Device Management OBEX Binding

Candidate Version 1.3 – 25 May 2010

Open Mobile Alliance

OMA-TS-DM_OBEXBinding-V1_3-20100525-C

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.

Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document. [OMA-Template-Spec-20050101-I]

Contents

.

Figures

.

Tables

.

1. Scope

This document describes OBEX Binding for carrying DM Messages based on DM representation [DMREPRO]. The document uses the primitives and methods defined in the OBEX specification V1.2 as defined in [OBEX]

2. References

2.1 Normative References

[BTAN]	"Bluetooth Assigned Numbers", Bluetooth SIG, URL: http://www.bluetoothsig.org/assigned-numbers/
[BTGOEP]	"Bluetooth V1.1 Profile Specifications" – PartK:10 Generic Object Exchange Profile, Bluetooth SIG, URL: http://www.bluetooth.org/foundry/specification/document/Bluetooth_11_Profiles_Book/en/1/Bluetooth_11_Profiles_Book.pdf
[BTSDP]	"Bluetooth V1.1 Core Specifications" - PartE: Service Discovery Protocol, Bluetooth SIG, URL: http://www.bluetooth.org/foundry/specification/document/Bluetooth_V1.1_Core_Specifications/en/1/Bluetooth_V1.1_Core_Specifications.pdf
[BTSEP]	"Bluetooth V1.1 Profile Specifications" – PartK:5 Serial Port Profile, Bluetooth SIG, URL: http://www.bluetooth.org/foundry/specification/document/Bluetooth_11_Profiles_Book/en/1/Bluetooth_11_Profiles_Book.pdf
[DMNOTI]	"DM Notification", Open Mobile Alliance^{TM} , OMA-TS-DM_Notification-V1_3, URL:http://www.openmobilealliance.org/
[DMREPRO]	"DM Representation Protocol", Open Mobile Alliance^{TM}, OMA-TS-DM_RepPro-V1_3, URL:http://www.openmobilealliance.org/
[IOPPROC]	"OMA Interoperability Policy and Process", Version 1.9, Open Mobile Alliance™, OMA-IOP-Process-V1_9, URL:http://www.openmobilealliance.org/
[OBEX]	"IrDA Object Exchange Protocol (IrOBEX) Version 1.2", Infrared Data Association, URL: http://www.irda.org/standards/pubs/OBEX1p2_Plus.zip
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt
[RFC2396]	"Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee, et al., August 1998, URL:http://www.ietf.org/rfc/rfc2396.txt
[SAN]	"DM Server Alerted Notification", Open Mobile Alliance^{TM} , OMA-TS-DM_Notification-V1_3, URL:http://www.openmobilealliance.org/

2.2 Informative References

None.

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.

Any reference to components of the DM DTD or XML snippets is specified in this `typeface`.

3.2 Definitions

OBEX	Objext Exchange Protocol that is defined in [OBEX] .
Originator	The network device that creates a SyncML request.
Recipient	The network device that receives a SyncML request, processes the request and sends any resultant SyncML response.

3.3 Abbreviations

OMA	Open Mobile Alliance
URI	Uniform Resource Identifier [RFC2396]
URL	Uniform Resource Locator [RFC2396]
WAP	Wireless Application Protocol
XML	Extensible Markup Language

4. Introduction

This document describes the OBEX Binding for carrying DM Messages based on DM representation [DMREPRO]. The document uses the primitives and methods defined in the OBEX specification V1.2 [OBEX] .

5. OBEX Introduction

OBEX [OBEX] is a protocol for exchanging objects. It was initially designed for infrared, but it has been adopted by Bluetooth, and is also used over RS232, USB and WAP.

OBEX is a session-oriented protocol, which allows multiple request/response exchanges in one session. An OBEX session is initiated by an OBEX CONNECT request, and is established when the other device returns a success response. The connection is terminated by sending a DISCONNECT request.

