Infrared Data Assciation® (IrDA®) Object Exchange Protocol (OBEX™) Test Specification

Version 1.0.1

Document History

Date	**Description**
2002-08-09	Moved existing OBEX tests from the IrFM test specification.
2002-08-13	Restructured event sequence charts.
2002-08-14	Added the Test Status field to track the progress on each test.
2002-09-06	Added more OBEX Session tests.
2002-09-09	Added the rest of the OBEX 1.2 Test Specification.
2002-09-24	Test Updates.
2002-10-01	Added mandatory and optional OBEX feature list.
2002-11-08	Added Spurious Transport Disconnection tests.
2002-11-25	Authentication test updates.
2002-12-1	Changed all tests to a Test Status of “Accepted”.
2003-01-03	Moved the mandatory/optional tables into the OBEX specification.

Table of Contents

1 Scope

This document defines the tests that are necessary to verify that a device that implements either role defined by [OBEX] is minimally compliant with the OBEX protocol. This specification will focus on the protocol implementation. IrDA believes that testing to the protocol is the most reliable means of assuring interoperability and compatibility between devices. Where a discrepancy exists between a relevant IrDA protocol document and this document, it should be assumed that the protocol document is normative. Please report any discrepancies to the current editor of this document.

It is assumed that each implementer has thoroughly tested their device during and after development. The test cases defined herein do not exercise every combination of commands and parameters that are possible to encounter during a given transaction. The requirement of demonstrating the behaviors defined below will give IrDA, its member organizations and the end user a certain level of confidence that the device will interoperate with other devices having complementary functions.

Test cases that have been defined in this specification are intended to exercise, at some level, all of the behaviors that a device must support. Where optional behaviors are claimed to be supported by a device, they must also be tested to ensure that they conform to the Protocol specifications.

For an implementation of the OBEX Protocol to be compliant with this specification, it must support a set of required operations. It makes sense to require a minimum level of support because these test procedures are used to certify IrDA reference devices. Reference devices are then used to certify interoperability between OBEX devices. Without specifying a minimal level of functionality, interoperability cannot be assured.

Interoperability testing should be a part of the compliance process. This specification does not attempt to define the method or extent of interoperability testing to be required. Implementers should attempt to create environments and conditions that will reflect the most common usage of their device. Implementers should diagnose the failures to help identify shortcomings in either the devices involved or the Protocol itself.

1.1 *Contributors*

Original author

Kevin Hendrix, Extended Systems (kevinh@extendsys.com)

Current editor

Kevin Hendrix, Extended Systems (kevinh@extendsys.com)

Other contributors

Extended Systems	Glade Diviney

1.2 *References*

[IrFM PnP]	*Infrared Financial Messaging Point and Pay Profile*
[IrFM Test]	*Infrared Financial Messaging Test Specification*
[OBEX]	*IrDA Object Exchange Protocol OBEX*
[IrLAP]	*Serial Infrared Link Access Protocol, IrLAP, Version 1.1*
[IrLMP]	*Link Management Protocol, IrLMP, Version 1.1*

2 Testing Procedures

It is the intent of this document to clearly define the behavior that must be observed for a given device.

2.1 *Test suite organization*

Test suites are broken into major test groups (such as “OBEX Client Tests”), test subgroups (such as “OBEX Connect PDU”), and individual tests (such as “Simple Connect Operation”). The major test group isolates the overall behavior being tested on the Implementation Under Test (IUT). Therefore, the “OBEX Client Tests” major group will test the ability of the IUT to initiate OBEX operations properly. Similarly, the “OBEX Server Tests” major test group will test the ability of the IUT to respond to various OBEX operations properly.

2.2 *Test organization*

The meaning of each section of an individual test is defined below.

Section	Example	Meaning
Test Title	“Test C-C-1: Simple Connect Operation…”	Defines a unique test identifier including test group, subgroup, and ordinal. Also provides a summary name and a description of the test.
Test Status	“Ready For Review”	Indicates the current development status of the test. The current status options are: “New”, “In Progress by Person X (Company Y)”, Ready For Review”, “Under Review by Person X (Company Y)”, and “Accepted”.
Test Procedure	“See **A.1.1** for a complete diagram of the event sequence chart…”	A complete definition of the required data exchanges from the OBEX Tester Unit to the Implementation Under Test (IUT) during the execution of a test. Usually includes a reference to Appendix A for a full message sequence chart. This section may also document changes or differences that supercede the referenced chart.
Test Condition	“If support for the Connect PDU is not present…”	Indicates the conditions under which a particular test needs to be executed. For example, if [OBEX] indicates that a particular feature is optional, the corresponding test(s) will be marked as conditional upon that feature being present.
Expected Outcome	“Pass Verdict: The OBEX Client transmits a Connect Request…”	Describes the specific conditions required for the test to reach a pass or fail verdict. These specific conditions will always describe the behavior required of the Implementation Under Test (IUT). May also include an “Inconclusive Verdict” section that describes conditions that may be valid, but prevent the test from running. If a test is required for a particular device, then it must be re-executed until a pass verdict is reached.
Notes	“Various other headers may be transmitted…”	Informative comments that describe acceptable variations in test results, comments on test setup, etc.

2.3 *Test Numbering*

Tests defined by this specification use a numbering system that identifies the following:

- Whether the test is for the client or server.
- The operation that the test exercises.

- The number of the test within the set.

For example the test numbered “C-C-1” is the first Client test for the **CONNECT** Operation. Required tests are noted with the statement **REQUIRED** within the “Test Conditions” section of the test.

2.4 *Test Devices*

Two devices are used when executing each OBEX Test:

- **OBEX Tester Unit** – the exact behavior of this device is described in the “Test Procedure” section of each test. This section will also describe the generalized behavior that can be expected from the IUT, but only when it is necessary to clearly describe when certain behavior must occur on the OBEX Tester Unit.
- **Implementation Under Test** (IUT) – the exact behavior of this device is described in the “Expected Outcome” section of each test. The exact requirements for Pass, Fail, or Inconclusive results are described in this section.

2.5 *OBEX Testing Requirements*

Refer to [OBEX] for a complete listing of all the mandatory and optional OBEX Roles, IrDA-Specific Features, and OBEX Features for both the OBEX Client and OBEX Server devices. All of the OBEX tests within this document follow the testing requirements presented in the [OBEX] document.

2.5.1 Required Behavior

Required tests are denoted with a **REQUIRED** statement in the “Test Conditions” section of each test. If a device does not support an OBEX server (at all) then the required server tests can be eliminated. The same holds true for the required client tests.

2.6 *Test Environment*

2.6.1 Physical Setup

Two ports will be placed facing each other on a black, horizontal, insulating surface .75m apart. Devices manufactured to the short range IrDA physical standard of 20cm should be placed 10cm apart.

2.6.2 Electromagnetic Interference Sources

The ability of the product to exhibit correct protocol behavior should not be affected by environmental conditions during protocol testing. Note that the physical capabilities of hardware are evaluated by tests described in the Physical Layer Test Specification.

The protocol test area is flexible for the applicant. However, it is expected to be easily reproducible for test verification. The applicant will describe the light sources used in the test area including information regarding: distance to the test surface, orientation with respect to the ports, manufacturer, part number and intensity. All equipment is expected to be common and available. The applicant will further describe any other E-M sources that could potentially affect hardware functionality. Drawings are appropriate. Alternatively, the applicant may choose to submit measurements of the intensity of emissions in the center of the test area (lux) in a wavelength range of 100nm to .05mm.

NOTE: Refer to Appendix A.1 of the SIR Physical Layer Link Specification for expected capabilities of any IrDA product. As a guideline, it is recommended that the test environment fall within the parameters described in A.1.

2.6.3 Test Personnel

This test specification was written to give the Test Engineers in your group a set of OBEX test cases. This specification assumes that the Test Engineers are familiar with the terms and procedures of the protocol. It is assumed that the Test Engineers have the ability to generate specific frames and procedures and can also record both sides of the transaction.

3 OBEX Client Tests

Tests in this section assume the Device Under Test is acting as the OBEX Client and the tester device is acting as the OBEX Server.

3.1 *OBEX Connect PDU*

3.1.1 Test C-C-1: Simple Connect Operation

Demonstrates the transmission of a simple OBEX Connect operation.

3.1.1.1 *Test Status*

Accepted

3.1.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet.
- The Connect Response packet length is seven bytes (0x0007).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- No headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

3.1.1.3 *Test Condition*

If support for the Connect PDU is not present, this test may be **SKIPPED**.

3.1.1.4 *Expected Outcome*

3.1.1.4.1 Pass Verdict

The OBEX Client transmits a Connect Request. The following are true:

- The entire Connect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Connect Packet Length field indicates the correct size.
- The Protocol field indicates OBEX version 1.0.
- The Flags field has no bits set.
- The Maximum OBEX Packet Length field specifies a size of 255 or above.

The Connect Operation is successful.

3.1.1.4.2 Fail Verdict

The OBEX Client does not transmit a Connect Request.

The Connect Request does not contain acceptable values.

The Connect Operation is not successful.

3.1.1.5 *Notes*

Various headers may be transmitted with the Connect Request; these do not affect the result of the test.

3.1.2 Test C-C-2: Simple Directed Connection

Demonstrates the transmission of a simple directed Connect Request PDU.

3.1.2.1 *Test Status*

Accepted

3.1.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Connect Request. The following must occur:

- The Connect Request contains a **Target** (0x46) header describing the desired OBEX service.
- The Connect Request can optionally contain a **Who** (0x4A) header if it is necessary to describe the client initiating the connection.

The OBEX Server transmits a Connect Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet.
- The Connect Response packet length does not exceed the default OBEX packet size of 255 bytes, but will vary based on the headers included in the response.
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Response contains a properly formed four-byte **Connection Identifier** (0xCB) header as the first header in the packet. The length of this header is five bytes and its value will be 0x00000001.
- The Connect Response also contains a properly formed **Who** (0x4A) header. The length of this header is a two-byte value including 3 bytes of header description, but will vary based on the length of the UUID received in the Connect Request's Target header. The value of the Who header will include the UUID from the received Target header.
- The Connect Response may optionally contain a **Target** (0x46) header. This header will be formed if a Who header was received in the Connect Request. The length of this header is a two-byte value including 3 bytes of header description, but will vary based on the length of the client description received in the Connect Request's Who header. The value of the Target header will include the client description from the received Who header.
- No other headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

3.1.2.3 *Test Condition*

If support for the Connect PDU is not present, this test may be **SKIPPED**.

3.1.2.4 *Expected Outcome*

3.1.2.4.1 Pass Verdict

The OBEX Client transmits a Connect Request. The following are true:

- The entire Connect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set
- The Connect Packet Length field indicates the correct size.
- The Protocol field indicates OBEX version 1.0.
- The Flags field has no bits set.
- The Maximum OBEX Packet Length field specifies a size of 255 or above.

- The Connect PDU contains a properly formed **Target** header for the OBEX service being used. This Target header should describe a valid OBEX service. See the OBEX specification for a listing of the valid OBEX services.

The Connect Operation is successful.

3.1.2.4.2 Fail Verdict

The OBEX Client does not transmit a Connect Request.

The Connect Request does not contain acceptable values.

The Connect Response is not successful.

3.1.2.5 *Notes*

Various other headers may be transmitted with the Connect Request; these do not affect the result of the test.

