Download Over the Air Architecture

Approved Version 2.0 – 29 Mar 2011

Open Mobile Alliance

OMA-AD-DLOTA-V2_0-20110329-A

Contents

Figures

Tables

1. Introduction

OMA Download Over-the-Air (DLOTA) provides a flexible mechanism for downloading Media Objects of any type and size from a network. A typical Media Object targeted by OMA DLOTA is downloaded and stored in the device, for example, in order to personalise the device or enhance its functionality. Examples of such Media Objects are ring-tones, background images, music/video files and applications.

Concerning the download of applications, DLOTA version 2.0 improves the previous version [DLOTAv1.0] by removing the constraint on the download of Java™ MIDlets [MIDP] and thus specifying a truly execution environment neutral protocol.

OMA DLOTA version 2.0 is also agonistic of the protection mechanism, i.e. OMA DRM [DRM2.0] or others, used to protect the Media Object.

1.1 Use Cases

While OMA DLOTA version 1.0 [DLOTAv1.0] provides a general framework for downloading Media Objects, OMA DLOTA version 2.0 extends OMA DLOTA version 1.0 to support following new use cases.

- Functions that are already provided by OMA DLOTA version 1.0:
    1. Basic download
    2. Combined delivery
- New functions that extends OMA DLOTA version 1.0:
    3. Updating and removing Media Objects
    4. Download from multiple servers
    5. Download of compound objects and multiple objects
    6. Download of chunked Media Objects
    7. Control of User confirmation prompt
    8. Support for resumable download session
    9. Authentication of trusted entity and content integrity check
- New major functions that satisfy market's requirements:
    10. Pre-downloading of Media Objects
    11. Download timing reservation
    12. Progressive download
    13. Download OTA over broadcast protocols

Details of these new use cases and the requirements are specified in the OMA DLOTA version 2.0 requirements specification [DLREQ].

1.2 Requirements

Requirements that satisfy the use cases listed above are defined in the OMA DLOTA version 2.0 requirements specification [DLREQ].

1.3 Planned Phases

This architecture specification targets phase 2.0 of OMA DLOTA.

OMA DLOTA version 1.0 (DLOTAv1.0) provides a mechanism for User-initiated download of content, such as ring-tones, images, and applications. While OMA DLOTA provides much of the functionality needed to provide for a more reliable download solution than basic HTTP, other protocols exist in the mobile industries that provide functionality beyond that of OMA DLOTA.

OMA DLOTA version 2.0 (DLOTAv2.0) is an evolution of the OMA DLOTA version 1.0 protocol. The purpose of OMA DLOTA version 2.0 is to add functionality that was missing from the previous version of the protocol and at the same time to guarantee backward compatibility.

2. References

2.1 Normative References

No normative references.

2.2 Informative References

[BCAST] "OMA Broadcast", version 1.0, Open Mobile Alliance™, OMA-ERP-BCAST-V1_0. URL: http://www.openmobilealliance.org/

[BCMCS] "Broadcast/Multicast Services - Stage 1", 3GPP2 S.R0030-A v1.0 – Revision A, URL: http://www.3gpp2.org/

[DICTIONARY] "Dictionary for OMA Specifications", version 2.3, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_3. URL: http://www.openmobilealliance.org/"

[DLOTAv1.0] "OMA Download", version 1.0, Open Mobile Alliance™, OMA-DL-V1_0. URL: http://www.openmobilealliance.org/

[DLREQ] "Download Over-The-Air Requirements", version 2.0, Open Mobile Alliance™, OMA-RD-DLOTA-V2_0. URL: http://www.openmobilealliance.org/

[DRM1.0] "OMA Digital Rights Management", version 1.0, Open Mobile Alliance™, OMA-DRM-V1_0. URL: http://www.openmobilealliance.org/

[DRM2.0] "OMA Digital Rights Management", version 2.0, Open Mobile Alliance™, OMA-ERP-DRM-V2_0. URL: http://www.openmobilealliance.org/

[DVB-H] "Digital Video Broadcasting (DVB): Transmission System for Handheld Terminals (DVB-H)", ETSI EN 302 304. URL: http://portal.etsi.org/

[HTTP] J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.1", RFC2616, June 1999. URL: http://www.ietf.org/rfc/rfc2616.txt

[MBMS] "Multimedia Broadcast/Multicast Service (MBMS); (Release 6)", 3GPP TS 22.146 v6.6.0. URL: http://www.3gpp.org/ftp/Specs/html-info/22146.htm
"Multimedia Broadcast/Multicast Service (MBMS) User services; (Release 6)", 3GPP TS 22.246 v6.2.0. URL: http://www.3gpp.org/ftp/Specs/html-info/22246.htm

[MIDP] "JSR-000118 Mobile Information Device Profile 2.0", Java Community Process, URL: http://jcp.org/aboutJava/communityprocess/final/jsr118/

[MMS] "OMA Multimedia Messaging Service", version 1.3, Open Mobile Alliance™, OMA-ERP-MMS-V1_3. URL: http://www.openmobilealliance.org/

[SSL] A. Frier, P. Karlton, and P. Kocher, "The SSL 3.0 Protocol", Netscape Communications Corp., Nov 18, 1996.

[TLS] T. Dierks and C. Allen, "Transport Layer Security (TLS) Version 1.0", RFC 2246, Jan 1999. URL: http://www.ietf.org/rfc/rfc2246.txt

[WAP Push] "Push Architectural Overview", WAP Forum™, WAP-250-PushArchOverview-20010703-a. URL: http://www.openmobilealliance.org/

[WSP] "Wireless Session Protocol Specification", WAP Forum™, WAP-230-WSP-20010705-a. URL: http://www.openmobilealliance.org/

[WTLS] "Wireless Transport Layer Security", WAP Forum™, WAP-261-WTLS-20010406-a. URL: http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

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

3.2 Definitions

Compound Objects	Content which is composed of one or more Media Objects which have relationship each other.
Content Handler	An entity in the mobile device responsible for the processing of a particular media type. The content handler typically handles issues related to installation of content, in addition to execution of content. The actual processing of retrieved content is outside the scope of this specification.
Content Provider	A human entity or a company which provides Media Objects to the User
Download Agent	A User agent in the device responsible for downloading a Media Object described by a download descriptor.
Download Descriptor	Metadata about a Media Object and instructions to the download agent for how to download it.
Download OTA Server	A logical entity which encompasses the all server components (i.e. Presentation Server, Download Server, and Status Report Server).
Download Server	A Web server hosting Media Objects available for download using the DLOTA protocol
Media Object	A resource on a Web server that can be downloaded.
Multiple Objects	Content which is composed of one or more Media Objects. However the Media Objects do not always have relationship each other.
Presentation Server	A Web server presenting a download service to the User.
Status Report Server	A Web server accepting status reports from the download agent.
User	An entity which uses services as defined in [DICTIONARY].

3.3 Abbreviations

BCMCS	Broadcast Multicast Services
DLOTA	Download Over The Air
DRM	Digital Rights Management
DVB-H	Digital Video Broadcasting for Handheld
HTTP	Hypertext Transfer Protocol
MBMS	Multimedia Broadcast Multicast Service
MIDP	Mobile Information Device Profile
MMS	Multimedia Messaging Service
OMA	Open Mobile Alliance
OTA	Over The Air
SSL	Secure Socket Layer
TLS	Transport Layer Security
WSP	Wireless Session Protocol
WTLS	Wireless Transport Layer Security

4. Architecture Model

4.1 Dependencies

OMA DLOTA version 2.0 provides a flexible mechanism for downloading Media Objects of any type and size from a network. Therefore, OMA DLOTA version 2.0 can be used by the other OMA enablers to download Media Objects. Examples of such enablers are OMA Digital Rights Management [DRM1.0] [DRM2.0] and Device Management [OMADM].

On the other hand, since OMA DLOTA version 2.0 is a bearer independent protocol, any underlying protocols such as HTTP [HTTP] and WSP [WSP] can be used. Security protocols such as SSL [SSL]/TLS [TLS] or WTLS [WTLS] can also be used to download Media Objects in a secure manner. If a broadcast network such as MBMS [MBMS] and DVB-H [DVB-H] is used to download Media Objects, OMA DLOTA version 2.0 may need to rely on OMA BCAST enabler release [BCAST] for service discovery, content protection and so on.

An example of the relationship between the enablers is illustrated in Figure 1.

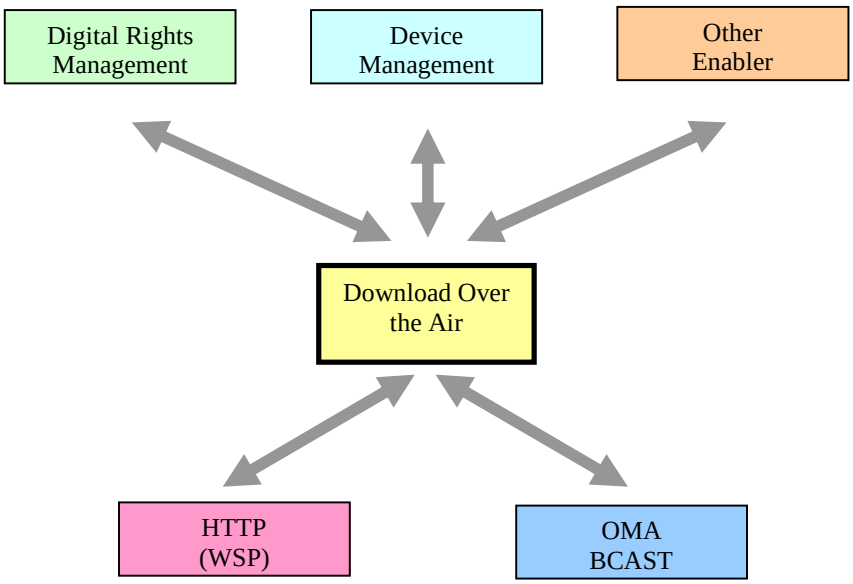

Figure 1: Relationship between OMA Download OTA and other OMA Enablers

4.2 Architectural Diagram

Figure 2 shows functional architecture of OMA DLOTAv2.0.