In this specification, the DM Client can work either as an OBEX client or as an OBEX server at the OBEX protocol layer. In consequence, the DM Server can work either as an OBEX client or as an OBEX server. The OBEX role depends on the fact which one, the DM Client or the DM Server, initiates the DM session. DM Client

When a session has been established, the data is transferred using the PUT request. The remote device acknowledges the data, by sending a response with a status code.

DM requires that an OBEX connection is established. Connectionless OBEX cannot be used with DM.

5.1 OBEX Over IrDA

The diagram below demonstrates the position of OBEX within the IrDA stack.

e.g. Phone	**e.g. PC**
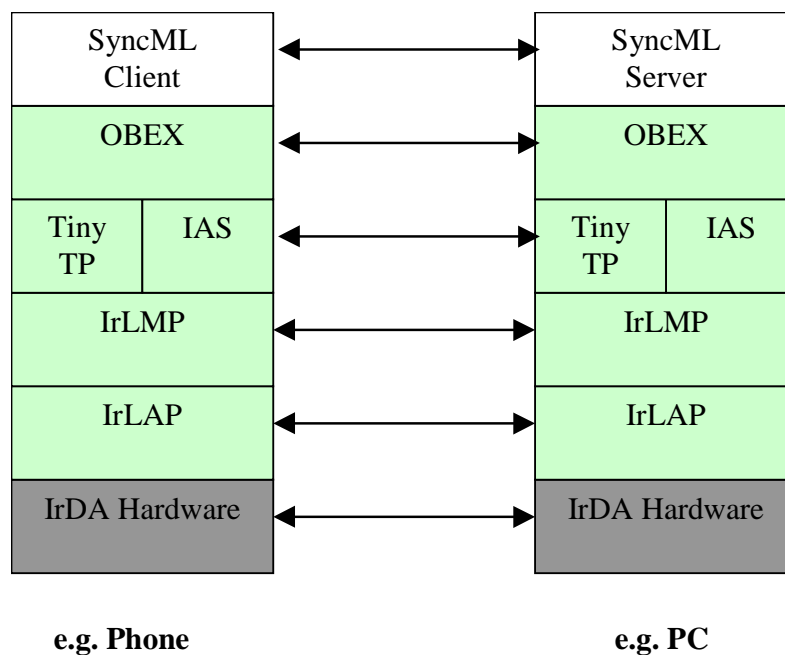
IrLAP is the link level protocol.

IrLMP is a multiplexing layer.

Tiny TP provides flow control.

IAS is the Information Access Service.

OBEX includes both a session level protocol and an application framework.

5.1.1   IAS Entry

To enable an OBEX connection over IrDA, the OBEX protocol stack needs to provide IAS setting information to the IAS protocol stack. The DM Server and DM Client SHOULD use the following IAS entry settings for DM communication via OBEX over IrDA.

5.1.1.1   IAS Entry Settings for DM Server

DM Server SHOULD use the following IAS entry settings.

Class		OBEX:DM-Server
Attribute	Name	IrDA:TinyTP:LsapSel
	Type	Integer
	Description	IrLMP LSAP selector for DM over IrOBEX, legal values from 0x01 to 0x6F

5.1.1.2   IAS Entry Settings for DM Client

DM Client SHOULD use the following IAS entry settings.

Class		OBEX:DM-Client
Attribute	Name	IrDA:TinyTP:LsapSel
	Type	Integer
	Description	IrLMP LSAP selector for DM over IrOBEX, legal values from 0x01 to 0x6F

5.2   OBEX Over Bluetooth

The Bluetooth section is specified so that the DM Client MUST be able to function as either an OBEX client, or an OBEX server, or both. The DM Server MUST be able to function as both the OBEX server and client.

