OMA-TP-2016-0032-INP_DM_V2_0_ERP_for_final_Approval

Appendix B. Flows (informative)

B.1 Client Initiated Session

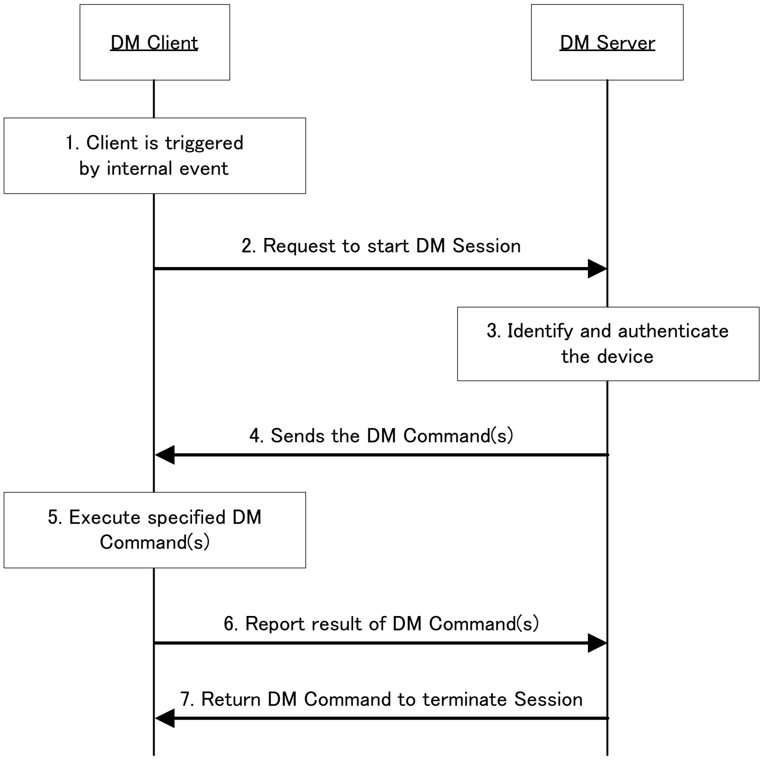
DM Client
DM Server
1. Client is triggered by internal event
2. Request to start DM Session
3. Identify and authenticate the device
4. Sends the DM Command(s)
5. Execute specified DM Command(s)
6. Report result of DM Command(s)
7. Return DM Command to terminate Session

Figure 2: Client Initiated Management Call Flow

This call flow is triggered by internal device event.

1. The DM Client is triggered by internal event, such as scheduled timer.
2. The DM Client sends the request to the DM Server to start Client Initiated Session.
3. The DM Server identifies and authenticates the device.
4. The DM Server sends the DM Command(s) to the Device.
5. The DM Client executes specified DM Command(s).
6. The DM Client reports result of the DM Command operation(s).
7. The DM Server returns DM Command to the Device to terminate the management session.

B.2 Server Initiated Session

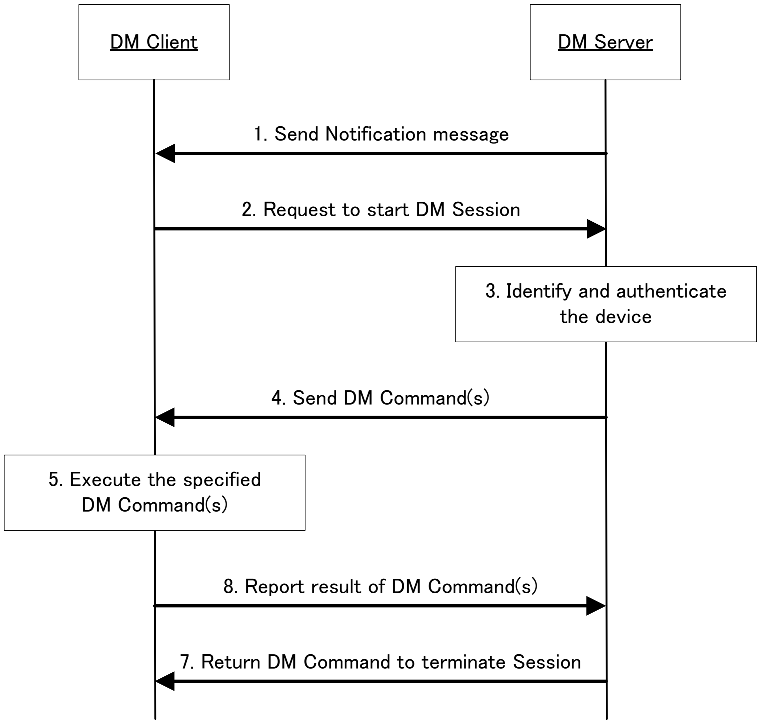
DM Client
DM Server
1. Send Notification message
2. Request to start DM Session
3. Identify and authenticate the device
4. Send DM Command(s)
5. Execute the specified DM Command(s)
8. Report result of DM Command(s)
7. Return DM Command to terminate Session

Figure 3: Server Initiated Management Call Flow

This call flow is triggered by a Notification message sent by DM Server:

1. The DM Server sends Notification message to the DM Client.
2. The DM Client starts Server Initiated Session.
3. The DM Server identifies and authenticates the device.
4. The DM Server sends the DM Command(s) to the Device.
5. The DM Client executes the specified DM Command(s).
6. The DM Client reports the result of the DM Command operation(s).
7. The DM Server returns DM Command to terminate the management session.