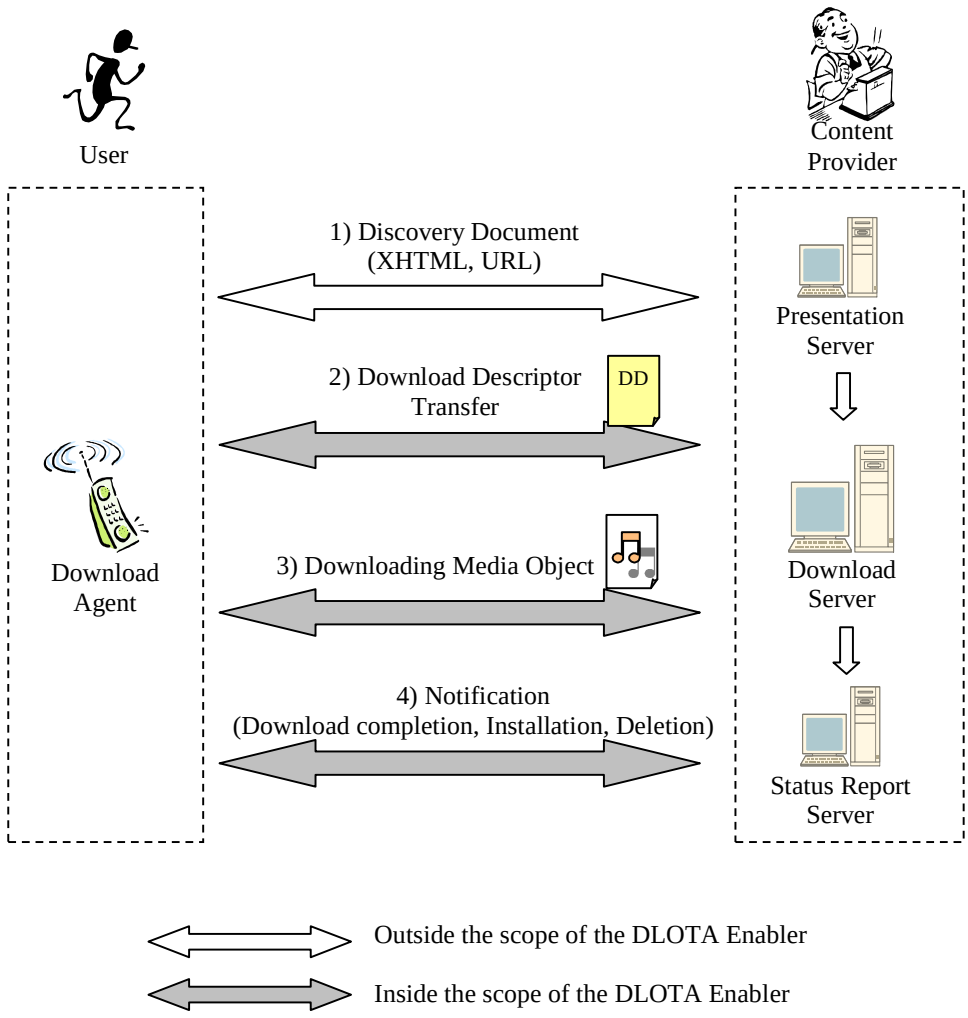

Figure 2: Functional architecture

4.3 Functional Components and Interfaces

This section defines actors and entities of the OMA DLOTA system.

There is the Download Agent at the User side, and there are the Presentation Server, the Download Server and the Status Report Server at the content provider side. The User is a human entity or an application which consumes the Media Object. The content provider is a human entity or a company which provides Media Objects to the Download Agent which then delivers Media Objects to the User. The definitions of the various entities within the DLOTAv2.0 architecture are as follows.

- Download Agent (Download User Agent)

  Download Agent or Download User Agent is a User agent in the device responsible for downloading a Media Object described by a Download Descriptor. It is responsible for the download transaction from the client perspective. It is triggered by the reception or activation of a Download Descriptor.

  The Download Agent works together with the User (i.e. the human User of Media Objects). For example, the Download Agent requests confirmation from the User to determine whether to download the Media Object or not. The Download Agent also makes the Media Object available to the User after the completion of the installation process.

- Presentation Server

  The term Presentation Server describes the logical server function that presents a download service to the User. It is one of the possible discovery mechanisms. The client device may browse a Web or WAP page at the presentation server and be redirected to the Download Server for the OMA Download transaction.

- Download Server

  A term Download Server describes the logical server function that hosts Media Objects that are available for download. It is responsible for the download transaction from the server perspective. It handles download session management including actions triggered by the installation status report.

1. Status Report Server

   A term Status Report Server describes the logical server function that accepts status reports from the Download Agent.

Note that the logical server functions (Presentation Server, Download Server and Status Report Server) do not have to be provided by the same service provider. Therefore, it is possible for a content provider to have only a web portal which refers to Media Objects that are stored by different servers and owned by different service providers.

4.4 Flows

An example of the communication procedure between each entity which constitutes the functional architecture (Figure 2) is explained below.

1) Discovery Document

A User discovers a Media Object on the network by using a discovery application. How to find the Media Object is outside the scope of DLOTA. However, there are several ways to find the Media Object. For example, a picture editor may discover pictures, a melody composer may discover melodies, and an application manager may discover applications (e.g. games) on dedicated Web sites; an email message may contain Web addresses to Media Objects available for downloading, a Web browser may be used to discover the Media Object over the network. These types of applications are collectively referred to as a discovery application.

2) Download Descriptor Transfer

The Download Agent downloads a descriptor file of the Media Object from the Download Server called the Download Descriptor (DD). The DD contains metadata about the Media Object such as the media type of the Media Object and the size of the Media Object. The DD also includes instructions to the Download Agent how to download the Media Object. The Download Descriptor can be located on either the Presentation Server or the Download Server.

3) Downloading Media Object

The Download Agent downloads the Media Object from the Download Server according to the information provided in the DD. Example of these information are the URI where to download the Media Object from and the protocol to be used, e.g. HTTP [HTTP].

4) Notification

The Download Agent may be required to post notifications to the Notification Server. The notifications include information about the outcome of the download transaction. There are several types of notifications.

- Download Completion Notification

  The download completion notification indicates to the server that the Download Agent has successfully downloaded the Media Object. However, the Media Object is not yet available to the User.

- Installation Notification

  The installation Notification indicates to the server that the Download Agent has successfully installed the Media Object, and the Media Object (to the best knowledge of the Download Agent) will be made available to the User.

- Deletion Notification

  The deletion notification indicates to the server that the Media Object has removed from the device, and the Media Object (to the best knowledge of the Download Agent) is no longer available to the User.

4.5 Service Flow in a Browsing Environment

The control flow mechanism covers the basic use case for context management where the end User selects to continue with a browsing operation after the completion of the download transaction. The mechanism can be used both in the situation where the download was completed successfully and in the situation where it was terminated with some kind of a failure or error.

If after the download has completed the end User decides to perform some other action, such as rendering the downloaded Media Object, then this is outside the scope of the OMA DLOTA version 2.0 specifications.

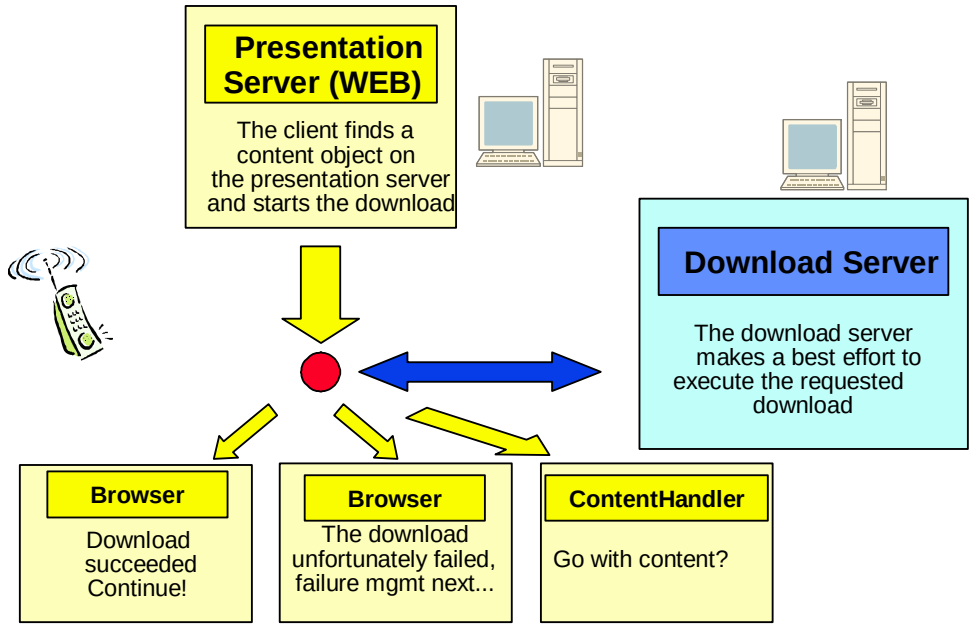

Figure 3: Basic service User experience flow control

5. DLOTAv2.0 Delivery Methods

Since DLOTA is a bearer independent protocol, the Download Descriptor and the Media Object can be delivered to the Download Agent using any underlying transport protocol such as HTTP [HTTP] and WSP [WSP]. The Download Descriptor and the Media Object can also be delivered using push protocols such as MMS [MMS], WAP Push [WAP Push] and MBMS [MBMS]. Therefore; there are four different scenarios where the Download Descriptor and the Media Object are provided to the Download Agent (Table 1).

		Media Object delivery method	
		Pull	Push
DD delivery method	Pull	Case 1: Pull-Pull Scenario	Case 2: Pull-Push Scenario
	Push	Case 3: Push-Pull Scenario	Case 4: Push-Push Scenario

Table 1: DLOTAv2.0 Delivery Methods

- Case1: Pull-Pull Scenario

  This is the most typical scenario where the Download Descriptor and the Media Object are delivered by using a pull method such as HTTP and WSP.

- Case2: Pull-Push Scenario

  The Download Descriptor is delivered by using a pull method such as HTTP and WSP, and then the Media Object is delivered by using a push method such as WAP Push and MBMS. In MBMS, Media Objects are delivered over a broadcast bearer. This scenario can be cost effective if the size of the Media Object is large and only limited bandwidth is available.

- Case3: Push-Pull Scenario

  The Download Descriptor is pushed to the Download Agent, and then the Media Object is delivered by using a pull method such as HTTP and WSP. This scenario can used for the situation where the content provider wants to notify the end User that a Media Object is available for download, User, upon reception of the Download Descriptor the User can then decide to download the Media Object using a pull method to download it.

- Case4: Push-Push Scenario

  The Download Descriptor and the Media Object are pushed to the Download Agent. This scenario is often used if the network bearer only has a unidirectional bearer. MBMS is often used over the unidirectional bearer.