The figure below shows the protocols when DM and OBEX are run over the Bluetooth protocol stack.
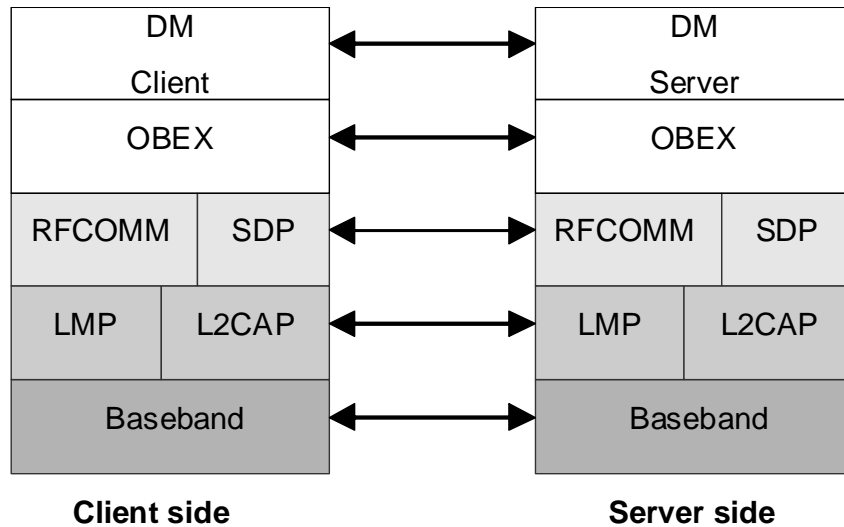
Figure 1: OBEX over Bluetooth

The Baseband, LMP, and L2CAP are the OSI layer 1 and 2 Bluetooth protocols. RFCOMM is the Bluetooth adaptation of GSM TS 07.10. SDP is the Bluetooth Service Discovery Protocol [BTSDP].

The DM Client layer shown in Figure 1 is the entity providing the sync client agent functionality. The DM Server is the SW providing the sync engine functionality.

In this specification, the DM Client can work either as an OBEX client or as an OBEX server at the OBEX protocol layer. In consequence, the DM Server can work either as an OBEX client or as an OBEX server. The OBEX role depends on the fact which one, the DM Client or the DM Server, initiates the DM session.

5.2.1   Bluetooth Service Discovery

To enable the OBEX connection over the Bluetooth protocol stack, the DM Server MUST advertise and the DM Client SHOULD advertise service records, which can be retrieved by a connecting device using the Bluetooth SDP [BTSDP].

In the case of the DM Server, the following information, i.e., service records MUST be put into the SDDB (Service Discovery DataBase).

Item	Definition:	Type/ Size:	Value:	AttrID:	Status:	Default Value:
Service Class ID List			N/A	0x0001**	MUST	
Service Class #0	DMServer	UUID	*	N/A	MUST	
Protocol Descriptor list			N/A	0x0004**	MUST	
Protocol ID #0	L2CAP	UUID	0x0100**	N/A	MUST	
Protocol ID #1	RFCOMM	UUID	0x0003**	N/A	MUST	
Param #0	CHANNEL	Uint8	Varies	N/A	MUST	
Protocol ID #2	OBEX	UUID	0x0008**	N/A	MUST	
Service name	Displayable Text name	String	Varies	0x0000+b***	MAY	"DM Server"

Table 1: DM Server Service Records

* The value 00000001-0000-1000-8000-0002EE000002 SHOULD be used in this place.

** The value or the attribute ID is specified in the Bluetooth Assigned Numbers specification [BTAN].

*** 'b' in this table represents a base offset as given by the LanguageBaseAttributeIDList attribute. For the principal language b MUST be equal to 0x0100 as described in the Bluetooth SDP specification [BTSDP].

The service records, which the DM Client SHOULD put into its SDDB, are listed below.

Item	Definition:	Type/ Size:	Value:	AttrID:	Status:	Default Value:
Service Class ID List			N/A	0x0001**	MUST	
Service Class #0	DMClient	UUID	*	N/A	MUST	
Protocol Descriptor list			N/A	0x0004**	MUST	
Protocol ID #0	L2CAP	UUID	0x0100**	N/A	MUST	
Protocol ID #1	RFCOMM	UUID	0x0003**	N/A	MUST	
Param #0	CHANNEL	Uint8	Varies	N/A	MUST	
Protocol ID #2	OBEX	UUID	0x0008**	N/A	MUST	
Service name	Displayable Text name	String	Varies	0x0000+b***	MAY	"DM Client"

Table 2: DM Client Service Records

* The value 00000002-0000-1000-8000-0002EE000002 SHOULD be used in this place.