3.2 *OBEX Disconnect PDU*

3.2.1 Test C-D-1: Simple Disconnect Operation

Demonstrates the transmission of a Disconnect Request PDU.

3.2.1.1 *Test Status*

Accepted

3.2.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server successfully responds to an OBEX Connect operation. See **3.1.1.2** for the full description of this process.

The OBEX Server receives a Disconnect Request.

The OBEX Server transmits a Disconnect Response. The following must occur:

- The Success response code with the Final Bit set is sent (0xA0) in response to the Disconnect Request packet.

See **A.2.1** for a complete diagram of the event sequence chart.

3.2.1.3 *Test Condition*

If support for the Disconnect PDU is not present, this test may be **SKIPPED**.

3.2.1.4 *Expected Outcome*

3.2.1.4.1 Pass Verdict

The OBEX Connection is established successfully.

The OBEX Client transmits a Disconnect Request. The following are true:

- The entire Disconnect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.

The Disconnect Operation is successful.

3.2.1.4.2 Fail Verdict

The OBEX Connection is unsuccessful.

The OBEX Client does not transmit a Disconnect Request.

The Disconnect Request does not contain acceptable values.

The Disconnect Operation is not successful.

3.2.1.5 *Notes*

Various headers may be transmitted with the Disconnect Request; these do not affect the result of the test.

3.2.2 Test C-D-2: Simple Directed Disconnect Operation

Demonstrates the transmission of a directed Disconnect Request PDU.

3.2.2.1 *Test Status*

Accepted

3.2.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server successfully responds to a directed OBEX Connect operation. See **3.1.2.2** for the full description of this process.

The OBEX Server receives a Disconnect Request.

- The Disconnect Request contains a **Connection ID** (0xCB) header describing the desired OBEX service.

The OBEX Server transmits a Disconnect Response. The following must occur:

- The Success response code with the Final Bit set is sent (0xA0) in response to the Disconnect Request packet.

See **A.2.1** for a complete diagram of the event sequence chart.

3.2.2.3 *Test Condition*

If support for the Disconnect PDU is not present, this test may be **SKIPPED**.

3.2.2.4 *Expected Outcome*

3.2.2.4.1 Pass Verdict

The directed OBEX Connection is established successfully.

The OBEX Client transmits a Disconnect Request. The following are true:

- The entire Disconnect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Disconnect PDU contains a properly formed **Connection ID** header. This header must contain a 4-byte value matching the Connection ID header sent by the OBEX Server in the Connect Response (0x00000001).

The Disconnect Operation is successful.

3.2.2.4.2 Fail Verdict

The directed OBEX Connection is unsuccessful.

The OBEX Client does not transmit a Disconnect Request.

The Disconnect Request does not contain acceptable values.

The Disconnect Operation is not successful.

3.2.2.5 *Notes*

Various headers may be transmitted with the Disconnect Request; these do not affect the result of the test.

3.3 *OBEX SetPath PDU*

3.3.1 Test C-SP-1: Simple SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation for a downward path change.

3.3.1.1 *Test Status*

Accepted

3.3.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the SetPath Request packet.

See **A.3.1** for a complete diagram of the event sequence chart.

3.3.1.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

3.3.1.4 *Expected Outcome*

3.3.1.4.1 Pass Verdict

The OBEX Client transmits a SetPath Request. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The SetPath Packet Length indicates the correct size.
- The Flags field is expected to have no bits set. However, it is acceptable if the **Don't create the directory if it does not exist** bit (bit 1) is set. All other bits must be set to 0.
- The Constants field has no bits set.
- The SetPath PDU contains a properly formed **Name** header. This Name header must indicate the downward (sub-directory) path change for the OBEX Server device.

The SetPath Operation is successful.

3.3.1.4.2 Fail Verdict

The OBEX Client does not transmit a SetPath Request.

The SetPath Request does not contain acceptable values.

The SetPath Operation is not successful.

3.3.1.5 *Notes*

Various other headers may be transmitted with the SetPath Request; these do not affect the result of the test.

3.3.2 Test C-SP-2: Backup SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation for an upward path change.

3.3.2.1 *Test Status*

Accepted

3.3.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the SetPath Request packet.

See **A.3.1** for a complete diagram of the event sequence chart.

3.3.2.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

3.3.2.4 *Expected Outcome*

3.3.2.4.1 Pass Verdict

The OBEX Client transmits a SetPath Request. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The SetPath Packet Length indicates the correct size.
- The Flags field has the **Backup** bit (bit 0) set. However, it is acceptable if the **Don't create the directory if it does not exist** bit (bit 1) is set, as this behavior is undefined in the OBEX specification. All other bits must be set to 0.
- The Constants field has no bits set.

The SetPath Operation is successful.

3.3.2.4.2 Fail Verdict

The OBEX Client does not transmit a SetPath Request.

The SetPath Request does not contain acceptable values.

The SetPath Operation is not successful.

3.3.2.5 *Notes*

Various other headers may be transmitted with the SetPath Request; these do not affect the result of the test.

3.3.3 Test C-SP-3: Reset SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation to reset to the default path.

3.3.3.1 *Test Status*

Accepted

3.3.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the SetPath Request packet.

See **A.3.1** for a complete diagram of the event sequence chart.

3.3.3.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

3.3.3.4 *Expected Outcome*

3.3.3.4.1 Pass Verdict

The OBEX Client transmits a SetPath Request. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The SetPath Packet Length indicates the correct size.

- The Flags field is expected to have no bits set. However, it is acceptable if the **Don't create the directory if it does not exist** bit (bit 1) is set, as this behavior is undefined in the OBEX specification. All other bits must be set to 0.
- The Constants field has no bits set.
- The SetPath PDU contains a properly formed empty **Name** header. An empty Name header indicates a path reset request.

The SetPath Operation is successful.

3.3.3.4.2 Fail Verdict

The OBEX Client does not transmit a SetPath Request.

The SetPath Request does not contain acceptable values.

The SetPath Operation is not successful.

3.3.3.5 *Notes*

Various other headers may be transmitted with the SetPath Request; these do not affect the result of the test.

3.4 *OBEX Get PDU*

3.4.1 Test C-G-1: Simple Get Operation

Demonstrates the retrieval of a 25-byte object using the OBEX Get operation.

3.4.1.1 *Test Status*

Accepted

3.4.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of these Get responses is three bytes (0x0003).
- The Continue response code with the Final Bit set (0x90), along with a **Length** (0xC3) header will be sent in response to the first Get Request packet with the Final Bit set. The length of the header is 5 bytes and will contain the length of the object to be transferred (25 bytes or 0x00000019). The length of this Get response is eight bytes (0x0008).
- The Success response code with the Final Bit set (0xA0), along with an **End Of Body** (0x49) header will be sent in the last response packet. The **25-byte** object will be sent in response to any **Name** header sent in the Get Request. The End Of Body header will contain 25 bytes of random data forming the object being sent. The length of this header is 28 bytes (0x001C). The overall length of this Get Response is 31 bytes (0x001F).

See **A.4.1** for a complete diagram of the event sequence chart.

3.4.1.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

3.4.1.4 *Expected Outcome*

3.4.1.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client. The following are true:

- The Get Packet Length indicates the correct size.
- The Get Request PDU contains a properly formed **Name** header indicating the name of the object being requested. The name must exist, but is arbitrary, as the Server should respond with a 25-byte object in response to any name.
- All Get Request packets that do not contain the final segment of headers must not have the Final Bit set.
- All Get Request packets that do not contain headers must have the Final Bit set.
- All Get Request packets without the Final bit set must occur prior to any Get Request packets with the Final Bit set.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Get Request packets without the final bit set are used to send multiple Get Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Get Operation is successful. The following are true:

- A 25-byte object should have been received and stored according to the name provided by the OBEX Client.

3.4.1.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Get Requests do not contain acceptable values.

The Get Operation is not successful.

The Get Operation does not result in a 25-byte object being transferred.

3.4.1.5 *Notes*

Various other headers may be transmitted with the Get Requests; these do not affect the result of the test.

3.4.2 Test C-G-2: Maximum Get Operation

Demonstrates the retrieval of a 10 Kbyte object using the OBEX Get operation.

3.4.2.1 *Test Status*

Accepted

3.4.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get packets must fit within the 255 minimum OBEX packet size. This ensures that all devices can interoperate.
- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of such a Get response is three bytes (0x0003).
- The Continue response code with the Final Bit set (0x90), along with the **Length** (0xC3) and **Body** (0x48) headers will be sent in response to the first Get Request packet with the Final Bit set. The Length header is 5 bytes and will contain the length of the object

to be transferred (10240 bytes or 0x00002800). The Body header has a length of 43 bytes (0x002B) and will contain the first 40 bytes of the object being transferred. The length of this Get response is 51 bytes (0x0033).

- **50** additional Get Response packets will be sent with the remainder of the Body header data.
- The last Get Response packet has the Success response code with the Final Bit set (0xA0).
- **10240 bytes** of object data will be sent in response to any **Name** header sent in the Get operation. The first packet with body header data will contain 40 bytes of object data, while the next 50 packets will contain 204 bytes each.
- All subsequent Get Response packets will contain a properly formed **Body** (0x48) header with the exception of the last Get packet that contains an **End Of Body** (0x49) header instead. The Body/End Of Body header will contain 204 bytes of random object data. The length of this header is 207 bytes (0x00CF). The overall length of each Get Response is 210 bytes (0x00D2).
- No other headers are included.

See **A.4.1** for a complete diagram of the event sequence chart.

3.4.2.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

3.4.2.4 *Expected Outcome*

3.4.2.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client. The following are true:

- The Get Packet Length indicates the correct size.
- The Get Request PDU contains a properly formed **Name** header indicating the name of the object being requested. The name must exist, but is arbitrary, as the Server should respond with a 10 Kbyte object in response to any name.
- All Get Request packets that do not contain the final segment of headers must not have the Final Bit set.
- All Get Request packets that do not contain headers must have the Final Bit set.
- All Get Request packets without the Final bit set must occur prior to any Get Request packets with the Final Bit set.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Get Request packets without the final bit set are used to send multiple Get Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Get Operation is successful. The following are true:

- A 10 Kbyte object should have been received and stored according to the name provided by the OBEX Client.

3.4.2.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Get Requests do not contain acceptable values.

The Get Operation is not successful.

The Get Operation does not result in a 10 Kbyte object being transferred.

3.4.2.5 *Notes*

Various other headers may be transmitted with the Get Requests; these do not affect the result of the test.

3.4.3 Test C-G-3: Zero Byte Get Operation

Demonstrates the retrieval of a 0-byte object using the OBEX Get operation.

3.4.3.1 *Test Status*

Accepted

3.4.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of such a Get response is three bytes (0x0003).
- A **0-byte** object will be sent in response to any **Name** header sent in the Get Request.
- The Success response code with the Final Bit set (0xA0), along with the **Length** (0xC3) and **End Of Body** (0x49) headers will be sent in response to the first Get Request packet with the Final Bit set. The Length header is 5 bytes and will contain the length of the object to be transferred (0 bytes or 0x00000000). The End Of Body header has a length of 3 bytes (0x0003) and will contain the **0-byte** object being transferred. The length of this Get response is 11 bytes (0x000B).