In case of using push, it might be possible to enable unwanted content to be downloaded and installed into a target terminal. For example, a malicious back-end entity may push a Download Descriptor or hide a Download Descriptor behind a hidden hyper-link, thus to trigger the download of a large (and/or malicious) Media Object to the User terminal. Solutions that resolve such security issues should be incorporated into DLOTAv2.0.

6. Example Flow of Use Cases

In this section, example flows of each use case are explained. Details of the use cases and the requirements are specified in the OMA DLOTA version 2.0 requirements specification [DLREQ].

This document often uses the term “DLOTA Server” as a logical entity which encompasses the all server components (i.e. Presentation Server, Download Server, and Status Report Server) to simplify figures used in this section.

6.1 Basic Download

This use case describes the basic functionality provided in OMA DLOTA version 1.0 [DLOTAv1.0]. The User gets a Download Descriptor, downloads a Media Object, and the Status Report Server is notified when the download is complete.

Figure 4 shows an example of the download process of the basic download by using a pull method.

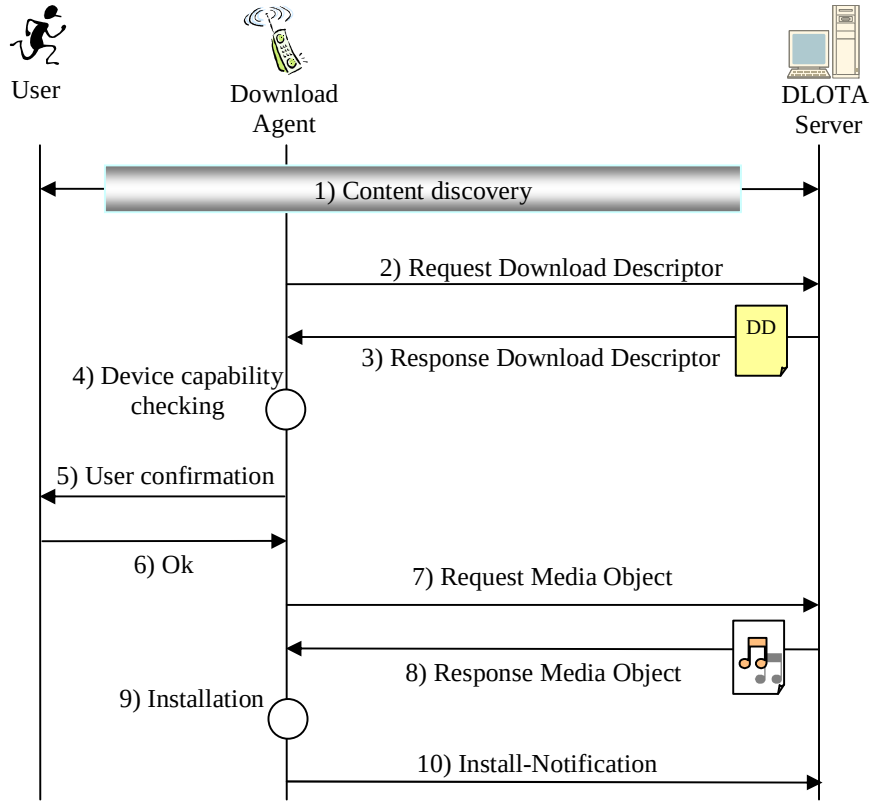

Figure 4: Basic download

1. While using Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor, is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes a URI that points to the Media Object.
4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).

5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.
6. The User decides to proceed with the download transaction.
7. The Download Agent selects the URI that points to the Media Object and the Download Agent proceeds with the download transaction.
8. The Download Agent retrieves the Media Object from the DLOTA Server.
9. The Download Agent installs the Media Object
10. The Download Agent successfully reports the status of the download transaction to the DLOTA Server and makes the content available to the User.

6.2 Basic Download without Installation Notification

This download scenario illustrates the case when an Installation Notification is not required. The advantage of this scenario is that one request-reply pair can be omitted from the process if the use case does not require application level transaction confirmation.

This scenario represents an unreliable download mechanism (similar to an HTTP retrieval of content) particularly suitable for use cases where absolute reliability is less of a priority than bandwidth optimisation.

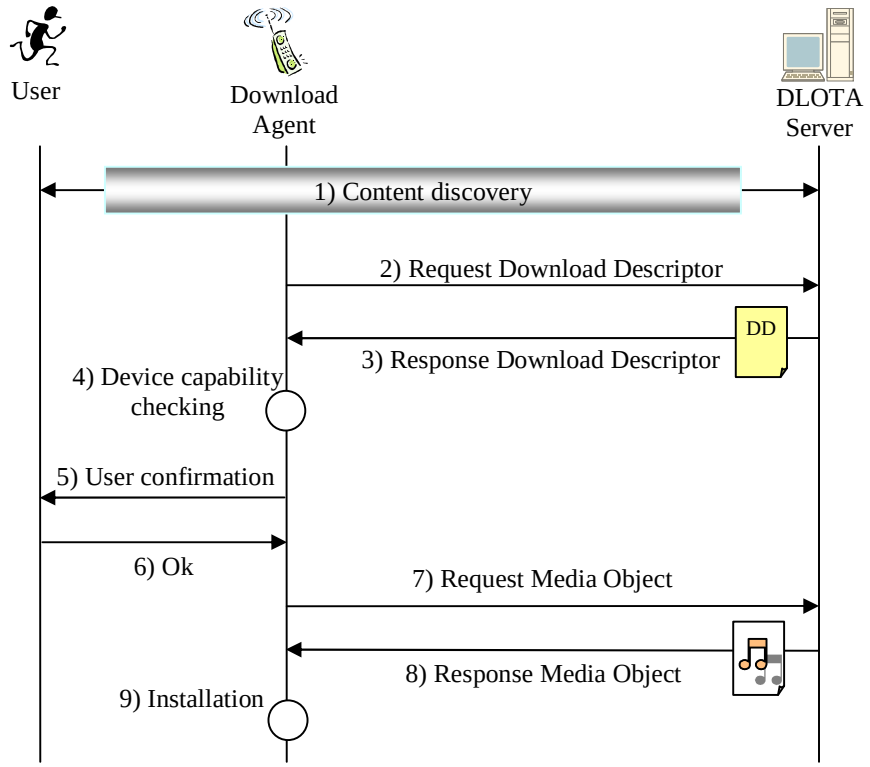

Figure 5: Basic download without installation notification

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes a URI that points to the Media Object.
4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.
6. The User decides to proceed with the download transaction.
7. The Download Agent selects the URI that points to the Media Object and the Download Agent proceeds with the download transaction.
8. The Download Agent retrieves the Media Object from the DLOTA Server.
9. The Download Agent installs the Media Object and makes it available to the User.

6.3 Combined Delivery

Combined delivery is a delivery method where a Media Object and a Download Descriptor are downloaded at the same time. As the Media Object and the Download Descriptor are delivered together, the User is unable to leverage the information in the Download Descriptor to decide whether to allow the download process to proceed or not. However, this use case supports the installation notification.

Figure 6 shows an example of the download process of the combined delivery by using a pull method.

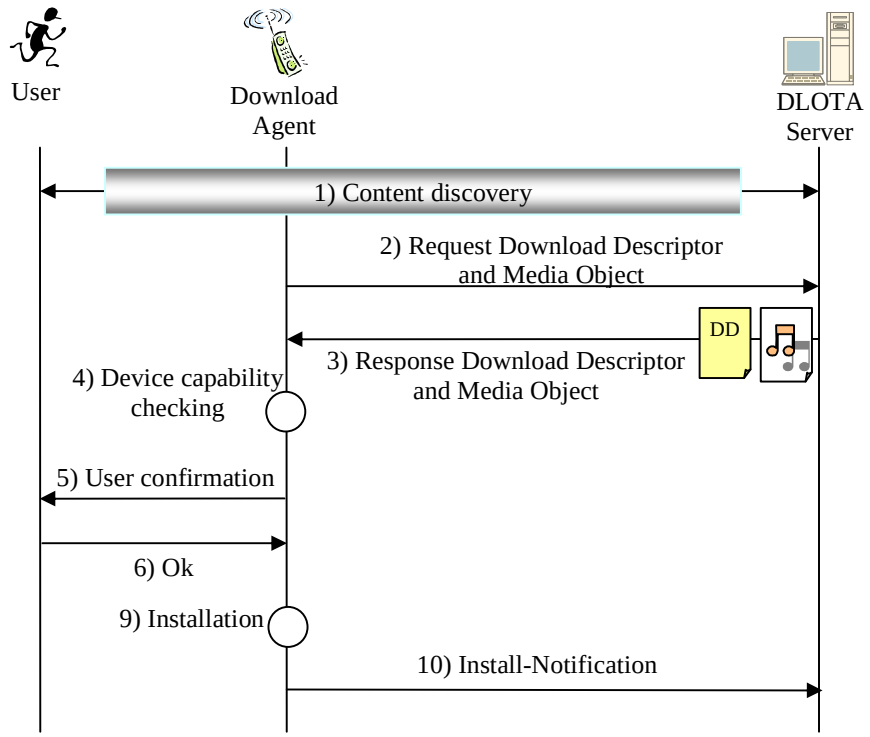

Figure 6: Basic download

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor and the Media Object. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor and the Media Object is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to a multipart entity composed of the Download Descriptor and the Media Object.
3. The multipart entity is delivered to the Download Agent.
4. The Download Agent analyses the Download Descriptor and checks the capability of the device (e.g. available memory size, content type of the Media Object, etc.).
5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the installation of the Media Object or not..
6. The User decides to install the Media Object.
7. The Download Agent installs the Media Object.
8. The Download Agent successfully reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.4 Updating Media Objects

This use case describes updating a Media Object that is already installed on the device.

Figure 7 shows an example of the process for updating Media Objects using a pull method.