** The value or the attribute ID is specified in the Bluetooth Assigned Numbers specification [BTAN].

*** 'b' in this table represents a base offset as given by the LanguageBaseAttributeIDList attribute. For the principal language b MUST be equal to 0x0100 as described in the Bluetooth SDP specification [BTSDP].

5.2.1.1 SDP Protocol Data Units

Table 3 shows the specified SDP PDUs (Protocol Data Units), which areREQUIRED.

PDU no.	SDP PDU	Ability to Send		Ability to Retrieve	
		DM Client	**DM Server**	**DM Client**	**DM Server**
1	SdpErrorResponse	MUST*	MUST	MUST**	MUST
2	SdpServiceSearchAttribute-Request	MUST**	MUST	MUST*	MUST
3	SdpServiceSearchAttribute-Response	MUST*	MUST	MUST**	MUST

Table 3: SDP PDUs

* This is only applicable if the DM Client is able to function as the OBEX server.

** This is only applicable if the DM Client is able to function as the OBEX client.

5.2.2 Other Bluetooth Protocol Requirements

This specification partially requires compliance to the Bluetooth Serial Port (SeP) Profile [BTSEP] if Bluetooth is used as a physical medium for OBEX. These are:

- The compliance is REQUIRED to the RFCOMM requirements as defined in Chapter 4 in the SeP Profile.

- The compliance is REQUIRED to the L2CAP requirements as defined in Chapter 5 in the SeP Profile.

- The compliance is REQUIRED to the LM protocol requirements as defined in Chapter 7 in the SeP Profile.

The SDP requirements are defined by this specification and thus, any of the requirements defined in the SeP profile (Chapter 6 in the SeP profile) does not apply to this specification. The DM Server MUST comply with both the Device 'A' and Device 'B' requirements as defined in the SeP Profile. The DM Client MUST comply with either the Device 'A' requirements, or with the Device 'B' requirements, or both as defined in the SeP Profile.

The Bluetooth LC (Link Controller) capabilities and The Bluetooth Generic Access Profile (GAP) requirements for this specification are defined in Chapter 6.5 and Chapter 7 of the Bluetooth GOEP [BTGOEP], respectively. The DM Server MUST comply with both the client and server requirements as defined in Chapter 6.5 and Chapter 7 in the GOEP. The DM Client MUST comply with either the client requirements, or the server requirements, or both as defined in Chapter 6.5 and Chapter 7 in the GOEP.

6. OBEX Mapping to DM Message

The following sections define the requirements for the binding of DM Message to OBEX.

In client initiated sync, the DM Client initiates the OBEX link, so it is also the OBEX client. The DM Client can disconnect the OBEX link when it has received the last DM Message from the DM Server.

With server alerted sync, the DM Server initiates the OBEX link, so it is the OBEX client. The DM Server cannot disconnect the OBEX link before it has received the SyncML response message for the last DM Message.

6.1   OBEX Operations

The OBEX layer SHOULD be disconnected using the OBEX DISCONNECT operation.   The OBEX specification also allows the link to be disconnected by disconnecting the underlying transport layer.

The OBEX connection can be authenticated as part of the OBEX CONNECT request/response messages, using the authenticate challenge and response headers

The OBEX client MAY send an OBEX ABORT request, to terminate a multi-packet operation (such as PUT) before it would normally end.

The following OBEX operations are REQUIRED for Client Initiated sessions.

OBEX Operation	DM Server / OBEX Server	DM Client / OBEX Client
Connect	MUST	MUST
Disconnect	MUST	MUST
Put	MUST	MUST
Get	MUST	MUST
Abort	MUST	MAY

The following OBEX operations are REQUIRED for Server Initiated sessions.