See **A.4.1** for a complete diagram of the event sequence chart.

3.4.3.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

3.4.3.4 *Expected Outcome*

3.4.3.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client. The following are true:

- The Get Packet Length indicates the correct size.
- The Get Request PDU contains a properly formed **Name** header indicating the name of the object being requested. The name must exist, but is arbitrary, as the Server should respond with a 0-byte object in response to any name.
- All Get Request packets that do not contain the final segment of headers must not have the Final Bit set.
- All Get Request packets that do not contain headers must have the Final Bit set.
- All Get Request packets without the Final bit set must occur prior to any Get Request packets with the Final Bit set.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Get Request packets without the final bit set are used to send multiple Get Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Get Operation is successful. The following are true:

- A 0-byte object should have been received and stored according to the name provided by the OBEX Client.

3.4.3.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Get Requests do not contain acceptable values.

The Get Operation is not successful.

The Get Operation does not result in a 0-byte object being transferred.

3.4.3.5 *Notes*

Various other headers may be transmitted with the Get Requests; these do not affect the result of the test.

3.4.4 Test C-G-4: Default Object Get Operation

Demonstrates the retrieval of a 25-byte default object using the OBEX Get operation.

3.4.4.1 *Test Status*

Accepted

3.4.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of such a Get response is three bytes (0x0003).
- A **25-byte** default business card object will be sent in response to the empty or nonexistent **Name** header sent in the Get Request.
- The Success response code with the Final Bit set (0xA0), along with the **Length** (0xC3) and **End Of Body** (0x49) headers will be sent in response to the first Get Request packet with the Final Bit set. The Length header is 5 bytes and will contain the length of the object to be transferred (25 bytes or 0x00000019). The End Of Body header will contain 25 bytes of random data forming the default business card object being sent. The length of this header is 28 bytes (0x001C). The length of this Get response is 36 bytes (0x0024).

See **A.4.1** for a complete diagram of the event sequence chart.

3.4.4.3 *Test Condition*

If support for the Get PDU or the Default Get behavior is not present, this test may be **SKIPPED**.

3.4.4.4 *Expected Outcome*

3.4.4.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client. The following are true:

- The Get Packet Length indicates the correct size.
- The Get Request PDU may contain a properly formed **Name** header, but it must be empty if it exists. It is also legal to send no Name header at all. Either of these situations indicates a request for the default object.
- The Get Request PDU contains a properly formed **Type** header indicating the type of default object to be retrieved. In this case the value should be “text/x-vCard”.
- All Get Request packets that do not contain the final segment of headers **must not** have the Final Bit set.
- All Get Request packets that do not contain headers **must** have the Final Bit set.
- All Get Request packets without the Final bit set **must** occur prior to any Get Request packets with the Final Bit set.
- Issuing of the Name header can occur in either event 1 or event 3 in the event sequence chart. Get Request packets without the final bit set are used to send multiple Get Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Get Operation is successful. The following are true:

- A 25-byte default object should have been received and stored by the OBEX Client.

3.4.4.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Get Requests do not contain acceptable values.

The Get Operation is not successful.

The Get Operation does not result in a 25-byte object being transferred.

3.4.4.5 *Notes*

Various other headers may be transmitted with the Get Requests; these do not affect the result of the test.

3.5 *OBEX Put PDU*

3.5.1 Test C-P-1: Simple Put Operation

Demonstrates the transmission of at least a 25-byte object using the OBEX Put operation.

3.5.1.1 *Test Status*

Accepted

3.5.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

See **A.5.1** for a complete diagram of the event sequence chart.

3.5.1.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

3.5.1.4 *Expected Outcome*

3.5.1.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- Combination of Body and End Of Body headers should meet or exceed the 25-byte object size requirement.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Put Request packets without the final bit set are used to send multiple Put

Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Put Operation is successful.

3.5.1.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.5.1.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.5.2 Test C-P-2: Maximum Put Operation

Demonstrates the transmission of at least a 10-Kbyte object using the OBEX Put operation.

3.5.2.1 *Test Status*

Accepted

3.5.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

See **A.5.1** for a complete diagram of the event sequence chart.

3.5.2.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

3.5.2.4 *Expected Outcome*

3.5.2.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- Combination of Body and End Of Body headers should meet or exceed the 10-Kbyte object size requirement.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Put Request packets without the final bit set are used to send multiple Put Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Put Operation is successful.

3.5.2.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.5.2.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.5.3 Test C-P-3: Zero Byte Put Operation

Demonstrates the transmission of a 0-byte object using the OBEX Put operation.

3.5.3.1 *Test Status*

Accepted

3.5.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

See **A.5.1** for a complete diagram of the event sequence chart.

3.5.3.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

3.5.3.4 *Expected Outcome*

3.5.3.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed empty **End Of Body** header as its final header in the Put operation.
- Combination of Body and End Of Body headers should meet the 0-byte object size requirement.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Put Request packets without the final bit set are used to send multiple Put Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Put Operation is successful.

3.5.3.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.5.3.4.3 Inconclusive Verdict

The OBEX Client sends an object larger than 0-bytes.

3.5.3.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.5.4 Test C-P-4: Put Using Advertised Packet Size

Demonstrate that the test C-P-2 when performed after a successful OBEX **CONNECT** operation utilizes the larger OBEX packet size advertised in the **CONNECT** response.

3.5.4.1 *Test Status*

Accepted

3.5.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet.
- The Connect Response packet length is seven bytes (0x0007).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5K bytes (0x1400).
- No headers are included.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

See **A.5.3** for a complete diagram of the event sequence chart.

3.5.4.3 *Test Condition*

If support for the Connect PDU is not present, this test may be **SKIPPED**. The Put PDU is **REQUIRED** by the OBEX protocol.

3.5.4.4 *Expected Outcome*

3.5.4.4.1 Pass Verdict

The OBEX Client transmits a Connect Request.

The OBEX Client receives a Connect Response.

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size. For the Put Requests including object data, the larger OBEX packet size (5K) supported by the OBEX Server should be utilized. If packet sizes greater than 1K are used, this is sufficient.

- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- Combination of Body and End Of Body headers should meet or exceed the 10-Kbyte object size requirement.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Put Request packets without the final bit set are used to send multiple Put Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Put Operation is successful.

3.5.4.4.2 Fail Verdict

The OBEX Client does not transmit a Connect Request.

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.5.4.5 *Notes*

Various other headers may be transmitted with the Connect Request or the Put Requests; these do not affect the result of the test.

3.5.5 Test C-P-5: Put Delete Operation

Demonstrates the transmission of an OBEX Put operation to delete an object.

3.5.5.1 *Test Status*

Accepted

3.5.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

See **A.5.1** for a complete diagram of the event sequence chart.

3.5.5.3 *Test Condition*

If Put Delete operations are not supported, this test may be **SKIPPED**.

3.5.5.4 *Expected Outcome*

3.5.5.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being deleted.

- The Put Request PDU contains no **Body** or **End Of Body** headers.
- Issuing of the headers can occur in either event 1 or event 3 in the event sequence chart. Put Request packets without the final bit set are used to send multiple Put Request packets with header information. Based on the OBEX Client being used, events 1 and 2 may not be needed, and event 3 may be sufficient.

The Put Operation is successful.

3.5.5.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.5.5.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.6 *OBEX Abort PDU*

3.6.1 Test C-A-1: Simple Put Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Put operation. At least a 256-byte object should be transferred, in order to guarantee that the Put operation takes at least two packets to complete. However, larger objects will have a better chance of being aborted, since they will give the OBEX Client more time to abort the operation.

3.6.1.1 *Test Status*

Accepted

3.6.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.

The OBEX Server will also transmit an Abort Response if an Abort Request is received. The following must occur:

- The Success response code with the Final Bit set (0xA0) will be sent.
- The Abort Response will have a length of three bytes (0x0003).

See **A.6.1** for a complete diagram of the event sequence chart.

3.6.1.3 *Test Condition*

If support for the Abort PDU is not present, this test may be **SKIPPED**. The Put PDU is **REQUIRED** by the OBEX protocol.

3.6.1.4 *Expected Outcome*

3.6.1.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client.

The OBEX Client transmits an Abort Request. The following is true:

- The Abort Packet Length indicates the correct size.

- The Final Bit is set.

The Put Operation is successfully canceled, with no further Put Request packets being sent.

3.6.1.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Abort Request does not contain acceptable values.

The Put Operation is not successfully canceled.

3.6.1.4.3 Inconclusive Verdict

The OBEX Client does not transmit an Abort Request.

3.6.1.5 *Notes*

Various headers may be transmitted with the Put Requests or the Abort Request; these do not affect the result of the test.

3.6.2 Test C-A-2: Immediate Put Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Put operation. The Abort Request is issued with one outstanding Put Request, without waiting for the subsequent Put Response packet from the OBEX Server. This behavior is valid only with the Abort PDU. At least a 256-byte object should be transferred, in order to guarantee that the Put operation takes at least two packets to complete. However, larger objects will have a better chance of being aborted, since they will give the OBEX Client more time to abort the operation.

3.6.2.1 *Test Status*

Accepted

3.6.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- All Put Response packets have a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- Each Put Response will wait **500ms** before sending the next response in order to give the OBEX Client adequate time to abort the operation before the next Put response is sent.

The OBEX Server will transmit an Abort Response if an Abort Request is received. The Abort Request is expected to be received immediately after the Put Request, but before the Put Response is issued. The following must occur:

- The Success response code with the Final Bit set (0xA0) will be sent.
- The Abort Response will have a length of three bytes (0x0003).

See **A.6.1** for a complete diagram of the event sequence chart.

3.6.2.3 *Test Condition*

If support for the Abort PDU is not present, this test may be **SKIPPED**. The Put PDU is **REQUIRED** by the OBEX protocol.

3.6.2.4 *Expected Outcome*

3.6.2.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client.

The OBEX Client transmits an Abort Request. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set.
- The Abort is issued while a Put command is outstanding. This violates the typical OBEX command/response order, but is allowed for OBEX Abort operations only.

The Put Operation is successfully canceled, with no further Put Request packets being sent.

3.6.2.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Abort Request does not contain acceptable values.

The Put Operation is not successfully canceled.

3.6.2.4.3 Inconclusive Verdict

The OBEX Client does not transmit an Abort Request.

The Abort Request is issued in proper OBEX command/response order instead of while a Put command is outstanding.

3.6.2.5 *Notes*

Various headers may be transmitted with the Put Requests or the Abort Request; these do not affect the result of the test.

3.6.3 Test C-A-3: Simple Get Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Get operation.

3.6.3.1 *Test Status*

Accepted

3.6.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get packets must fit within the 255 minimum OBEX packet size. This ensures that all devices can interoperate.
- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of such a Get response is three bytes (0x0003).
- The Continue response code with the Final Bit set (0x90), along with the **Length** (0xC3) and **Body** (0x48) headers will be sent in response to the first Get Request packet with the Final Bit set. The Length header is 5 bytes and will contain the length of the object to be transferred (10240 bytes or 0x00002800). The Body header has a length of 43 bytes (0x002B) and will contain the first 40 bytes of the object being transferred. The length of this Get response is 51 bytes (0x0033).
- **50** additional Get Response packets will be sent with the remainder of the Body header data.
- The last Get Response packet has the Success response code with the Final Bit set (0xA0).
- **10240 bytes** of object data will be sent in response to any **Name** header sent in the Get operation. The first packet with body header data will contain 40 bytes of object data, while the next 50 packets will contain 204 bytes each.