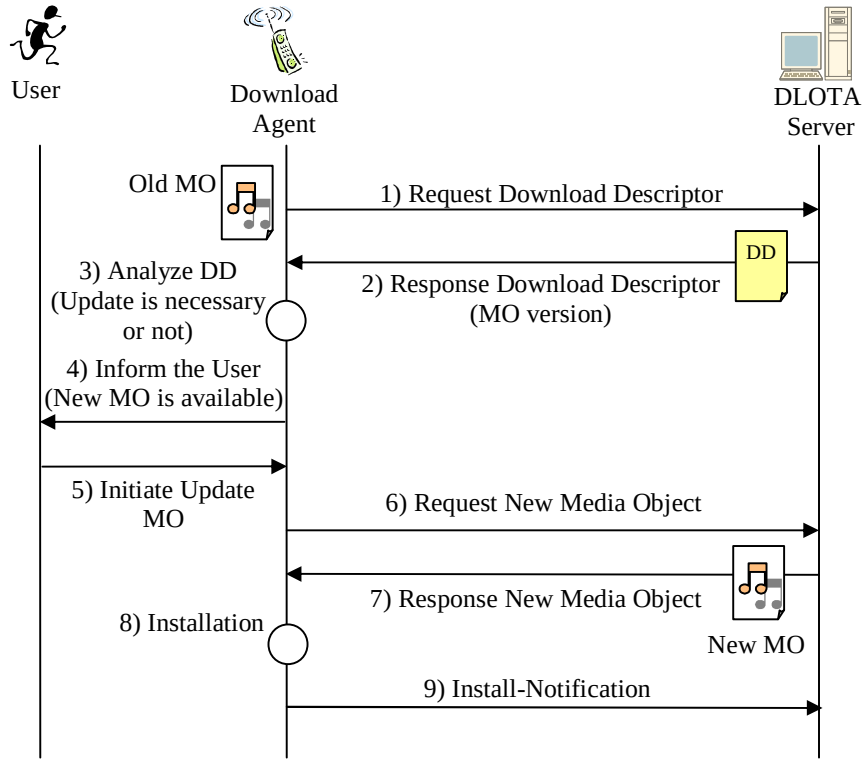

Figure 7: Updating Media Objects

1. The Download Agent downloads the Download Descriptor from the DLOTA Server. The download transaction may be triggered by the User or by the timer set within the Download Agent.
2. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes the version information of the new Media Object.
3. The Download Agent analyses the Download Descriptor and determines if a new version of the Media Object is available. The Download Agent also analyses the capability of the device (e.g. available memory size, content type of the Media Object, etc.).
4. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.
5. The User decides to proceed with the download transaction.
6. The Download Agent selects the URI that points to the Media Object and the Download Agent proceeds with the download transaction.
7. The Download Agent retrieves the Media Object from the DLOTA Server.
8. The Download Agent replaces the old Media Object with the new Media Object.
9. The Download Agent successfully reports the status of the download transaction to the DLOTA Server and makes it available to the User.

6.5 Removing Media Objects

This use case describes removing a Media Object that was previously downloaded and installed.

Figure 8 shows an example of the process of the removing Media Objects use case.

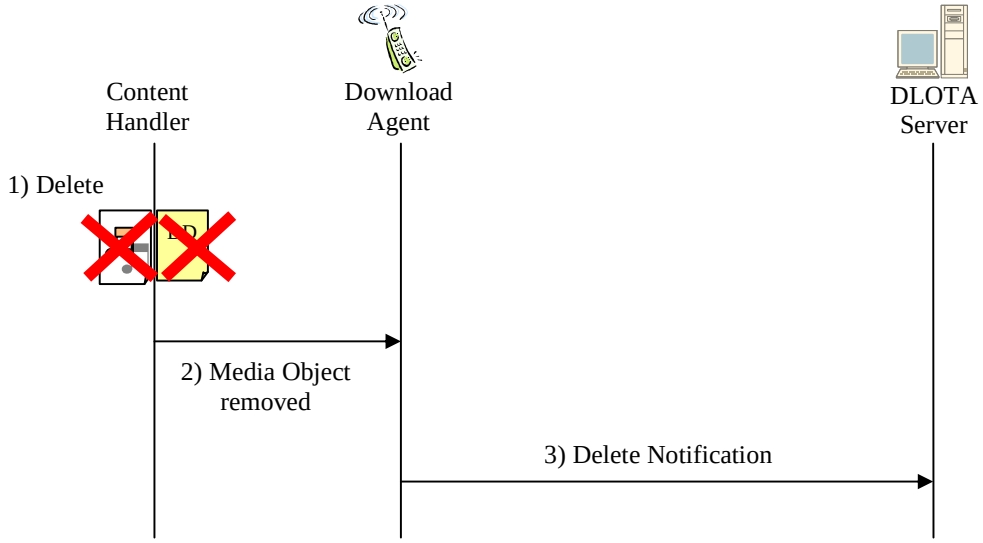

Figure 8: Removing Media Objects

1. The Content Handler removes a Media Object (e.g. by the instruction from the User).
2. The Media Object is “uninstalled” on the device (e.g. removed from the file system).
3. The Download Agent notifies the DLOTA Server that the removal was a success.

6.6 Download from Multiple Servers

The User chooses content from Content Portal. When the Download Agent downloads a Download Descriptor from Content Portal, the Download Descriptor may include multiple sources for content object(s) if multiple copies of the Media Object(s) are located on more than one DLOTA Server. The multiple sources can be used by the Download Agent to download different chunks of the Media Objects from different sources in parallel or it can be used as alternative source for the Media Object if one URI is not resolvable by the Download Agent. The Download Agent uses information in the Download Descriptor to fetch the Media Object(s) from multiple DLOTA Servers simultaneously. The Download Agent reconstructs content object(s) as soon as the Download Agent receives data from a subset of the DLOTA Servers.

Figure 9 shows an example of the download from multiple servers use case by using a pull method.

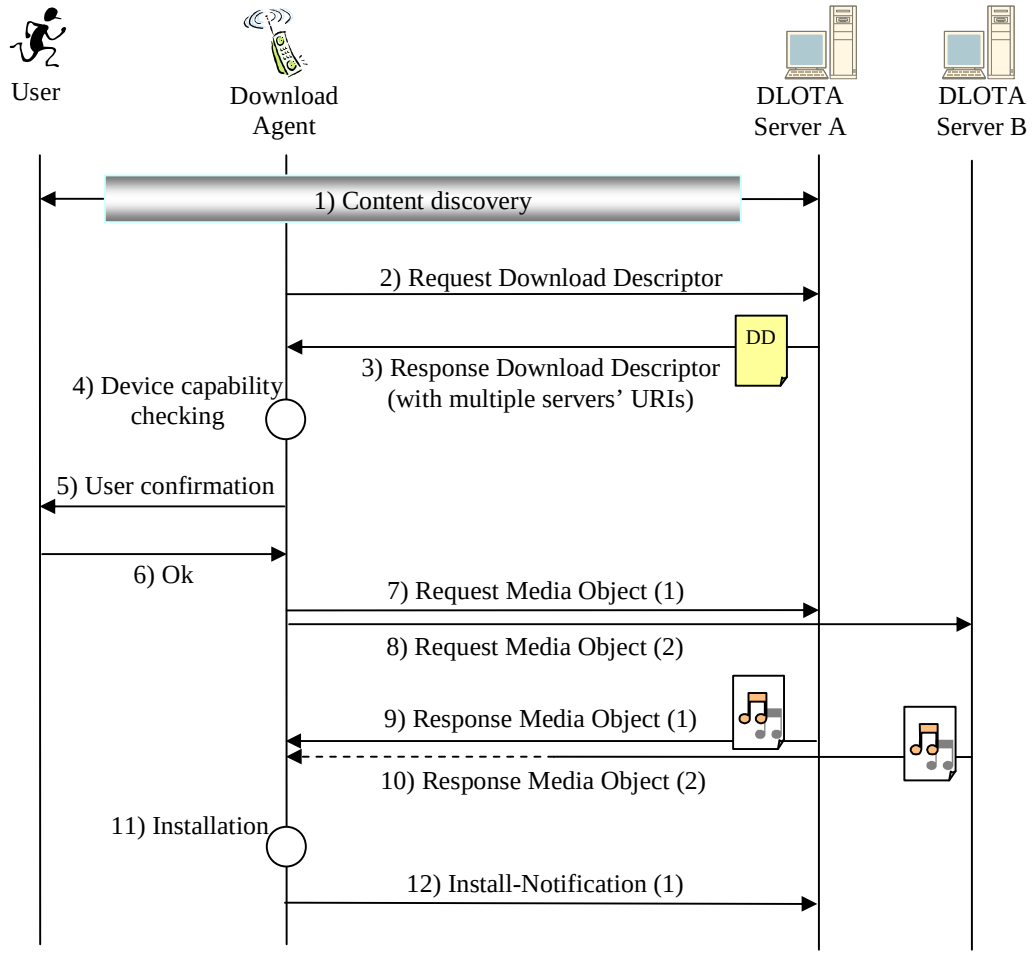

Figure 9: Download from Multiple Servers

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.

2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.

3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor may include multiple URIs for the Media Object.

4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).

5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.

6. The User decides to proceed with the download transaction.

7. and 8. The Download Agent uses information in the Download Descriptor to fetch the Media Object chunks from multiple DLOTA Servers simultaneously.

9. and 10. The Download Agent retrieves the Media Objects chunks from the DLOTA Servers. As shown by the figure, the Download Agent does not need to wait for the reception of the first Media Object chunk.

11. The Download Agent concatenates the chunks and installs the Media Object as soon as the Download Agent receives data from a DLOTA Server.

12. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.7 Download of Multiple Objects

DLOTAv2.0 provides a flexible solution that allows the download of multiple objects possibly from different sources in the same download OTA session. Content can be composed of several distinct Media Objects that may need to be downloaded from different sources. This is regarded as a “shopping-cart use case”, where a User selects many pieces of content to be downloaded.

Figure 10 shows an example of the download of multiple objects use case by using a pull method, and the Download Agent also can download and install the multiple objects simultaneously.