OBEX Operation	DM Server / OBEX Client	DM Client / OBEX Server
Connect	MUST	MUST
Disconnect	MUST	MUST
Put	MUST	MUST
Get	MUST	MUST
Abort	MAY	MUST

6.2   OBEX Connection Overview

The OBEX connection is made at the start of the DM session, and remains open until the DM session has completed.

The following example shows the creation of an OBEX connection, the mapping of PUT and GET requests to the DM Message transfers, and the OBEX disconnection.

This example is not intended to show a complete a DM Session but merely illustrates the use of PUT and GET within a DM OBEX binding implementation.

OBEX Client	OBEX Server	Message Direction
CONNECT Request		
	Success Response	
PUT Request		DM Message from
	Continue Response	OBEX Client to OBEX Server
PUT Request …		
	Continue Response …	
PUT Final Request		
	Success Response	
GET Request		DM Message from
	Continue Response	OBEX Server to OBEX Client
GET Request …		
	Continue Response …	
GET Final Request		
	Success Response	
DISCONNECT Request		
	Success Response	

6.2.1   Multiple Messages Per Package

Each DM Message MUST be transferred as a DM MIME media type within the body of the OBEX request or response. However in order to transfer the message the OBEX / transport layer MAY split the message into many PUT requests, followed by a PUT Final Request. When there are multiple DM Messages per DM package to transfer, each message is transferred in a separate 'set' of PUT/GET commands; depending on whether it is a DM request or response.

The recipient of a DM Message can determine if there are more DM Messages in the package by the absence of the Final element in the last received DM Message. When the recipient receives a DM Message with the Final element, it is the final message within that SyncML package.

Similarly if the PUT is not a PUT final then the recipient knows it is not the final part of the DM Message, or if the response to the GET Final Request is not an OK/success then there is more data still to transfer.

6.2.2   Type header requirement

DM Clients and DM Servers MUST support this header with either the `"application/vnd.syncml.dm+xml"` or `"application/vnd.syncml.dm+wbxml"` MIME media type values.

In the case of DM Notification, the Type header of the OBEX packet transferring the notification message MUST include the MIME media type, `"application/vnd.syncml.dm.notification"`.

6.3   OBEX Connection Establishment

The OBEX connection is established by the DM Client or DM Server generating a Connect Request, and the remote device indicates that the connection has been established, by returning a Connect Response. For each DM Session, a separate OBEX connection MUST be established.

The OBEX CONNECT request packet contains the following fields and headers:

Field/ Header	Name	Value	M/O	Explanation
Field	Opcode for CONNECT	0x80	M	
Field	Packet Length	Varies	M	
Field	OBEX Version Number	0x10	M	In [OBEX] it is stated that current version is 1.0
Field	Flags	Varies	M	
Field	Max OBEX Packet Length	Varies	M	
Header	Target	Varies	M	The UUID to be used in device management is SYNCML-DM.

The OBEX CONNECT response packet contains the following fields and headers:

Field/ Header	Name	Value	M/O	Explanation
Field	Response code for CONNECT request	0x0A	M	0xA0 for success, otherwise fail
Field	Packet Length	Varies	M	
Field	OBEX Version Number	0x10	M	In [OBEX] it is stated that current version is 1.0
Field	Flags	Varies	M	
Field	Max OBEX Packet Length	Varies	M	
Header	Connection ID	Varies	M	Connection ID is set by the Server during the OBEX Connect operation as a shorthand way for the client to direct the requests. **This MUST be the first header.**
Header	Who	Varies	M	The UUID returned is the same UUID that was sent in the connect request target header

6.4 Exchanging DM Messages over the OBEX Connection

Once an OBEX connection has been established, DM Messages can be transferred over the link.