- All subsequent Get Response packets will contain a properly formed **Body** (0x48) header with the exception of the last Get packet that contains an **End Of Body** (0x49) header instead. The Body/End Of Body header will contain 204 bytes of random object data. The length of this header is 207 bytes (0x00CF). The overall length of each Get Response is 210 bytes (0x00D2).
- No other headers are included.

The OBEX Server will also transmit an Abort Response if an Abort Request is received. The following must occur:

- The Success response code with the Final Bit set (0xA0) will be sent.
- The Abort Response will have a length of three bytes (0x0003).

See **A.6.2** for a complete diagram of the event sequence chart.

3.6.3.3 *Test Condition*

If support for the Abort or Get PDU is not present, this test may be **SKIPPED**.

3.6.3.4 *Expected Outcome*

3.6.3.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client.

The OBEX Client transmits an Abort Request. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set.

The Get Operation is successfully canceled, with no further Get Request packets being sent.

3.6.3.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Abort Request does not contain acceptable values.

The Get Operation is not successfully canceled.

3.6.3.4.3 Inconclusive Verdict

The OBEX Client does not transmit an Abort Request.

3.6.3.5 *Notes*

Various headers may be transmitted with the Get Requests or the Abort Request; these do not affect the result of the test.

3.6.4 Test C-A-4: Immediate Get Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Get operation. The Abort Request is issued with one outstanding Get Request, without waiting for the subsequent Get Response packet from the OBEX Server. This behavior is valid only with the Abort PDU.

3.6.4.1 *Test Status*

Accepted

3.6.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Get packets must fit within the 255 minimum OBEX packet size. This ensures that all devices can interoperate.

- All Get Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Get Request packets without the Final Bit set. The length of such a Get response is three bytes (0x0003).
- The Continue response code with the Final Bit set (0x90), along with the **Length** (0xC3) and **Body** (0x48) headers will be sent in response to the first Get Request packet with the Final Bit set. The Length header is 5 bytes and will contain the length of the object to be transferred (10240 bytes or 0x00002800). The Body header has a length of 43 bytes (0x002B) and will contain the first 40 bytes of the object being transferred. The length of this Get response is 51 bytes (0x0033).
- **50** additional Get Response packets will be sent with the remainder of the Body header data.
- The last Get Response packet has the Success response code with the Final Bit set (0xA0).
- **10240 bytes** of object data will be sent in response to any **Name** header sent in the Get operation. The first packet with body header data will contain 40 bytes of object data, while the next 50 packets will contain 204 bytes each.
- All subsequent Get Response packets will contain a properly formed **Body** (0x48) header with the exception of the last Get packet that contains an **End Of Body** (0x49) header instead. The Body/End Of Body header will contain 204 bytes of random object data. The length of this header is 207 bytes (0x00CF). The overall length of each Get Response is 210 bytes (0x00D2).
- Each Get Response will wait **500ms** before sending the next response in order to give the OBEX Client adequate time to abort the operation before the next Get response is sent.
- No other headers are included.

The OBEX Server will also transmit an Abort Response if an Abort Request was received. The Abort Request is expected to be received immediately after the Get Request, but before the Get Response is issued. The following must occur:

- The Success response code with the Final Bit set (0xA0) will be sent.
- The Abort Response will have a length of three bytes (0x0003).

See **A.6.2** for a complete diagram of the event sequence chart.

3.6.4.3 *Test Condition*

If support for the Abort or Get PDU is not present, this test may be **SKIPPED**.

3.6.4.4 *Expected Outcome*

3.6.4.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client.

The OBEX Client transmits an Abort Request. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set.
- The Abort is issued while a Get command is outstanding. This violates the typical OBEX command/response order, but is allowed for OBEX Abort operations only.

The Get Operation is successfully canceled, with no further Get Request packets being sent.

3.6.4.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Abort Request does not contain acceptable values.

The Get Operation is not successfully canceled.

3.6.4.4.3 Inconclusive Verdict

The OBEX Client does not transmit an Abort Request.

The Abort Request is issued in proper OBEX command/response order instead of while a Get command is outstanding.

3.6.4.5 *Notes*

Various headers may be transmitted with the Abort Request; these do not affect the result of the test.

3.7 *OBEX Session PDU*

3.7.1 Test C-S-1: Create Session Operation

Demonstrates the transmission of an OBEX Create Session operation.

3.7.1.1 *Test Status*

Accepted

3.7.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Create Session Request.

- The Create Session Request will contain a **Session Parameters** (0x52) header. This header will contain the *Session Opcode* (0x05), *Device Address* (0x00), and *Nonce* (0x01) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Create Session Response.

The OBEX Server transmits a Create Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Create Session Request packet.
- The Create Session Response packet length will be 48 bytes (0x0030).
- The Create Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 45 bytes (0x002D).
- The Session Parameters header must contain the *Device Address* (0x00)*, Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being created. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

See **A.7.1** for a complete diagram of the event sequence chart.

3.7.1.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.1.4 *Expected Outcome*

3.7.1.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain at least the *Session Opcode, Device Address, and Nonce* tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have the Create Session opcode as its value.

The Create Session Operation is successful.

3.7.1.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request.

The Create Session Request does not contain acceptable values.

The Create Session Operation is not successful.

3.7.1.5 *Notes*

Various other headers may be transmitted with the Create Session Request; these do not affect the result of the test.

3.7.2 Test C-S-2: Close Session Operation

Demonstrates the transmission of an OBEX Close Session operation.

3.7.2.1 *Test Status*

Accepted

3.7.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations. No active or suspended reliable OBEX Sessions should exist.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Close Session Request packet.
- The Close Session Response packet length will be 3 bytes (0x0003).
- No headers are sent.

See **A.7.2** for a complete diagram of the event sequence chart.

3.7.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.2.4 *Expected Outcome*

3.7.2.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Close Session Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.

- The Session Parameters header must contain at least the *Session Opcode and Session ID* tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have the Close Session opcode as its value.

The Close Session Operation is successful.

3.7.2.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request

The Create Session Operation is not successful.

The OBEX Client does not transmit a Close Session Request.

The Close Session Request does not contain acceptable values.

The Close Session Operation is not successful.

3.7.2.5 *Notes*

Various other headers may be transmitted with the Create Session Request or Close Session Request; these do not affect the result of the test.

3.7.3 Test C-S-3: Suspend Session Operation

Demonstrates the transmission of an OBEX Suspend Session operation.

3.7.3.1 *Test Status*

Accepted

3.7.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations. No active or suspended reliable OBEX Sessions should exist.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Suspend Session Request packet.
- The Suspend Session Response packet length will be 3 bytes (0x0003).
- No headers are sent.

See **A.7.4** for a complete diagram of the event sequence chart.

3.7.3.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.3.4 *Expected Outcome*

3.7.3.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Suspend Session Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.

- The Session Parameters header must contain at least the *Session Opcode* tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have the Suspend Session opcode as its value.

The Suspend Session Operation is successful.

3.7.3.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request

The Create Session Operation is not successful.

The OBEX Client does not transmit a Suspend Session Request.

The Suspend Session Request does not contain acceptable values.

The Suspend Session Operation is not successful.

3.7.3.5 *Notes*

Various other headers may be transmitted with the Create Session Request or Suspend Session Request; these do not affect the result of the test.

3.7.4 Test C-S-4: Resume Session Operation

Demonstrates the transmission of an OBEX Resume Session operation.

3.7.4.1 *Test Status*

Accepted

3.7.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations. No active or suspended reliable OBEX Sessions should exist.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Suspend Session Request packet.
- The Suspend Session Response packet length will be 3 bytes (0x0003).
- No headers are sent.

The OBEX Server receives a Resume Session Request.

- The Resume Session Request will contain a **Session Parameters** (0x52) header. This header will contain the *Session Opcode* (0x05), *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Response.

The OBEX Server transmits a Resume Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Resume Session Request packet.
- The Resume Session Response packet length will be 51 bytes (0x0033).
- The Resume Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x0030).
- The Session Parameters header must contain the *Device Address* (0x00)*, Nonce* (0x01), *Session ID* (0x02), and *Next Sequence Number* (0x03) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a

16-byte length (0x10) and contain a 16-byte value describing the session being resumed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce"). The *Next Sequence Number* will have a one-byte length (0x01) and contain the value of the next sequence number expected (0x00 in this case).

- No other headers are sent.

See **A.7.5** for a complete diagram of the event sequence chart.

3.7.4.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.4.4 *Expected Outcome*

3.7.4.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Suspend Session Request.

The OBEX Client receives a Suspend Session Response.

The OBEX Client transmits a Resume Session Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain at least the *Session Opcode, Device Address, Nonce, and Session ID* tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have the Resume Session opcode as its value.

The Resume Session Operation is successful.

3.7.4.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request

The Create Session Operation is not successful.

The OBEX Client does not transmit a Suspend Session Request

The Suspend Session Operation is not successful.

The OBEX Client does not transmit a Resume Session Request.

The Resume Session Request does not contain acceptable values.

The Resume Session Operation is not successful.

3.7.4.5 *Notes*

Various other headers may be transmitted with the Create Session Request, Suspend Session Request, or Resume Session Request; these do not affect the result of the test.

3.7.5 Test C-S-5: Unexpected Resume Session Operation

Demonstrates the transmission of an OBEX Resume Session operation. The resume operation is initiated after the OBEX transport connection is disconnected unexpectedly.

3.7.5.1 *Test Status*

Accepted

3.7.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations. No active or suspended reliable OBEX Sessions should exist.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX transport is disconnected. The reliable session should be automatically suspended.

The OBEX transport connection is reestablished.

The OBEX Client will resume the reliable OBEX session. See **3.7.4.2** for the full description of this process.

See **A.7.6** for a complete diagram of the event sequence chart.

3.7.5.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.5.4 *Expected Outcome*

3.7.5.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

The OBEX transport is disconnected. The reliable session should be automatically suspended.

The OBEX transport connection is reestablished.

The OBEX Client transmits a Resume Session Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain at least the *Session Opcode, Device Address, Nonce, and Session ID* tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have the Resume Session opcode as its value.

The Resume Session Operation is successful.

No operation is resumed, since the transport disconnection did not interrupt an active operation.

3.7.5.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request

The Create Session Operation is not successful.

The OBEX Client does not transmit a Resume Session Request.

The Resume Session Request does not contain acceptable values.

The Resume Session Operation is not successful.

An unknown operation is attempted to be resumed.

3.7.5.5 *Notes*

Various other headers may be transmitted with the Create Session Request or Resume Session Request; these do not affect the result of the test.

3.7.6 Test C-S-6: Set Session Timeout Operation

Demonstrates the transmission of a Session Request PDU to set the suspend session timeout for a reliable OBEX session.

3.7.6.1 *Test Status*

Accepted

3.7.6.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations. No active or suspended reliable OBEX Sessions should exist.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX Server receives a Set Session Timeout Request.

The OBEX Server transmits a Set Session Timeout Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Set Session Timeout Request packet.
- The Set Session Timeout Response packet length will be 3 bytes (0x0003).
- No headers are sent.

See **A.7.7** for a complete diagram of the event sequence chart.