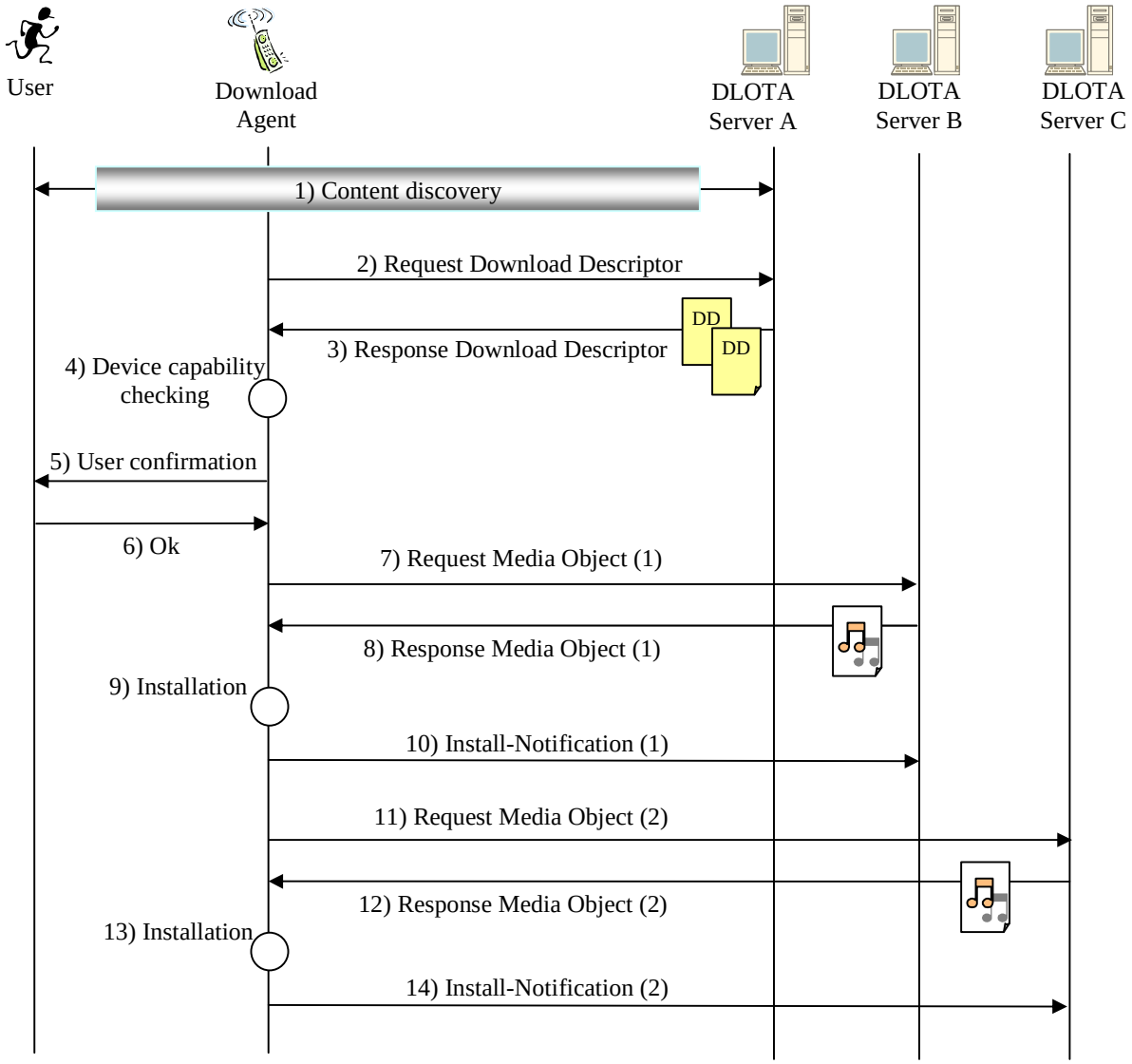

Figure 10: Download of Multiple Objects

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptors is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes several URIs, each URI points to a Media Object.
4. The Download Agent analyses the Download Descriptors and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.
6. The User decides to initiate the download transactions.

7. The Download Agent selects the URI that refers to the Media Object (1) and starts the download transaction.
8. The Download Agent retrieves the Media Object (1) from the DLOTA Server.
9. The Download Agent installs the Media Object (1).
10. The Download Agent reports the status of the download transaction of the Media Object (1) to the DLOTA Server and makes Media Object (1) available to the User.
11. The Download Agent selects the URI that refers to the Media Object (2) and starts the download transaction.
12. The Download Agent retrieves the Media Object (2) from the DLOTA Server.
13. The Download Agent installs the Media Object (2).
14. The Download Agent reports the status of the download transaction of the Media Object (2) to the DLOTA Server and makes Media Object (2) available to the User.

6.8 Download of Compound Objects

The function of download of compound objects provides flexible solution that allows the download of multiple objects possibly from different sources in the same download OTA session. Content can be composed of several distinct Media Objects that need to be downloaded from different sources.

Figure 11 shows an example of the download of compound objects use case by using a pull method.

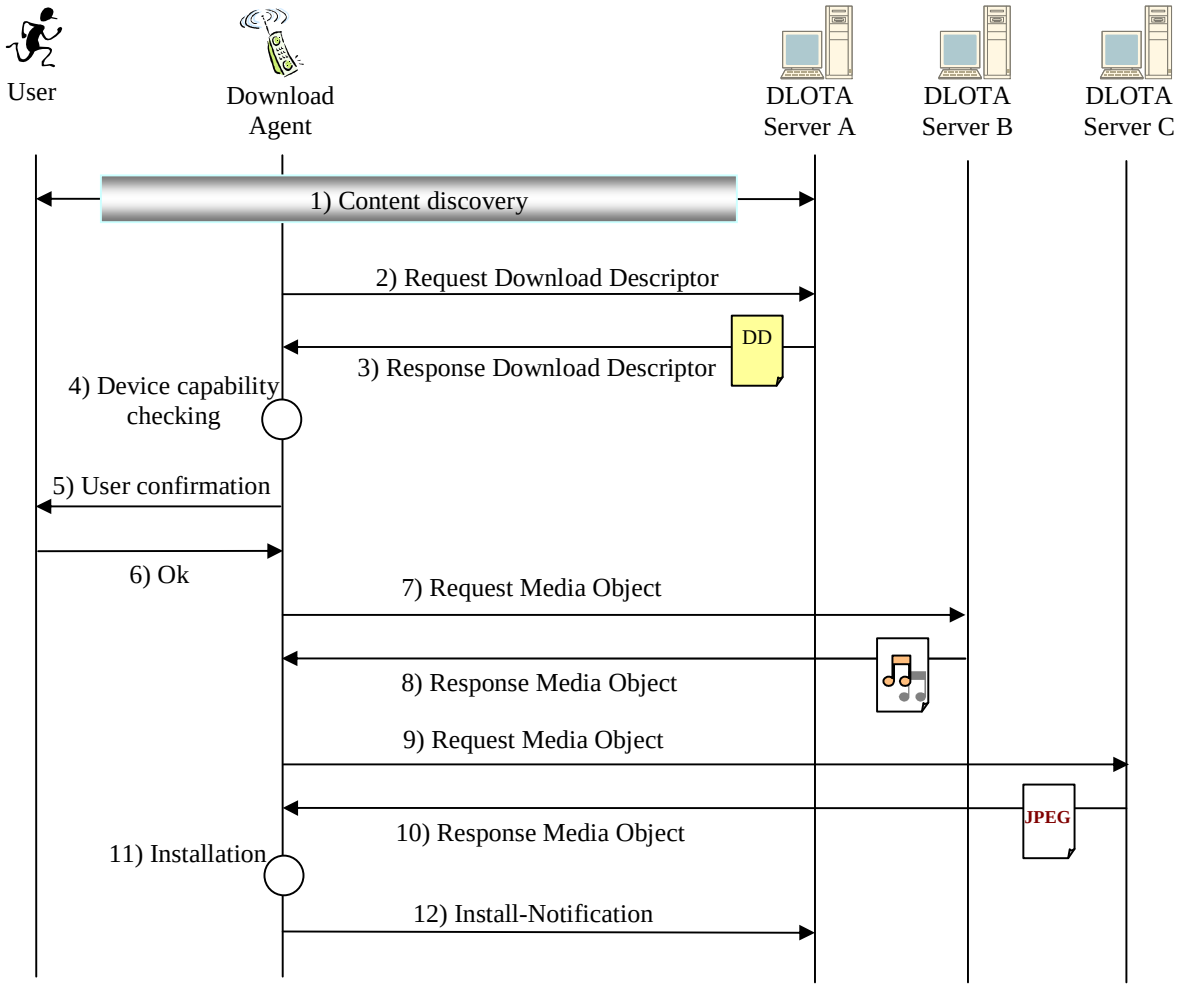

Figure 11: Download of Compound Objects

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.

2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.

3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor indicates that the Media Object is a compound object. The Download Descriptor includes several URIs. In this example, the Download Descriptor includes two URIs. One of which is a music data and the other is its JPEG image data of the CD jacket.

4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.). The Download Agent may reject the download transaction if it determines that it can not support the compound objects.

5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.

6. The User decides to proceed with the download transaction.

7. and 9. The Download Agent selects the URIs that point to Media Objects and starts the download transaction.

8. and 10. The Download Agent retrieves the compound objects from the DLOTA Servers. In this example, the Download Agent downloads the music data from the DLOTA Server B and the JPEG image from the DLOTA Server C.

11. The Download Agent installs the compound objects.

12. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the compound objects available to the User.

6.9 Download of chunked Media Objects

The download of chunked Media Objects use case provides downloading of large Media Objects (e.g. 1Mbytes～) that are divided into multiple chunks of data, in order to deal with a maximum transfer size that may be imposed by the underlying network.

Figure 12 shows an example of the download of chunked Media Objects use case by using a pull method.

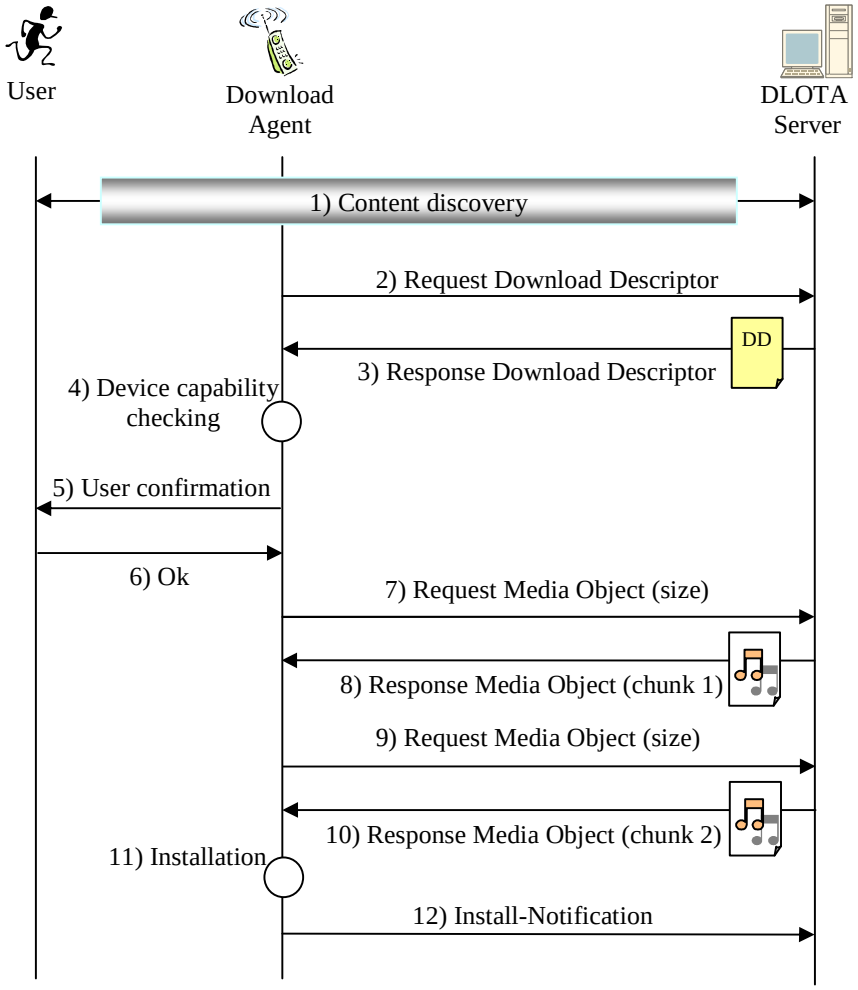

Figure 12: Download of Chunked Media Objects

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.