The OBEX PUT request packet contains the following fields and headers:

Field/ Header	Name	Value	M/O	Explanation
Field	Opcode for PUT	0x02 or 0x82	M	0x02 is used for packets previous to the last put packet. 0x82 (which is 0x02 with the high bit set) is used for the last put packet.
Field	Packet Length	Varies	M	
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This MUST be the first header. Sent only once in first packet in the request sequence.
Header	Type	Varies	M	The MIME type of the object. This MUST contain the DM MIME type declaration. Sent only once in first packet in the request sequence, MUST precede object Body headers.
Header	Length	Varies	O	Length of the object. This header is OPTIONAL but highlyRECOMMENDED. Sent only once in the request sequence.

Field/Header	Name	Value	M/O	Explanation
Header	Body/End of Body	Varies	M	End of Body identifies the last chunk of the object body. End of Body header (PUT final packet) SHOULD be sent with an empty body.

The OBEX PUT response packet contains the following fields and headers:

Field/Header	Name	Value	M/O	Explanation
Field	Response code for PUT	0x90, 0xAO, 0xCD, 0xCF, …	M	0x90 for continue 0xA0 for success 0xCD if the object is too large 0xCF if the object type is not supported
Field	Packet Length	Varies	M	

Other headers, which can be optionally used, are found in [OBEX]

The OBEX GET request packet contains the following fields and headers:

Field/Header	Name	Value	M/O	Explanation
Field	Opcode for GET	0x03 or 0x83	M	0x03 is used for packets previous to the last packet containing headers. 0x83 (which is 0x03 with the final bit set) is used for the last packet containing headers and for subsequent get packets in the current Get request sequence without headers. No headers can be sent in Get request packets once the 0c83 has been sent in a previous packet.
Field	Packet Length	Varies	M	
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This MUST be the first header. Sent only once in first packet in the request sequence.
Header	Type	0x42, …	M	The MIME type of the object. This MUST contain the DM MIME type declaration. Sent only once in first packet in the request sequence.

The OBEX GET response packet contains the following fields and headers:

Field/Header	Name	Value	M/O	Explanation
Field	Response code for GET	0x90, 0xAO, 0xC0, 0xC3, …	M	0x90 for continue 0xA0 for success 0xC0 bad request 0xC3 forbidden
Field	Packet Length	Varies	M	
Header	Length	0xC3, …	O	Length of the object. This header is OPTIONAL but highlyRECOMMENDED. Sent only once in the request sequence.
Header	Body/End of Body	0x48/0x49, …	M	End of Body identifies the last chunk of the object body.

Other headers, which can be optionally used, are found in [OBEX] .

6.5   OBEX Disconnection

The OBEX connection is disconnected by the DM Client or DM Server, generating a Disconnect Request, and the remote device indicates that the connection has been terminated, by returning a success Response.

The OBEX DISCONNECT request packet contains the following fields and headers:

Field/Header	Name	Value	M/O	Explanation
Field	Opcode for DISCONNECT	0x81	M	
Field	Packet Length	Varies	M	
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This MUST be the first header.

Other headers (such as Description) which can be optionally used are found in [OBEX] .

The OBEX DISCONNECT response packet contains the following fields and headers:

Field/Header	Name	Value	M/O	Explanation
Field	Response code for DISCONNECT	0xAO	M	0xA0 for success, otherwise fail
Field	Packet Length	Varies	M	

6.6   OBEX ABORT

The client can send an OBEX abort request to terminate a multi-packet operation (such as PUT) before it would normally end. The ABORT request and response always fit in one OBEX packet, and they always have the Final bit set.

The OBEX ABORT request packet contains the following fields and headers:

Field/ Header	Name	Value	M/O	Explanation
Field	Opcode for ABORT	0xFF	M	
Field	Packet Length	Varies	M	
Header	Connection ID	Varies	O	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This MUST be the first header if it is included.

Other headers (such as Description) which can be optionally used are found in [OBEX] .

The OBEX ABORT response packet contains the following fields and headers.

Field/ Header	Name	Value	M/O	Explanation
Field	Response code for ABORT	0xAO	M	0xA0 for success, otherwise fail and the client SHOULD disconnect the OBEX connection.
Field	Packet Length	Varies	M	

Other headers (such as Description) which can be optionally used are found in [OBEX] .