3.7.6.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.7.6.4 *Expected Outcome*

3.7.6.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Set Session Timeout Request. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain at least the *Session Opcode* tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have the Set Timeout opcode as its value.

The Set Session Timeout Operation is successful.

3.7.6.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request

The Create Session Operation is not successful.

The OBEX Client does not transmit a Set Session Timeout Request.

The Set Session Timeout Request does not contain acceptable values.

The Set Session Timeout Operation is not successful.

3.7.6.5 *Notes*

Various other headers may be transmitted with the Create Session Request or Set Session Timeout Request; these do not affect the result of the test.

3.8 *Server Reject*

3.8.1 Test C-SR-1: Server Reject Put Operation

Demonstrate the successful handling of a server rejection of a multi-packet Put operation. This should illustrate a client send of object body, responded to with a non-successful response code.

3.8.1.1 *Test Status*

Accepted

3.8.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Response(s) is/are transmitted by the OBEX Server. The following must occur:

- The Forbidden response code with the Final Bit set (0xC3) is sent in response to the first Put Request packet with a **Body** header containing object body data.
- The Continue response code with the Final Bit set (0x90) is sent in response to every other Put Request.
- The length of each Put Response is three bytes (0x0003).
- No headers are included.

See **A.9.1** for a complete diagram of the event sequence chart.

3.8.1.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

3.8.1.4 *Expected Outcome*

3.8.1.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following must occur:

- The Final Bit must not be set.
- The first packet to contain object data must use a **Body** header to include the portion of object data being transferred. This can be achieved by sending an object larger than the OBEX Packet size. Doing so will prevent the Client from sending all of the object data in an **End Of Body** header.

The Put operation is successfully canceled after the receipt of an unsuccessful response code from the OBEX Server.

3.8.1.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Operation is not canceled.

3.8.1.4.3 Inconclusive Verdict

No object is available which will require more than one OBEX packet to send.

3.8.1.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.8.2 Test C-SR-2: Server Reject Get Operation

Demonstrate the successful handling of a server rejection of a multi-packet Get operation. This should illustrate the client's request for object body receiving a non-successful response code.

3.8.2.1 *Test Status*

Accepted

3.8.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

Get Response(s) is/are transmitted by the OBEX Server. The following must occur:

- The Forbidden response code with the Final Bit set (0xC3) is sent in response to the first Get Request packet with the Final Bit set.
- The Continue response code with the Final Bit set (0x90) is sent in response to every other Get Request.
- The length of each Get Response is three bytes (0x0003).
- No headers are included.

See **A.9.2** for a complete diagram of the event sequence chart.

3.8.2.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

3.8.2.4 *Expected Outcome*

3.8.2.4.1 Pass Verdict

Get Request(s) is/are transmitted by the Client. The following must occur:

- The first packet to request object data must set the Final Bit.
- An unsuccessful response code will be received in response to the first packet to request object data.

The Get operation is successfully canceled after the receipt of an unsuccessful response code from the OBEX Server.

3.8.2.4.2 Fail Verdict

The OBEX Client does not transmit the Get Requests.

The Get Operation is not canceled.

3.8.2.5 *Notes*

Various other headers may be transmitted with the Get Requests; these do not affect the result of the test.

3.9 *Spurious Transport Disconnects*

3.9.1 Test C-TD-1: Transport Disconnect During Put Operation

Demonstrates a transport disconnection during a reliable OBEX session while an OBEX Put operation is in progress. The transport disconnection occurs while the Put operation is in the process of transferring the object data. Resuming the reliable OBEX Session should resume the OBEX Put operation.

3.9.1.1 *Test Status*

Accepted

3.9.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Create Session Request packet.
- The Create Session Response packet length will be 48 bytes (0x0030).

- The Create Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 45 bytes (0x002D).
- The Session Parameters header must contain the *Device Address* (0x00)*, Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being created. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

The OBEX Server receives Put Request(s) that may contain all headers except Body headers. The OBEX Server will transmit a Put Response for each Put Request that matches this requirement. The following must occur:

- The Continue response code with the Final Bit set (0x90) will be sent.
- The Put Response packet has a length of 5 bytes (0x0005).
- The Put Response packet will contain a **Session-Sequence-Number** (0x93) header. This header will contain the next expected sequence number, which is one value larger than the last received sequence number value. The length of this header is 2 bytes and will contain a 1-byte sequence number value.

The OBEX Server receives the first Put Request that contains Body headers and initiates a **Transport Disconnection**. This encompasses disconnecting all lower layer protocols below OBEX (TinyTP, IrLMP, and IrLAP).

The OBEX Client restores the Transport Connection.

The OBEX Server receives a Resume Session Request.

- The Resume Session Request will contain a **Session Parameters** (0x52) header. This header will contain the *Session Opcode* (0x05), *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Response.

The OBEX Server transmits a Resume Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Resume Session Request packet.
- The Resume Session Response packet length will be 51 bytes (0x0033).
- The Resume Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x0030).
- The Session Parameters header must contain the *Device Address* (0x00)*, Nonce* (0x01), *Session ID* (0x02), and *Next Sequence Number* (0x03) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being resumed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce"). The *Next Sequence Number* will have a one-byte length (0x01) and contain the value of the next sequence number expected (the sequence number from the last received Put Request).
- No other headers are sent.

The OBEX Server will receive a Put Request retransmission. The resending of this packet will cause the OBEX Server to send the Put Response that was never sent. The following must occur:

- All Put Response packets will have the Final Bit set.

- All Put Response packets have a length of 5 bytes (0x0005).
- All response packets will contain a **Session-Sequence-Number** (0x93) header. This header will contain the next expected sequence number, which is one value larger than the last received sequence number value. The length of this header is 2 bytes and will contain a 1-byte sequence number value.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

See **A.13.1** for a complete diagram of the event sequence chart.

3.9.1.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.9.1.4 *Expected Outcome*

3.9.1.4.1 Pass Verdict

The OBEX reliable session is started successfully.

The Put operation is started successfully.

The OBEX Server disconnects the OBEX transport connection.

The OBEX Client reestablishes the transport connection.

The OBEX Client transmits an OBEX Resume Session Request.

The Put operation is resumed and succeeds.

The Combination of Body and End Of Body headers should equal the size of the object sent by the OBEX Client.

3.9.1.4.2 Fail Verdict

The OBEX Client does not reestablish the transport connection.

The OBEX Client does not transmit an OBEX Resume Session Request.

The Put operation is not resumed.

The Put operation is not successful.

The Combination of Body and End Of Body headers does not equal the size of the object sent by the OBEX Client.

3.9.1.5 *Notes*

Various other headers may be transmitted with the Put Request(s); these do not affect the result of the test.

3.9.2 Test C-TD-2: Transport Disconnect During Get Operation

Demonstrates a transport disconnection during a reliable OBEX session while an OBEX Get operation is in progress. The transport disconnection occurs while the Get operation is in the process of transferring the object data. Resuming the reliable OBEX Session should resume the OBEX Get operation.

3.9.2.1 *Test Status*

Accepted

3.9.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Create Session Request packet.
- The Create Session Response packet length will be 48 bytes (0x0030).
- The Create Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 45 bytes (0x002D).
- The Session Parameters header must contain the *Device Address* (0x00)*, Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being created. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

The OBEX Server receives Get Request(s) that contain headers but do not have the Final Bit set. The OBEX Server will transmit a Get Response for each Get Request that matches this requirement. The following must occur:

- The Continue response code with the Final Bit set (0x90) will be sent.
- The Get Response packet has a length of 5 bytes (0x0005).
- The Get Response packet will contain a **Session-Sequence-Number** (0x93) header. This header will contain the next expected sequence number, which is one value larger than the last received sequence number value. The length of this header is 2 bytes and will contain a 1-byte sequence number value.

The OBEX Server receives a Get Request that has the Final Bit set. The following must occur:

- The Continue response code with the Final Bit set (0x90) will be sent.
- The Get Response packet has a length of 38 bytes (0x0026).
- The Get Response packet will contain a **Session-Sequence-Number** (0x93) header. This header will contain the next expected sequence number, which is one value larger than the last received sequence number value. The length of this header is 2 bytes and will contain a 1-byte sequence number value.
- The **Length** (0xC3) header will be sent. The length of the header is 5 bytes and will contain the length of the object to be transferred (50 bytes or 0x00000032).
- The **Body** (0x48) header will be sent. The Body header will contain 25 bytes of random data forming the first half of the object being sent. The length of this header is 28 bytes (0x001C).

The OBEX Server receives another Get Request with the Final Bit set and initiates a **Transport Disconnection**. This encompasses disconnecting all lower layer protocols below OBEX (TinyTP, IrLMP, and IrLAP).

The OBEX Server receives a Get Request.

The OBEX Server initiates a **Transport Disconnection**. This encompasses disconnecting all lower layer protocols below OBEX (TinyTP, IrLMP, and IrLAP).

The OBEX Client restores the Transport Connection.

The OBEX Server receives a Resume Session Request.

- The Resume Session Request will contain a **Session Parameters** (0x52) header. This header will contain the *Session Opcode* (0x05), *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Response.

The OBEX Server transmits a Resume Session Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Resume Session Request packet.
- The Resume Session Response packet length will be 51 bytes (0x0033).
- The Resume Session Response contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x0030).
- The Session Parameters header must contain the *Device Address* (0x00), *Nonce* (0x01), *Session ID* (0x02), and *Next Sequence Number* (0x03) tag/length/value triplets. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being resumed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce"). The *Next Sequence Number* will have a one-byte length (0x01) and contain the value of the next sequence number expected (the sequence number from the Get Request).
- No other headers are sent.

The OBEX Server will receive a Get Request retransmission. The resending of this packet will cause the OBEX Server to send the Get Response that was never sent. The only header included in the Get Request will be the Session-Sequence-Number header. The following must occur:

- The Get Response packet will have the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent.
- The length of the Get Response is 33 bytes (0x0021).
- The Get Response packet will contain a **Session-Sequence-Number** (0x93) header. This header will contain the next expected sequence number, which is one value larger than the last received sequence number value. The length of this header is 2 bytes and will contain a 1-byte sequence number value.
- The **End Of Body** (0x49) header will be sent. The End Of Body header will contain 25 bytes of random data forming the object being sent. The length of this header is 28 bytes (0x001C).

Get Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Get Response packets for every Get Request packet received.

See **A.13.2** for a complete diagram of the event sequence chart.

3.9.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.9.2.4 *Expected Outcome*

3.9.2.4.1 Pass Verdict

The OBEX reliable session is started successfully.

The Get operation is started successfully.

The OBEX Server disconnects the OBEX transport connection.

The OBEX Client reestablishes the transport connection.

The OBEX Client transmits an OBEX Resume Session Request.

The Get operation is resumed and succeeds.

The Get Operation is successful. The following are true:

- A 50-byte object should have been received and stored according to the name provided by the OBEX Client.

3.9.2.4.2 Fail Verdict

The OBEX Client does not reestablish the transport connection.

The OBEX Client does not transmit an OBEX Resume Session Request.

The Get operation is not resumed.

The Get operation is not successful.

3.9.2.5 *Notes*

Various other headers may be transmitted with the Get Request(s); these do not affect the result of the test.