2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.

3. The Download Descriptor is delivered to the Download Agent.

4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).

5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.

6. The User decides to proceed with the download transaction.

7. and 9. The Download Agent decides the chunked size, and starts the download transaction issuing several requests. The Download Agent can download the chunks in sequence or in parallel depending on the capabilities of the Download Agent and DLOTA Server.

8. and 10. The Download Agent retrieves the chunked Media Object from the DLOTA Server.

11. The Download Agent concatenates the data from each chunked object request to form the Media Object and installs it.

12. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.10 Control of User Confirmation Prompt

In order to support the use cases where the download of a Media Object occurs without the need for explicit User confirmation, DLOTA 2.0 supports a mechanism to control the User confirmation prompt. Moreover, in a pull scenario, where the content is discovered and downloaded during a browsing session, the User confirmation prompt may represent an unnecessary step. In the push scenario, where the content provider pushes the download descriptor to a device, security issues must be considered. Removing the User confirmation prompt may allow malicious content providers to push malicious content to a target device, therefore a level of trust between the content provider and the User's device must exist prior the download of a Media Object where the User confirmation is disabled.

Figure 13 shows an example of the control of User confirmation prompt use case by using a pull method.

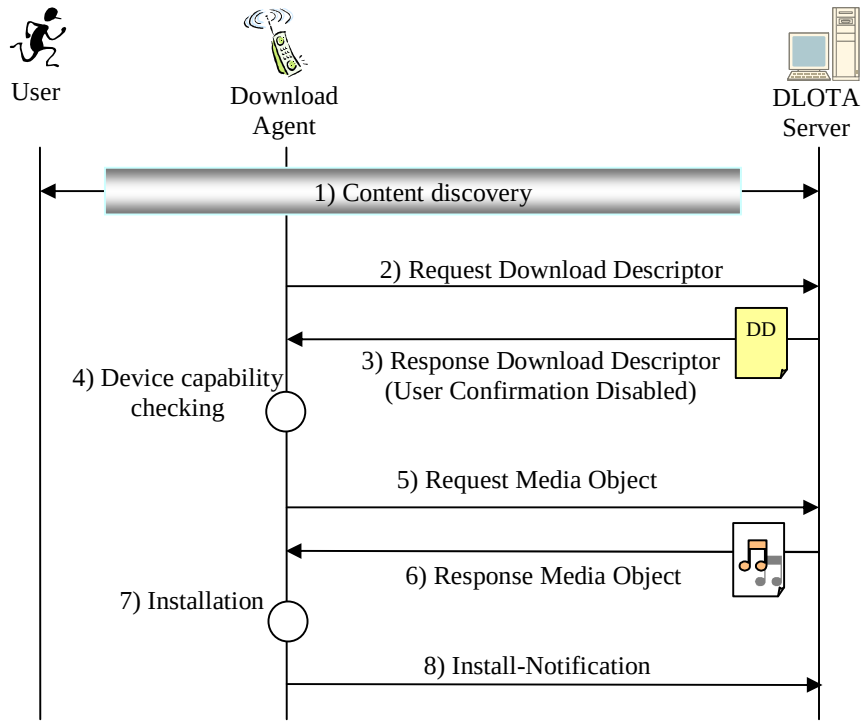

Figure 13: Control of User Confirmation Prompt

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes a flag to disable User confirmation.
4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
5. The Download Agent selects the URI that points to the Media Objects and starts the download transaction.
6. The Download Agent retrieves the Media Object from the DLOTA Server.
7. The Download Agent installs the Media Object.
8. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.11 Support for Resumable Download Session

With the Media Objects becoming larger in size, the time needed to download them increase, and this therefore increases the probability for the OTA transaction to be interrupted. Temporary lack of coverage, lack of resources within the device or some other event may occur with the result of interrupting the download. Therefore DLOTA 2.0 supports a mechanism to pause a download session and resume it at some point later in time.

As resume download would begin from the interrupted point, the downloaded data should be stored in the terminal. In order to resume a download session the Download Agent needs to calculate the size of the partially downloaded Media Object, this is the begin point for the resume., This parameter needs to be sent to the DLOTA Server when sending a resume download request. The resume download request could include a Range request element such as (X, Y), here X is the size of the partially downloaded Media Object (or the begin point) and Y is result of subtracting the size of the partially received Media Object from the total size of the Media Object. Figure 11 shows an example of resume download session.

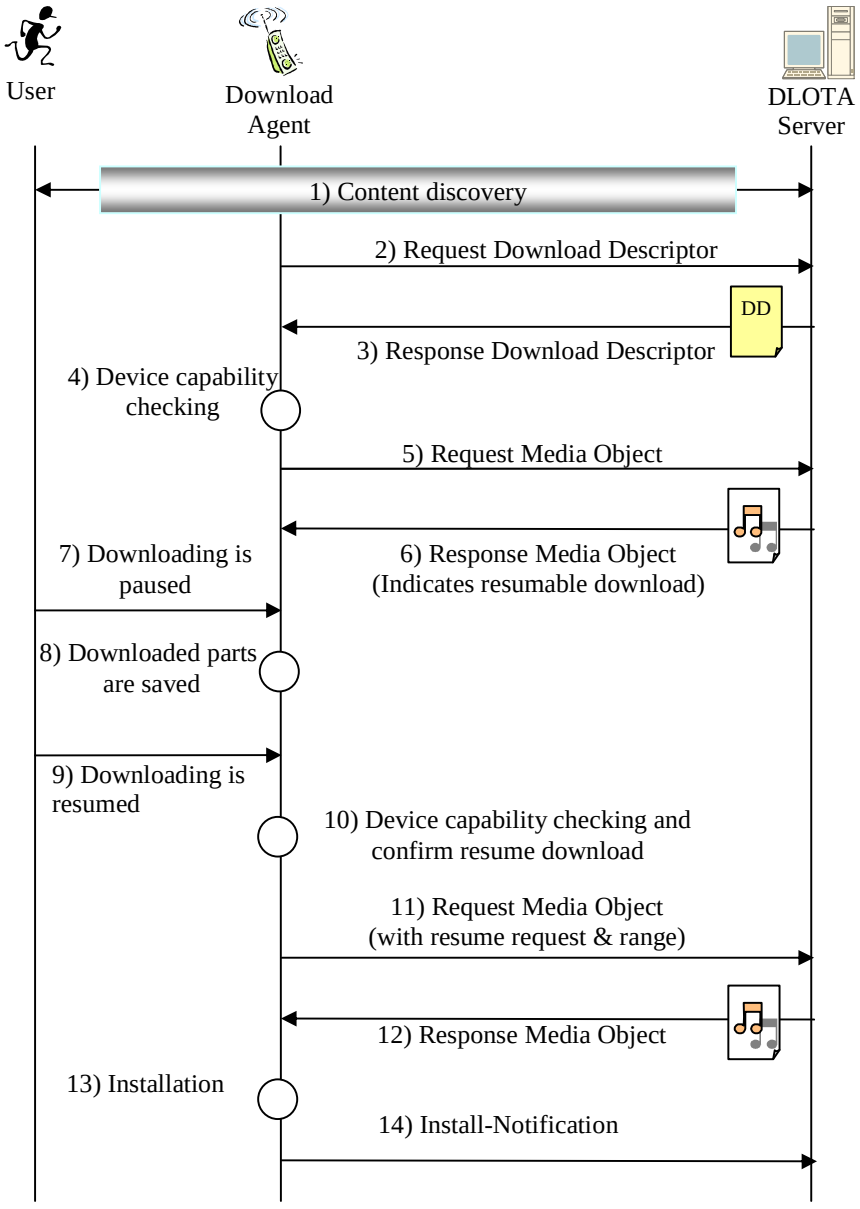

Figure 14: Resumable Download Session

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.

2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent.
4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
5. The Download Agent selects the URI that points to the Media Objects and starts the download transaction.
6. The Download Agent retrieves the Media Object from the DLOTA Server. In this step, the DLOTA Server indicates that the download transaction is resumable.
7. The download transaction is paused by the User or it is interrupted by a temporary loss of coverage or some other reason.
8. The Download Agent stores the partially downloaded Media Object and notifies the User that the download transaction was interrupted.
9. User decides to resume download the object
10. The Download Agent calculates the size of the downloaded part of the object and analyses the capability of device (e.g. available memory size) using attributes from the Download Descriptor that was downloaded at the start of the paused download transaction.
11. The Download Agent calculates the range, and attempts to resume the download transaction.
12. The Download Agent retrieves the remaining part of the Media Object from the DLOTA Server.

    Note: Several requests and responses may be needed to resume download of multiple objects, compound objects and chunked Media Objects.
13. The Download Agent reconstructs the Media Object from partially received Media Object and data downloaded as a result of the resume request and installs it.
14. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object(s) available to the User.

6.12 Pre-downloading of Media Objects

The Download Agent downloads a number of Media Objects for later use by using a network bearer that is economical and/or has an adequate bandwidth. Afterward, the User decides to consume some of the Media Objects that have already been downloaded to the device.

If requested by the DLOTA Server the Download Agent may notify the DLOTA Server that the Media Objects have been downloaded successfully. The Download Agent may also notify the DLOTA Server what Media Objects are selected by the User and whether each of the selected Media Objects is installed correctly or not.

Figure 15 shows an example of the pre-downloading of Media Objects use case by using a pull method. In this example, the Download Agent downloads three Media Objects.