6.7   DM Notification

The OBEX connection is established by the DM Server generating a CONNECT Request. The OBEX CONNECT request MUST have as the Target header a UUID of SYNCML-DM for Device Management packages. Once an OBEX connection has been established the DM Notification message data can be transferred over the link.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.

Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.   [OMA-Template-Spec-20050101-I]

7. Examples

The following examples are formatted as the examples in [OBEX] .

7.1   OBEX Connect Example

Client Request:	Bytes	Meaning
Opcode	0x80	**CONNECT**, Final bit set
	0x0015	packet length = 21
	0x10	version 1.0 of OBEX
	0x00	flags, all zero for this version of OBEX
	0x0200	512 bytes is the max OBEX packet size client can accept
	0x46	HI for **Target** header
	0x000E	Length of **Target** header
	0x53594E434D4C	UUID for SyncML DM ("SYNCML-DM")
	2D444D	
Server Response:		
response code	0xA0	SUCCESS, Final bit set
	0x001A	packet length of 26
	0x10	version 1.0 of OBEX
	0x00	Flags
	0x0200	1K max packet size
	0xCB	HI for **Connection Id** header
	0x00000001	ConnId = 1
	0x4A	**Who** HI
	0x000E	Length of **Who** Header
	0x53594E434D4C	UUID of responding application (same value as Target header in request ("SYNCML-DM")
	2D444D	

7.2   OBEX Disconnect Example

Client Request:	bytes	Meaning
opcode	0x81	**DISCONNECT**, Final bit set
	0x0008	packet length = 8
	0xCB	HI for **Connection Id** header
	0x00000001	ConnId = 1
Server Response:		
response code	0xA0	SUCCESS, Final bit set
	0x0003	packet length = 3

7.3   OBEX Abort Example

Client Request:	bytes	Meaning
opcode	0xFF	**ABORT**, Final bit set
	0x0008	packet length = 8
	0xCB	HI for **Connection Id** header
	0x00000001	ConnId = 1
Server Response:		
response code	0xA0	SUCCESS, Final bit set
	0x0003	packet length = 3

7.4   OBEX Put Example

Client Request:	Bytes	Meaning
opcode	0x02	**PUT**, Final bit not set
	0x0200	512 bytes is length of packet
	0xCB	HI for **Connection Id** header
	0x00000001	ConnId = 1
	0x42	HI for **Type** header
	0x0020	Length of **Type** header
	0x6170706C6963	Type of object (application/vnd.syncml.dm+wbxml)

	6174696F6E2F76	(null terminated ASCII text)
	6E642E73796E63	
	6D6C2E646D2B776 2786D 6C00	
	0xC3	HI for **Length** header
	0x000004FE	Length of object is 1278 bytes
	0x48	HI for Object **Body** chunk header
	0x01D3	Length of **Body** header = 467. 464 plus HI and header length
	0x..........	464 bytes of body
Server Response:		
response code	0x90	CONTINUE, Final bit set
	0x0003	length of response packet
Client Request:		
opcode	0x02	**PUT**, Final bit not set
	0x0200	512 bytes is length of packet
	0x48	HI for Object **Body** chunk
	0x01FD	Length of **Body** header = 509. 506 plus HI and header length
	0x..........	next 506 bytes of body
Server Response:		
response code	0x90	CONTINUE, Final bit set
	0x0003	length of response packet
Client Request:		
opcode	0x82	**PUT**, Final bit set
	0x013A	314 bytes is length of packet
	0x49	HI for Object **End-of-Body** chunk
	0x0137	Length of header = 311. 308 plus HI and header length
	0x..........	last 308 bytes of body
Server Response:		
response code	0xA0	SUCCESS, Final bit set