B.3 UI Interaction using Web Browser component

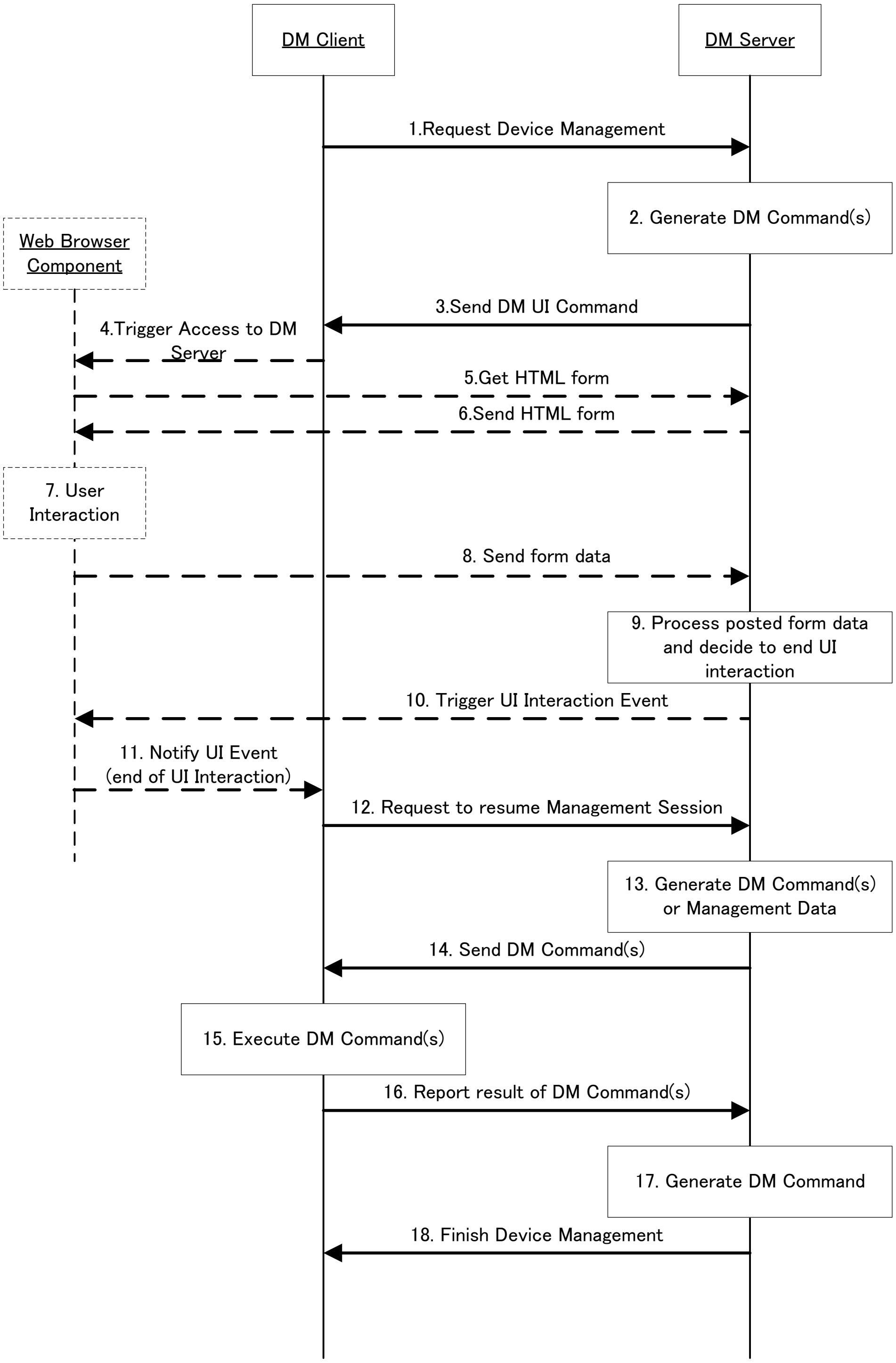
DM Client
DM Server
1.Request Device Management
2. Generate DM Command(s)
Web Browser Component
3.Send DM UI Command
4.Trigger Access to DM Server
5.Get HTML form
6.Send HTML form
7. User Interaction
8. Send form data
9. Process posted form data and decide to end UI interaction
10. Trigger UI Interaction Event
11. Notify UI Event (end of UI Interaction)
12. Request to resume Management Session
13. Generate DM Command(s) or Management Data
14. Send DM Command(s)
15. Execute DM Command(s)
16. Report result of DM Command(s)
17. Generate DM Command
18. Finish Device Management

Figure 4: UI Interaction with Web Browser Call Flow

This call flow contains UI interaction using Web Browser component.

1. The DM Client sends the request to the DM Server to start Management Session.
2. The DM Server generates DM Command(s) internally.
3. The DM Server sends the DM Command(s) to be performed on the Device.
4. The DM Client triggers Web Browser to access the DM Server internally.
5. The Web Browser requests to get a HTML form document for UI interaction.
6. The DM Server sends a HTML form document to the Web Browser.
7. The Web Browser performs UI interaction with User internally.
8. The Web Browser sends form data to the DM Server.
9. The DM Server processes posted form data, and decides to end UI interaction.
10. The DM Server triggers the Web Browser to notify UI interaction event.
11. The Web Browser notifies the UI interaction event to the DM Client.
12. The DM Client requests to resume the Management Session.
13. The DM Server generates DM Command(s) internally.
14. The DM Server sends the DM Command(s) to be performed on the Device
15. The DM Client executes specified DM Command operation(s).
16. The DM Client reports result of the DM Command operation(s).
17. The DM Server generates DM Command internally.
18. The DM Server sends DM Command to the Device for finishing the Management Session.