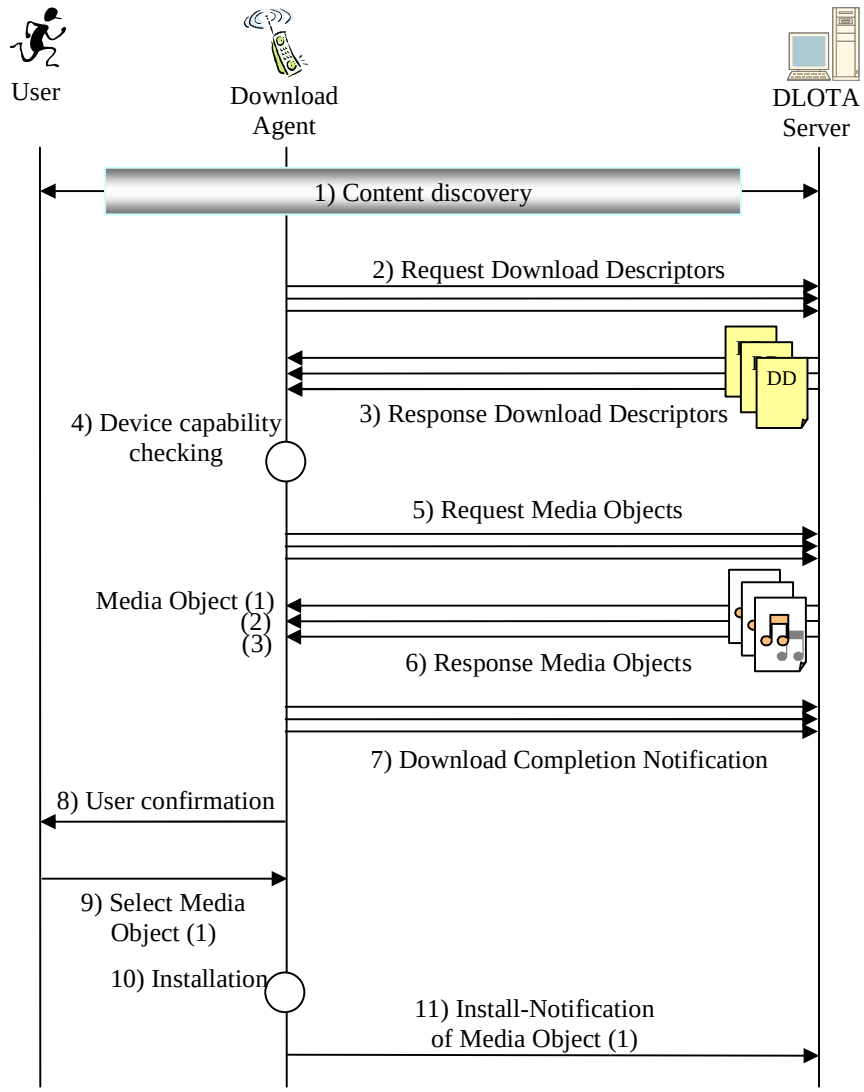

Figure 15: Pre-downloading of Media Objects

2. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and. How to find the URI to the Download Descriptor is outside the scope of DLOTA.

3. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.

4. The Download Descriptors are delivered to the Download Agent. Each of the Download Descriptor includes URIs that refers to several Media Objects.

5. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Objects, etc.).

6. The Download Agent selects the URIs that point to the Media Objects and the Download Agent starts the download transactions. The download transaction may be executed either in serial or parallel.

7. The Download Agent retrieves the Media Objects from the DLOTA Server.

8. The Download Agent may notify the DLOTA Server that the download transaction was successful. The Download Complete Notification indicates to the DLOTA Server that the Download Agent has successfully downloaded the Media Object. However, at this point the Media Object is not yet available to the User. The sending of the Download Complete Notification is optional.

9. At some point later in time the Download Agent notifies the User that Media Objects resident on the device are available for installation.

10. The User decides to consume the Media Object. In this example, the User decides to consume Media Object (1).

11. The Download Agent installs the Media Object. In this example, the Download Agent installs the Media Object (1). (For those MOs which were not selected by the user here, if the user wants to consume them in the future, the user may initiate the installation request to the Download Agent and after installing, the Download Agent should reports the status of the installation transaction to the DLOTA Server if requested by the DLOTA Server.)

12. The Download Agent reports the Installation of the Media Object to the DLOTA Server and makes the Media Object available to the User.

6.13 Download Timing Reservation

Download timing reservation provides a flexible solution that enables Users to establish and schedule reservations such that Media Objects can be automatically downloaded at the scheduled time(s).

Typically network traffic during the early morning and late at night (off-peak) is lighter than that during the day (peak) and therefore network tariffs during these periods are normally lower. Download reservation allows Users and Network Operators to use the network in both a cost and capacity effective manner.

Figure 16 shows an example of the download timing reservation use case by using a pull method.

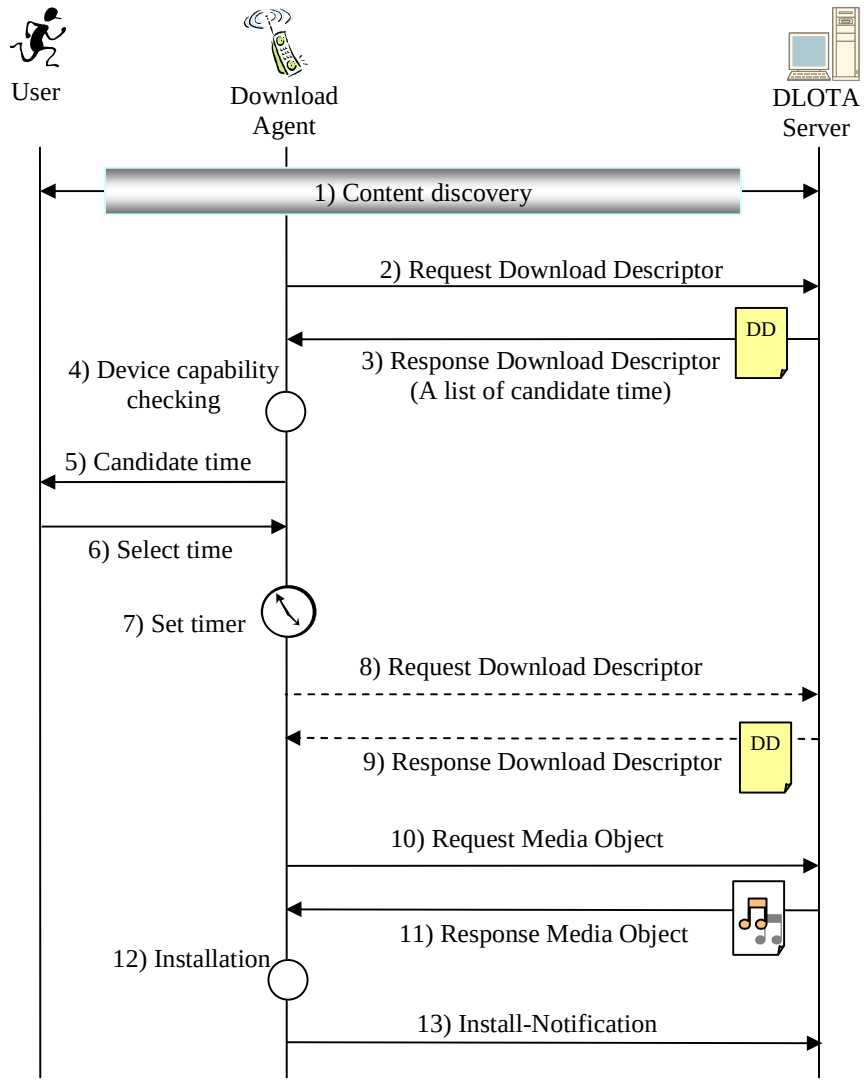

Figure 16: Download Timing Reservation

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content Discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.

2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.

3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor may contain a list of candidate times when the Media Object can be automatically downloaded.

4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).

5. If the Download Descriptor contains a list of candidate times then the Download Agent may ask the User to select the execution time when the Media Object should be automatically downloaded.

6. The User selects/agrees the time when the automatic download should take place.

7. The Download Agent sets the timer.

8. Before the Download Agent executes the download transaction at the designated time, if the Download Descriptor contains a URL that points to the updated Download Descriptor, the Download Agent fetch the Download Descriptor in order to determine if the Media Object has been replaced or updated.

9. If it is available, the updated Download Descriptor is delivered to the Download Agent.

10. The Download Agent selects the URI that points to the Media Object and starts download transaction at the specified time.

11. The Download Agent retrieves the Media Object from the DLOTA Server.

12. The Download Agent installs the Media Object.

13. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.14 Server Initiated Automatic Download

The Download Descriptor is pushed to the Download Descriptor, and then the Download Agent automatically downloads the Media Object. The use case provides the Push-Push Scenario as explain in section 5. In the Push-Push Scenario, where the DLOTA Server pushes the Download Descriptor to the Download Agent, security issues must be considered. To deal with the security issue, this use case is often used with Control of User Confirmation Prompt with the user confirmation enabled as explained in section 6.10. The other security issues such as DoS attack mush also be considered as explained in section 7.

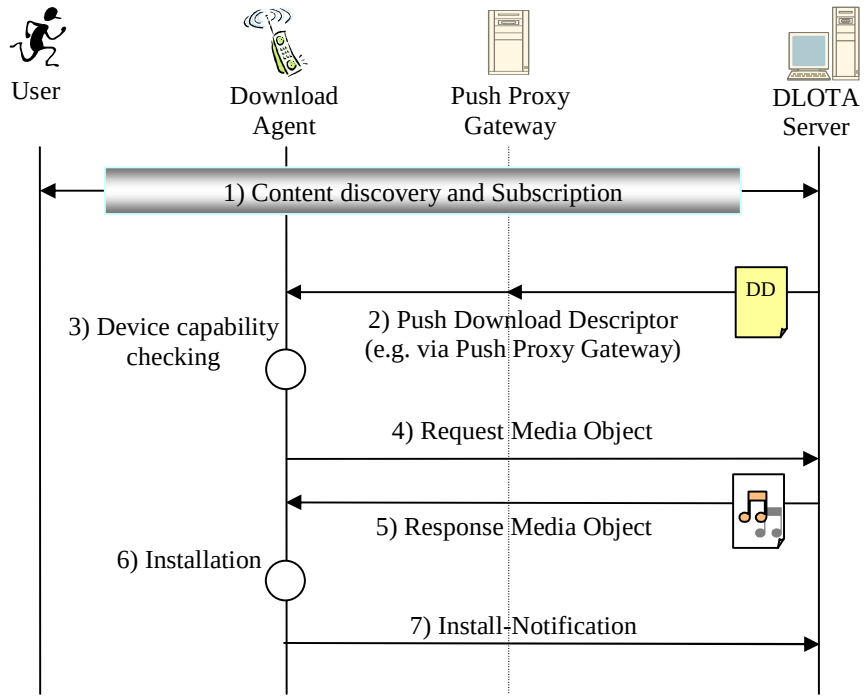

Figure 17: Server Initiated Automatic Download

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content Discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.
2. In that case, a link to the Download Descriptor is pushed to the Download Agent via Push Proxy Gateway as specified in [WAP Push].
3. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
4. The Download Agent selects the URI that points to the Media Object and starts download transaction.
5. The Download Agent retrieves the Media Object from the DLOTA Server.
6. The Download Agent installs the Media Object.
7. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

6.15 Progressive Download

In order to improve the User experience DLOTA 2.0 supports the rendering of Media Objects when they are being still downloaded. The term "progressive download" is intended to describe this kind of functionality. If progressive download is allowed and supported by the device, the Download Agent can start to render the content while the download is still in progress.