3.10 *OBEX Headers*

3.10.1 Test C-H-1: Tiny TP Split Header

Demonstrate that OBEX headers broken over Tiny TP packet boundaries are properly handled. This should be demonstrated by sending a **PUT** request with the **Body/End Of Body** header split into two TinyTP packets. The device should properly receive the complete object.

3.10.1.1 *Test Status*

Accepted

3.10.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

During the transport connection, during the UA frame, a 64-byte packet size should be requested. This will ensure that the TinyTP packet size is restricted to 64 bytes. This limits the OBEX Client data per TinyTP packet to 58 bytes (64 bytes – 2 (IrLMP) – 1 (TinyTP) – 3 (OBEX) = 58 bytes).

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set and a length of three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

3.10.1.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

3.10.1.4 *Expected Outcome*

3.10.1.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Packet Length indicates the correct size.
- The Put Request PDU contains a properly formed **Body/End Of Body** header that is at least 59 bytes in length, as this will assure two TinyTP packets are used to transmit this header. This is guaranteed with the requirement that the TinyTP packet size is 64 bytes.
- Combination of Body and End Of Body headers should meet or exceed the 59-byte object size requirement.

The Put Operation is successful.

3.10.1.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Operation is not successful

The Put operation does not handle the use of the **Body/End Of Body** headers properly.

3.10.1.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.10.2 Test C-H-2: One-Byte Headers

Demonstrate that one-byte style OBEX headers are properly formed when transmitted and properly handled when received. Since OBEX defines only one one-byte header, the Session-Sequence-Number header, the only way to guarantee that both the Client and Server will send a one-byte header is to establish a reliable OBEX session and then perform an operation. After the reliable session has been created, an OBEX Connect operation should be issued.

3.10.2.1 *Test Status*

Accepted

3.10.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **3.7.1.2** for the full description of this process.

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet.
- The Connect Response packet length is seven bytes (0x0009).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Response contains a properly formed **Session-Sequence-Number** (0x93) header as the first and only header in the packet. The length of this header is two bytes. The value of this header will include the next sequence number (0x01).
- No other headers are included.

See **A.10.1** for a complete diagram of the event sequence chart.

3.10.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

3.10.2.4 *Expected Outcome*

3.10.2.4.1 Pass Verdict

The OBEX Client transmits a Create Session Request.

The OBEX Client receives a Create Session Response.

Create Session operation is successful.

The OBEX Client transmits Connect Request. The following are true:

- The Connect Request contains a properly formed **Session-Sequence-Number** (0x93) header as the first header in the packet. The length of this header is two bytes. The value of this header will include the current sequence number (0x00).

The OBEX Client receives a Connect Response.

OBEX Connect operation is successful.

3.10.2.4.2 Fail Verdict

The OBEX Client does not transmit a Create Session Request.

Create Session operation is not successful.

The OBEX Client does not transmit a Connect Request with a properly formed Session-Sequence-Number header.

OBEX Connect operation is not successful.

3.10.2.5 *Notes*

Various other headers may be transmitted with the Create Session Request or Connect Request; these do not affect the result of the test.

3.10.3 Test C-H-3: Four-Byte Headers

Demonstrate that four-byte style OBEX headers are properly formed when transmitted and properly handled when received. This should be demonstrated by sending a **PUT** request with a **Length** header and receiving a **PUT** response with a **Count** header.

3.10.3.1 *Test Status*

Accepted

3.10.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set. The packet length of these responses is three bytes (0x0003).
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set. The packet length of this response is eight bytes (0x0008).
- The final Put response will contain a four-byte **Count** (0xC0) header. The length of the header is five bytes (0x0005). Its contents will be 0x00000001.
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

3.10.3.3 *Test Condition*

Support for four byte headers is **REQUIRED**.

3.10.3.4 *Expected Outcome*

3.10.3.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- A Put Request PDU contains a properly formed four-byte **Length** header.

The Put Operation is successful.

- The Put operation successfully handled sending a four-byte **Length** header and receiving a four-byte **Count** header.

3.10.3.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Operation is not successful

The Put operation does not handle the use of the **Length** and **Count** headers properly.

3.10.3.4.3 Inconclusive Verdict

A Put Request does not include a **Length** header. Some devices might not include a Length header, as it is an optional header for a normal Put operation.

3.10.3.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.10.4 Test C-H-4: Byte Sequence Headers

Demonstrate that byte sequence style OBEX headers are properly formed when transmitted and properly handled when received. This should be demonstrated by sending a **PUT** request with an **End Of Body** header and receiving a **PUT** response with an **HTTP** header.

3.10.4.1 *Test Status*

Accepted

3.10.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set. The packet length of these responses is three bytes (0x0003).
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set. The packet length of this response is fourteen bytes (0x000E).
- The final Put response will contain an **HTTP** (0x47) byte sequence header. The length of the header is eleven bytes (0x000B). Its contents will be “Testing” (with null-termination) displayed in hexadecimal as follows:

  54657374 696E6700
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

3.10.4.3 *Test Condition*

Support for Byte Sequence headers is **REQUIRED**.

3.10.4.4 *Expected Outcome*

3.10.4.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- A Put Request PDU contains a properly formed **End Of Body** byte sequence header.

The Put Operation is successful.

- The Put operation successfully handled sending an **End Of Body** byte sequence header and receiving an **HTTP** byte sequence header.

3.10.4.4.2 Fail Verdict

The OBEX Client does not transmit a Put Request with an **End Of Body** byte sequence header.

The Put Operation is not successful

The Put operation does not handle the use of the **End Of Body** and **HTTP** headers properly.

3.10.4.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.10.5 Test C-H-5: Unicode Headers

Demonstrate that UNICODE style OBEX headers are properly formed when transmitted and properly handled when received. This includes verification that the header data is null-terminated. This should be demonstrated by sending a **PUT** request with a **Name** header and receiving a **PUT** response with a **Description** header.

3.10.5.1 *Test Status*

Accepted

3.10.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set. The packet length of this response is three bytes (0x0003).
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set. The packet length of this response is twenty-two bytes (0x0016).
- The final Put response will contain a null-terminated, Unicode **Description** (0x05) header. The length of the header is nineteen bytes (0x0013) including null-termination. Its contents will be "Testing" displayed in Unicode as follows:

  00540065 00730074 0069006E 00670000 - All values in Hexadecimal
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

3.10.5.3 *Test Condition*

Support for Unicode headers is **REQUIRED**.

3.10.5.4 *Expected Outcome*

3.10.5.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- A Put Request PDU contains a properly formed null-terminated, Unicode **Name** header.

The Put Operation is successful.

- The Put operation successfully handled sending a null-terminated, Unicode **Name** header and receiving a null-terminated, Unicode **Description** header.

3.10.5.4.2 Fail Verdict

The OBEX Client does not transmit a Put Request with a null-terminated, Unicode **Name** header.

The Put Operation is not successful

The Put operation does not handle the use of the **Name** and **Description** headers properly.

3.10.5.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.11 *OBEX Authentication*

3.11.1 Test C-AU-1: Authenticate Client Connection

Demonstrates that the OBEX Client can successfully authenticate the OBEX Server during an OBEX Connect operation.

3.11.1.1 *Test Status*

Accepted

3.11.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Connect Request. The following must occur:

- An **Authentication Challenge** (0x4D) header is received containing at least the 16-byte Nonce value.

The OBEX Server transmits a Connect Response. The following must occur:

- The Success response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet.
- The Connect Response packet length will be 255 bytes or less, but will vary based on the size of the Authentication Response header.
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect response will contain an **Authenticate Response** (0x4E) byte sequence header in response to the Authentication Challenge header received from the OBEX Client. The length of the header will vary based on the tag/length/value triplets being returned.

  The contents of this header will include the following:

  Request Digest – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. The 16-byte Nonce from the Authentication Challenge header in the Connect Request, followed by a colon, and the password used on the OBEX Client are the values passed into the MD5 algorithm to generate the request digest (i.e. Request Digest = MD5(Nonce ":" Password)).

  The format for this field is as follows:

  ```
  00 10 <Request-Digest>                    - Hexadecimal values
  ```

  UserId – this value is optionally included based on the options field received in the Authentication Challenge header in the Connect Request. If the options field has the first bit set, the userId is required. In order to send this value, you will need to know the userId requested by the OBEX Client and provide it in this field.

  The format for this field is as follows:

  ```
  01 <varies, up to 20 bytes> <UserId>      - Hexadecimal values
  ```

  Nonce – this value is the 16-byte Nonce received in Authentication Challenge header in the Connect Request.

  The format for this field is as follows:

  ```
  02 10 <Nonce>                             - Hexadecimal values
  ```

- No other headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

3.11.1.3 *Test Condition*

If support for the Connect PDU or OBEX Authentication is not present, this test may be **SKIPPED**.

3.11.1.4 *Expected Outcome*

3.11.1.4.1 Pass Verdict

The OBEX Client transmits a Connect Request. The following are true:

- The Connect Operation initiates OBEX Client Authentication.
- The OBEX Client device should prompt the user to enter a password (and optionally a userId), or at least inform the user of the password (and optionally the userId) being used. This password will be used in formulating the Authentication Challenge header.
- The Connect Request contains an **Authenticate Challenge** byte sequence header. This header must contain a properly formed **Nonce** field and can optionally contain the **Options** and **Realm** fields.

The Connect Operation succeeds.

The OBEX Client Authentication procedure succeeds.

3.11.1.4.2 Fail Verdict

The OBEX Client does not transmit a Connect Request.

The Connect Request does not contain acceptable values.

The Connect Operation does not succeed.

The OBEX Client Authentication procedure does not succeed.

3.11.1.4.3 Inconclusive Verdict

The Connect Operation cannot initiate OBEX Authentication.

3.11.1.5 *Notes*

Various other headers may be transmitted with the Connect Request; these do not affect the result of the test.

3.11.2 Test C-AU-2: Authenticate Client Operation

Demonstrates that the OBEX Client can successfully authenticate the OBEX Server for an OBEX Put operation.

3.11.2.1 *Test Status*

Accepted

3.11.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives the first Put Request. The following must occur:

- An **Authentication Challenge** (0x4D) header is received containing at least the 16-byte Nonce value.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set.
- The packet length of all responses (except the first) is three bytes (0x0003).
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

- The first Put Response will contain an **Authenticate Response** (0x4E) byte sequence header in response to the Authentication Challenge header received from the OBEX Client. The length of the header will vary based on the tag/length/value triplets being returned.

  The contents of this header will include the following:

  Request Digest – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. The 16-byte Nonce from the Authentication Challenge header in the Put Request, followed by a colon, and the password used on the OBEX Client are the values passed into the MD5 algorithm to generate the request digest (i.e. Request Digest = MD5(Nonce ":" Password)).

  The format for this field is as follows:

  ```
  00 10 <Request-Digest>                    - Hexadecimal values
  ```

  UserId – this value is optionally included based on the options field received in the Authentication Challenge header in the Put Request. If the options field has the first bit set, the userId is required. In order to send this value, you will need to know the userId requested by the OBEX Client and provide it in this field.

  The format for this field is as follows:

  ```
  01 <varies, up to 20 bytes> <UserId>      - Hexadecimal values
  ```

  Nonce – this value is the 16-byte Nonce received in Authentication Challenge header in the Put Request.

  The format for this field is as follows:

  ```
  02 10 <Nonce>                             - Hexadecimal values
  ```

- No other headers are included.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

See **A.5.1** for a complete diagram of the event sequence chart.