	0x0003	length of response packet

7.5   OBEX Get Example

Client Request:	bytes	Meaning
opcode	0x83	**GET**, Final bit set
	0x0028	40 bytes is length of **GET** packet
	0xCB	HI for **Connection Id** header
	0x00000001	ConnId = 1
	0x42	HI for **Type** header
	0x0020	Length of **Type** header
	0x6170706C6963	Type of object (application/vnd.syncml.dm+wbxml)
	6174696F6E2F76	(null terminated ASCII text)
	6E642E73796E63	
	6D6C2E646D 2B7762786D6C00	
Server Response:		
Response code	0x90	CONTINUE, Final bit set
	0x0200	length of response packet
	0xC3	HI for **Length** header
	0x000002BE	Length of object is 702 bytes
	0x48	HI for Object **Body** chunk
	0x01F8	Length of header = 504. 501 plus HI and header length
	0x..........	501 bytes of body
Client Request:	**bytes**	**Meaning**
opcode	0x83	**GET**, Final bit set
	0x0003	3 bytes is length of **GET** packet
Server Response:		
Response code	0xA0	SUCCESS, Final bit set
	0x0038	length of response packet
	0x49	HI for **End-of-Body** chunk

0x00CC   Length of header = 204. 201 plus HI and header length

0x..........   last 201 bytes of body.

Appendix A.   Change History   (Informative)

A.1   Approved Version History

Reference	Date	Description
N/A	N/A	No prior 1.3 version

A.2   Draft/Candidate Version 1.3 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-SyncML_OBEXBinding-V1_3	27 Aug 2009	All	New baseline, using OMA-TS-SyncML_OBEXBinding-V1_2-20070221-A
	25 Nov 2009	All	Incorporated CR OMA-DM-DM13-2009-0097R02-CR_OBEXBinding_cleanup
Draft Versions OMA-TS-DM_OBEXBinding-V1_3	08 Jan 2010	All	DSO general editorial clean-up
	04 May 2010	1, 4, 6.7, App B	Incorporated CR OMA-DM-DM13-2010-0071R01-CR_OBEX_MIME Deletion of a typo in 6.7 Restoration of a reference in App B
Candidate Version OMA-TS-DM_OBEXBinding-V1_3	25 May 2010	N/A	Status changed to Candidate by TP Ref # OMA-TP-2010-0221-INP_DM_V1.3_ERP_and_ETR_for_Candidate_approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [IOPPROC]. The static conformance requirements are specified for Client Initiated and Server Initiated Sessions.

B.1 Client Features

B.1.1 Common SCRs – OBEX Session Requirements

The following OBEX operations are REQUIRED for all sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-C-001	Support for OBEX Connect operation	6.1	M	
DSDM-OBEX-C-002	Support for OBEX Disconnect operation	6.1	M	
DSDM-OBEX-C-003	Support for OBEX Put operation	6.1	M	
DSDM-OBEX-C-004	Support for OBEX Get operation	6.1	M	

B.1.2 SCRs for Client Initiated Sessions

The following OBEX operations are REQUIRED for Client Initiated sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-C-005	DM Client/OBEX Client support for OBEX Abort operation	6.1	O	

B.1.3 SCRs for Server Initiated Sessions

The following OBEX operations are REQUIRED for Server Initiated sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-C-006	DM Client/OBEX Server support for OBEX Abort operation	6.1	M	

B.2 Server Features

B.2.1 Common SCRs

The following OBEX operations are REQUIRED for all sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-S-001	Support for OBEX Connect operation	6.1	M	
DSDM-OBEX-S-002	Support for OBEX Disconnect operation	6.1	M	
DSDM-OBEX-S-003	Support for OBEX Put operation	6.1	M	
DSDM-OBEX-S-004	Support for OBEX Get operation	6.1	M	

B.2.2 SCRs for Client Initiated Sessions

The following OBEX operations are REQUIRED for Client Initiated sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-S-005	DM Server/OBEX Server support for OBEX Abort operation	6.1	M	

B.3   SCRs for Server Initiated Sessions

The following OBEX operations are REQUIRED for Server Initiated sessions.

Item	Function	Ref.	Status	Requirement
DSDM-OBEX-S-006	DM Server/OBEX Client support for OBEX Abort operation	6.1	O	