Figure 18 shows an example of the progressive download use case by using a pull method.

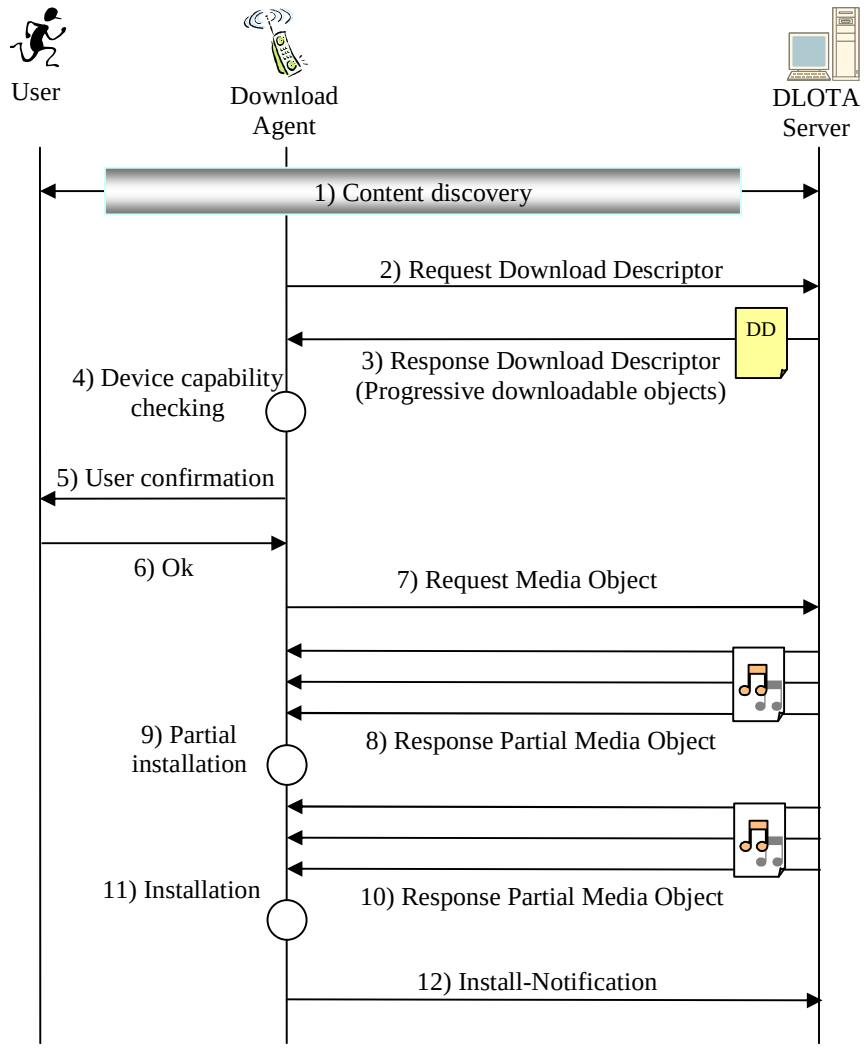

Figure 18: Progressive Download

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor includes a URI that references the Media Object. The Download Descriptor indicates that the Media Object is a continuous Media Object (i.e. that it can be progressively downloaded).
4. The Download Agent analyses the Download Descriptor and checks the capability of device (e.g. available memory size, content type of the Media Object, etc.).
5. Using the information included in the Download Descriptor, the Download Agent requests the User to confirm whether to proceed with the download transaction or not.
6. The User decides to proceed with the download transaction.

7. The Download Agent selects the URI that points to the Media Objects and starts download transaction.
8. The Download Agent retrieves a part of the Media Object from the DLOTA Server.
9. The Download Agent temporally installs the part of the Media Object to make it available for rendering.
10. The Download Agent continues to retrieve parts of the Media Object from the DLOTA Server and temporally install them to make them available for rendering.
11. The Download Agent installs the complete Media Object.
12. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the complete Media Object available to the User.

6.16 Download OTA over Broadcast Protocols

In this use case the download of a Media Object occurs via a broadcast bearer. The OMA DLOTA download descriptor (DD) is used to carry the necessary information for starting the broadcast session. This information can be carried within the DD itself or the DD can be used to download a session descriptor file. Regardless of the mechanism, the DLOTA session and the relative transaction outcome is relative to the download of the Media Object itself and not of the DD or any other session descriptor.

Figure 19 shows an example of the download OTA over broadcast protocols. In this example, the download descriptor is provided by using a pull method, and the Media Object is provided by using a push method. Note that the DLOTA Server shown in this figure also includes the logical functions of Presentation Server and Status Report Server.

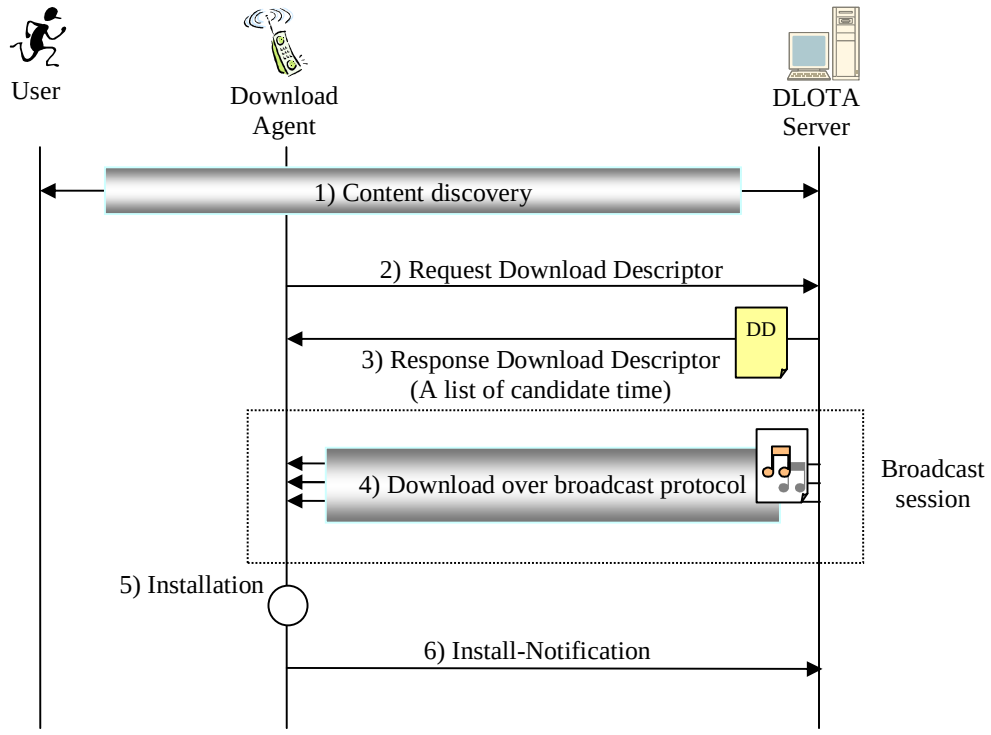

Figure 19: Download OTA over Broadcast Protocols

1. While using a Discovery Application, the User is typically presented with a reference to the Download Descriptor. The reference may be on a Web page, or inside an email or MMS message, or stored in memory or in an accessory attached to the phone. Content discovery and how to find the URI to the Download Descriptor is outside the scope of DLOTA.
2. The Download Agent selects a URI that points to the Download Descriptor in the DLOTA Server.
3. The Download Descriptor is delivered to the Download Agent. The Download Descriptor indicates that the Media Object should be downloaded via broadcast protocols.
4. The Download Agent receives the Media Object thorough a broadcast protocol such as FLUTE. How to download the Media Object through a broadcast protocol is outside the scope of DLOTA version 2.0.
5. The Download Agent installs the Media Object.
6. The Download Agent reports the status of the download transaction to the DLOTA Server and makes the Media Object available to the User.

7. Security Considerations

There are some security issues that need to be considered for DLOTAv2.0. This section summarizes high-level analysis of security threats for DLOTAv2. Detailed security solutions for DLOTAv2.0 are specified by the DLOTAv2.0 Technical Specification.

- Threats against Authentication:

  Failure to authenticate the Media Objects (MO) can result in DLOTAv2.0 clients to download potentially unauthorized and malicious MOs. Authentication of the DLOTAv2.0 client is also essential to provide secure charging for the DLOTA service providers (e.g. network operators) and access control for the Download Servers.

- Threats against Confidentiality and Integrity Protection:

  If the access network used does not provide integrity protection between the Download Agent and the Download Server, an attacker can modify the contents of an MO or a DD in transit. If authentication and integrity protection is provided by end-to-end using channel protection mechanisms like TLS then confidentiality protection can also be achieved without requiring any additional security mechanism.

- Threats against Authorization:

  Authorization allows the Download Agent to reliably know that the Download Server it is communicating with is trusted/authorized to deliver the Media Object. A secure authorization mechanism should be used by Download Agents to identify if a particular Download Server is authorized before initiating a download. If a Download Agent can not determine if a Download Server authorized then as a minimum it must request confirmation from the user in order to proceed with the download.

- DoS Threats:

  In use cases where the download service is remotely initiated by the Download Servers (i.e. server initiated automatic download), mechanisms must be employed to secure the trigger in order to protect and mitigate against Denial of Service (DoS) attacks against Download Servers. If the triggers are not secured, a malicious party can transmit a large number of triggers to a large set of DLOTAv2.0 terminals to start a download session at the same time.

Appendix A. Change History

A.1 Approved Version History

Reference	Date	Description
OMA-AD-DLOTA-V2_0-20110329-A	29 Mar 2011	Status changed to Approved by TP: OMA-TP-2011-0100-INP_Download_OTA_V2_0_ERP_for_Final_Approval