3.11.2.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

3.11.2.4 *Expected Outcome*

3.11.2.4.1 Pass Verdict

Put Request(s) is/are transmitted by the Client. The following are true:

- The Put Operation initiates OBEX Client Authentication.
- The OBEX Client device should prompt the user to enter a password (and optionally a userId), or at least inform the user of the password (and optionally the userId) being used. This password will be used in formulating the Authentication Challenge header.
- The first Put Request contains an **Authenticate Challenge** byte sequence header. This header must contain a properly formed **Nonce** field and can optionally contain the **Options** and **Realm** fields.

The Put Operation is successful.

The OBEX Client Authentication procedure succeeds.

3.11.2.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

The OBEX Client Authentication procedure does not succeed.

3.11.2.4.3 Inconclusive Verdict

The Put Operation cannot initiate OBEX Authentication.

3.11.2.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.11.3 Test C-AU-3: Authenticate Server Connection

Demonstrates that the OBEX Server can successfully authenticate the OBEX Client during an OBEX Connect operation.

3.11.3.1 *Test Status*

Accepted

3.11.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following must occur:

- The Unauthorized response code with the Final Bit set (0xC1) is sent in response to the Connect Request packet. The packet length of this response is 31 bytes (0x001F).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Response will contain an **Authenticate Challenge** (0x4D) byte sequence header. The length of the header is 24 bytes (0x0018).

  The contents of this header will include the following two tag/length/value triplets:

  Nonce – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. A 4-byte unique time-stamp, followed by a colon, and a private key known only to the sender are the values passed into the MD5 algorithm to generate the nonce (i.e. Nonce = MD5(time-stamp ":" private-key)).

  The format for this field is as follows:

00 10 <nonce>	- Hexadecimal values

  Options – this value indicates that neither of the two defined authentication options is required.

  The format for this field is as follows:

01 01 00	- Hexadecimal values

- No other headers are included.

The OBEX Server receives a Connect Request. The following must occur:

- An **Authentication Response** header is received with a request digest.

The OBEX Server transmits a Connect Response. The following must occur:

- The OBEX Server's 16-byte nonce, followed by a colon, and the password used on the OBEX Client should be passed into the MD5 algorithm and compared against the request digest received in the Authentication Response header. If the request digest matches the MD5 result, the **Success** response code with the Final Bit set (0xA0) is sent in response to the Connect Request packet, otherwise, the **Unauthorized** response code with the Final Bit set (0xC1) is returned.

See **A.1.2** for a complete diagram of the event sequence chart.

3.11.3.3 *Test Condition*

If support for the Connect PDU or OBEX Authentication is not present, this test may be **SKIPPED**.

3.11.3.4 *Expected Outcome*

3.11.3.4.1 Pass Verdict

The OBEX Client transmits a Connect Request.

The Connect Operation is aborted with the Unauthorized reason code due to OBEX Authentication being initiated by the OBEX Server.

The OBEX Client reissues a Connect Request. The following are true:

- The OBEX Client device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Response header.
- The Connect Request contains an **Authenticate Response** byte sequence header in response to the received Authenticate Challenge header from the OBEX Server. This header must contain a properly formed **Request Digest** field and can optionally contain the **UserId** and **Nonce** fields.

The Connect Operation is successful.

3.11.3.4.2 Fail Verdict

The OBEX Client does not transmit the Connect Requests.

The Connect Requests do not contain acceptable values.

The Connect Operation is not successful.

3.11.3.4.3 Inconclusive Verdict

3.11.3.5 *The Connect Operation cannot respond to the OBEX Authentication request.Notes*

Various other headers may be transmitted with the Connect Requests; these do not affect the result of the test.

3.11.4 Test C-AU-4: Authenticate Server Operation

Demonstrates that the OBEX Server can successfully authenticate the OBEX Client for an OBEX Put operation.

3.11.4.1 *Test Status*

Accepted

3.11.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following must occur:

- The Unauthorized response code with the Final Bit set (0xC1) is sent in response to the Put Request packet. The packet length of this response is 27 bytes (0x001B).
- The Put Response will contain an **Authenticate Challenge** (0x4D) byte sequence header. The length of the header is 24 bytes (0x0018).

  The contents of this header will include the following two tag/length/value triplets:

  Nonce – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. A 4-byte unique time-stamp, followed by a colon, and a private key known only to the sender are the values passed into the MD5 algorithm to generate the nonce (i.e. Nonce = MD5(time-stamp ":" private-key)).

  The format for this field is as follows:

  ```
  00 10 <nonce>                    - Hexadecimal values
  ```

  Options – this value indicates that neither of the two defined authentication options is required.

The format for this field is as follows:

01 01 00 - Hexadecimal values

- No other headers are included.

The OBEX Server receives a Put Request. The following must occur:

- An **Authentication Response** header is received with a request digest.

Put Responses(s) is/are transmitted by the OBEX Server. The following must occur:

- All Put Response packets will have the Final Bit set. The packet length of each response is three bytes (0x0003).
- For the first Put response only, the OBEX Server's 16-byte nonce, followed by a colon, and the password used on the OBEX Client should be passed into the MD5 algorithm and compared against the request digest received in the Authentication Response header. If the request digest does not match the MD5 result, the **Unauthorized** response code with the Final Bit set (0xC1) is returned in place of the normal response codes outlined below.
- Continue response codes with the Final Bit set (0x90) will be sent in response to all received Put Request packets without the Final Bit set.
- The Success response code with the Final Bit set (0xA0) will be sent in response to the Put Request packet with the Final Bit set.

Put Requests(s) is/are received by the OBEX Server. The OBEX Server will respond with Put Response packets for every Put Request packet received.

See **A.5.2** for a complete diagram of the event sequence chart.

3.11.4.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

3.11.4.4 *Expected Outcome*

3.11.4.4.1 Pass Verdict

The OBEX Client transmits a Put Request.

The Put Operation is aborted with the Unauthorized reason code due to OBEX Authentication being initiated by the OBEX Server.

The OBEX Client reissues a Put Request. The following are true:

- The OBEX Client device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Response header.
- The Put Request contains an **Authenticate Response** byte sequence header in response to the received Authenticate Challenge header from the OBEX Server. This header must contain a properly formed **Request Digest** field and can optionally contain the **UserId** and **Nonce** fields.

The Put Operation is successful.

3.11.4.4.2 Fail Verdict

The OBEX Client does not transmit the Put Requests.

The Put Requests do not contain acceptable values.

The Put Operation is not successful.

3.11.4.4.3 Inconclusive Verdict

The Put Operation cannot respond to the OBEX Authentication request.

3.11.4.5 *Notes*

Various other headers may be transmitted with the Put Requests; these do not affect the result of the test.

3.12 *Miscellaneous Tests*

3.12.1 Test C-IAS-1: OBEX IAS Query

Verify that the “OBEX” IAS entry is properly requested.

3.12.1.1 *Test Status*

Accepted

3.12.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server responds to the OBEX transport connection. The following will occur:

- Receipt of an IrLAP SNRM Command.
- Transmission of an IrLAP UA Response.
- Receipt of an IrLMP Connect Request. Source LSAP is the IAS Client LSAP (varies). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Transmission of an IrLMP Connect Response. Source LSAP is the IAS Server LSAP (LSAP 0). Destination LSAP is the IAS Client LSAP (varies).
- Receipt of an IrLMP Data packet containing an IrIAS query for the “OBEX” class name. Source LSAP is the IAS Client LSAP (varies). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Transmission of an IrLMP Data packet containing an IrIAS query response. The following are true:
  - Source LSAP is the IAS Server LSAP (LSAP 0)
  - Destination LSAP is the IAS Client LSAP (varies).
  - GetValueByClass IAS response is sent with the Last bit set (0x84).
  - Success response code is returned (0x00).
  - Result count of one is returned (0x0001).
  - Object ID value is returned (0x0000).
  - An Integer value is returned (0x01).
  - A 32-bit signed OBEX LSAP value is returned. The value 0x0000002 will be used.

See **A.12.1** for a complete diagram of the event sequence chart.

3.12.1.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

3.12.1.4 *Expected Outcome*

3.12.1.4.1 Pass Verdict

The OBEX Client initiates a transport connection. This following will occur:

- Successful IrLAP and IrLMP connections.
- Transmission of an IrLMP Data packet with IAS query information. The following will occur:
  - Source LSAP is the IAS Client LSAP (varies).
  - Destination LSAP is the IAS Server (LSAP 0).
  - GetValueByClass IAS query is sent.
  - “OBEX” IAS class name is sent in ASCII format. The length of the class name is 4 bytes.

- "IrDA:TinyTP:LsapSel" IAS attribute is sent in ASCII format. The length of the attribute name is 19 bytes.

The IAS Query is successful with an OBEX LSAP value located.

3.12.1.4.2 Fail Verdict

The OBEX Client fails to transmit an IrLMP Data packet with the IAS query.

The IAS query contains unacceptable values.

The IAS query is unsuccessful.

3.12.1.5 *Notes*

N/A

3.12.2 Test C-TTP-1: Tiny TP Connect

Verify that the Tiny TP Connection is successful. The MaxSduSize parameter must not present in the Tiny TP Connect Request packet in order for the connection to be considered successful.

3.12.2.1 *Test Status*

Accepted

3.12.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server responds to the OBEX transport connection. The following will occur:

- Receipt of an IrLAP SNRM Command.
- Transmission of an IrLAP UA Response.
- Receipt of an IrLMP Connect Request. Source LSAP is the IAS Client LSAP (varies). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Transmission of an IrLMP Connect Response. Source LSAP is the IAS Server LSAP (LSAP 0). Destination LSAP is the IAS Client LSAP (varies).
- Receipt of an IrLMP Data packet containing an IrIAS query for the "OBEX" class name. Source LSAP is the IAS Client LSAP (varies). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Transmission of an IrLMP Data packet containing an IrIAS query response. The following are true:
    - Source LSAP is the IAS Server LSAP (LSAP 0)
    - Destination LSAP is the IAS Client LSAP (varies).
    - GetValueByClass IAS response is sent with the Last bit set (0x84).
    - Success response code is returned (0x00).
    - Result count of one is returned (0x0001).
    - Object ID value is returned (0x0000).
    - An Integer value is returned (0x01).
    - A 32-bit signed OBEX LSAP value is returned. The value 0x0000002 will be used.
- Receipt of an IrLMP Connect Request (TinyTP Connect Request). Source LSAP is the OBEX Client's OBEX LSAP (varies). Destination LSAP is the OBEX Server's advertised OBEX LSAP (LSAP 2 will be used).
- Transmission of an IrLMP Connect Response (TinyTP Connect Response). The following are true:
    - Source LSAP is the advertised OBEX LSAP (LSAP 2 will be used).
    - Destination LSAP is the OBEX Client's OBEX LSAP (varies).
    - The Parameter bit is clear.

See **A.12.2** for a complete diagram of the event sequence chart.

3.12.2.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

3.12.2.4 *Expected Outcome*

3.12.2.4.1 Pass Verdict

The OBEX Client initiates a transport connection. This following will occur:

- Successful IrLAP and IrLMP connections.
- Successful IAS query for the "OBEX" class name.
- Transmission of an IrLMP Connect Request for the OBEX Service (TinyTP Connect Request). The following are true:
    - Source LSAP is the OBEX Client's OBEX LSAP (varies).
    - Destination LSAP is the OBEX Server's OBEX LSAP (LSAP 2 will be used).
    - The Parameter bit is clear.
    - No MaxSduSize parameter exists.

The TinyTP Connection to the OBEX Service is successful.

3.12.2.4.2 Fail Verdict

The OBEX Client fails to transmit an IrLMP Connect Request packet for the OBEX Service (TinyTP Connect Request).

The TinyTP Connect Request contains unacceptable values.

The TinyTP Connection is unsuccessful.

3.12.2.5 *Notes*

N/A

3.12.3 Test C-UP-1: Small Ultra Put

Demonstrate an Ultra Put of a 25-byte object.

3.12.3.1 *Test Status*

Accepted

3.12.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives the Put Request.

The OBEX Server **does not** transmit a Put Response, as a Put operation over Ultra does not follow the standard request/response model used by a full OBEX implementation.

See **A.5.4** for a complete diagram of the event sequence chart.

3.12.3.3 *Test Condition*

If support for Ultra is not present, this test may be **SKIPPED**.

3.12.3.4 *Expected Outcome*

3.12.3.4.1 Pass Verdict

The OBEX Client transmits a Put Request. The following are true:

- The Put Packet Length indicates the correct size, but does not exceed 255 bytes.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.

- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- No Body headers exist.
- The End Of Body header should meet or exceed the 25-byte object size requirement.

3.12.3.4.2 Fail Verdict

The OBEX Client does not transmit the Put Request.

The Put Request does not contain acceptable values.

3.12.3.5 *Notes*

Various other headers may be transmitted with the Put Request; these do not affect the result of the test.

3.12.4 Test C-UP-2: Maximum Ultra Put

Demonstrate an Ultra Put of the maximum sized object supported by the device. If the maximum sized object exceeds 500 bytes, then a 500-byte object should be used.

3.12.4.1 *Test Status*

Accepted

3.12.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives the Put Request.

The OBEX Server **does not** transmit a Put Response, as a Put operation over Ultra does not follow the standard request/response model used by a full OBEX implementation.

See **A.5.4** for a complete diagram of the event sequence chart.

3.12.4.3 *Test Condition*

If support for Ultra is not present, this test may be **SKIPPED**.

3.12.4.4 *Expected Outcome*

3.12.4.4.1 Pass Verdict

The OBEX Client transmits a Put Request. The following are true:

- The Put Packet Length indicates the correct size, but does not exceed 900 bytes.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- No Body headers exist.
- The End Of Body header should not exceed the 500-byte object size maximum.

3.12.4.4.2 Fail Verdict

The OBEX Client does not transmit the Put Request.

The Put Request does not contain acceptable values.

3.12.4.5 *Notes*

Various other headers may be transmitted with the Put Request; these do not affect the result of the test.

3.12.5 Test C-UP-3: Zero Byte Ultra Put

Demonstrate an Ultra Put of a 0-byte object.

3.12.5.1 *Test Status*

Accepted

3.12.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Server receives the Put Request.

The OBEX Server **does not** transmit a Put Response, as a Put operation over Ultra does not follow the standard request/response model used by a full OBEX implementation.

See **A.5.4** for a complete diagram of the event sequence chart.

3.12.5.3 *Test Condition*

If support for Ultra is not present, this test may be **SKIPPED**.

3.12.5.4 *Expected Outcome*

3.12.5.4.1 Pass Verdict

The OBEX Client transmits a Put Request. The following are true:

- The Put Packet Length indicates the correct size, but does not exceed 255 bytes.
- The Put Request PDU contains a properly formed **Name** header indicating the name of the object being sent.
- The Put Request PDU contains a properly formed **End Of Body** header as its final header in the Put operation.
- No Body headers exist.
- The End Of Body header should contain 0 bytes of object data.

3.12.5.4.2 Fail Verdict

The OBEX Client does not transmit the Put Request.

The Put Request does not contain acceptable values.

3.12.5.5 *Notes*

Various other headers may be transmitted with the Put Request; these do not affect the result of the test.

4 OBEX Server Tests

Tests in this section assume the Device Under Test is acting as the OBEX Server and the tester device is acting as the OBEX Client.

4.1 *OBEX Connect PDU*

4.1.1 Test S-C-1: Simple Connect Operation

Demonstrates the transmission of an OBEX Connect operation.

4.1.1.1 *Test Status*

Accepted

4.1.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is seven bytes (0x0007).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- No headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

4.1.1.3 *Test Condition*

OBEX Server behavior for the Connect PDU is **REQUIRED**.

4.1.1.4 *Expected Outcome*

4.1.1.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The entire Connect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Connect Packet Length field indicates the correct size.
- The Protocol field indicates OBEX version 1.0.
- The Flags field has no bits set.
- The Maximum OBEX Packet Length field specifies a size of 255 or above.
- The Success response code is returned.

The Connect Operation is successful.

4.1.1.4.2 Fail Verdict

The OBEX Server does not transmit a Connect Response.

The Connect Response does not contain acceptable values.

The Connect Operation is not successful.

4.1.1.5 *Notes*

Various headers may be transmitted with the Connect Response; these do not affect the result of the test.

4.1.2 Test S-C-2: Simple Directed Connection

Demonstrates the transmission of a Directed OBEX Connect operation.

4.1.2.1 *Test Status*

Accepted

4.1.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length does not exceed 255 bytes.
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Request packet contains a properly formed **Target** (0x46) header. This Target header should describe a valid OBEX service. See the OBEX specification for a listing of the valid OBEX services. The length of this header is a two-byte value.
- No other headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

4.1.2.3 *Test Condition*

OBEX Server behavior for the Connect PDU is **REQUIRED**.

4.1.2.4 *Expected Outcome*

4.1.2.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The entire Connect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set
- The Connect Packet Length field indicates the correct size.
- The Protocol field indicates OBEX version 1.0.
- The Flags field has no bits set.
- The Maximum OBEX Packet Length field specifies a size of 255 or above.
- The Connect PDU contains properly formed **Who** and **ConnId** headers describing the OBEX Service being connected to.
- The Success response code is returned.

The Connect Operation is successful.

4.1.2.4.2 Fail Verdict

The OBEX Server does not transmit a Connect Response.

The Connect Response does not contain acceptable values.

The Connect Operation is not successful.

4.1.2.5 *Notes*

Various other headers may be transmitted with the Connect Response; these do not affect the result of the test.

4.1.3 Test S-C-3: Invalid Directed Connection

Demonstrates the transmission of an unsuccessful Directed OBEX Connect operation. This directed connection is targeted at an invalid OBEX service.

4.1.3.1 *Test Status*

Accepted

4.1.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is 22 bytes (0x0016).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Request packet contains a properly formed **Target** (0x46) header. For this test, the invalid OBEX service "OBEX-INVALID", encoded as 12 bytes of ASCII data, should be used. The length of the header is 15 bytes (0x000F)
- No other headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

4.1.3.3 *Test Condition*

OBEX Server behavior for the Connect PDU is **REQUIRED**.

4.1.3.4 *Expected Outcome*

4.1.3.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The entire Connect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set
- The Connect Packet Length field indicates the correct size.
- The Protocol field indicates OBEX version 1.0.
- The Flags field has no bits set.
- The Maximum OBEX Packet Length field specifies a size of 255 or above.
- The Connect PDU does not contain **Who** or **ConnId** headers.
- The Success response code is returned.

The Connect Operation is successful.

4.1.3.4.2 Fail Verdict

The OBEX Server does not transmit a Connect Response.

The Connect Response does not contain acceptable values.

4.1.3.4.3 Inconclusive Verdict

A non-successful response code is returned. The OBEX specification merely states that it **strongly recommends** that an OBEX Server accept a connection to an invalid service.

4.1.3.5 *Notes*

Various other headers may be transmitted with the Connect Response; these do not affect the result of the test.

4.2 *OBEX Disconnect PDU*

4.2.1 Test S-D-1: Simple Disconnect Operation

Demonstrates the transmission of an OBEX Disconnect operation.

4.2.1.1 *Test Status*

Accepted

4.2.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client establishes an OBEX Connection. See **4.1.1.2** for the full description of this process.

The OBEX Client transmits a Disconnect Request. The following must occur:

- A Disconnect Request packet with the Final Bit set is sent (0x81).
- The Disconnect Request packet length is 3 bytes (0x0003).
- No headers are included.

See **A.2.1** for a complete diagram of the event sequence chart.

4.2.1.3 *Test Condition*

OBEX Server behavior for the Disconnect PDU is **REQUIRED**.

4.2.1.4 *Expected Outcome*

4.2.1.4.1 Pass Verdict

The OBEX Connection is established successfully.

The OBEX Server receives a Disconnect Request.

The OBEX Server transmits a Disconnect Response. The following are true:

- The entire Disconnect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Success response code is returned.

The Disconnect Operation is successful.

4.2.1.4.2 Fail Verdict

The OBEX Connection is unsuccessful.

The OBEX Server does not transmit a Disconnect Response.

The Disconnect Response does not contain acceptable values.

The Disconnect Operation is not successful.

4.2.1.5 *Notes*

Various headers may be transmitted with the Disconnect Response; these do not affect the result of the test.

4.2.2 Test S-D-2: Simple Directed Disconnect Operation

Demonstrates the transmission of a directed OBEX Disconnect operation.

4.2.2.1 *Test Status*

Accepted

4.2.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client establishes a directed OBEX Connection. See **4.1.2.2** for the full description of this process.

The OBEX Client transmits a Disconnect Request. The following must occur:

- A Disconnect Request packet with the Final Bit set is sent (0x81).
- The Disconnect Request packet length is 8 bytes (0x0008).
- The Disconnect Request packet contains a properly formed **Connection ID** (0xCB) header. The length of this header is 5 bytes and will contain the 4-byte value that was returned by the OBEX Server in the OBEX Connect Response packet.
- No other headers are included.

See **A.2.1** for a complete diagram of the event sequence chart.

4.2.2.3 *Test Condition*

OBEX Server behavior for the Disconnect PDU is **REQUIRED**.

4.2.2.4 *Expected Outcome*

4.2.2.4.1 Pass Verdict

The directed OBEX Connection is established successfully.

The OBEX Server receives a Disconnect Request.

The OBEX Server transmits a Disconnect Response. The following are true:

- The entire Disconnect PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Success response code is returned.

The Disconnect Operation is successful.

4.2.2.4.2 Fail Verdict

The directed OBEX Connection is unsuccessful.

The OBEX Server does not transmit a Disconnect Response.

The Disconnect Response does not contain acceptable values.

The Disconnect Operation is not successful.

4.2.2.5 *Notes*

Various headers may be transmitted with the Disconnect Response; these do not affect the result of the test.

4.3 *OBEX SetPath PDU*

4.3.1 Test S-SP-1: Simple SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation for a downward path change.

4.3.1.1 *Test Status*

Accepted

4.3.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a SetPath Request. The following must occur:

- A SetPath Request packet with the Final Bit set is sent (0x85).
- The SetPath Request packet length is 24 bytes (0x0018).
- The Flags field has no bits set (0x00).
- The Constants field has no bits set (0x00).
- The SetPath Request contains a properly formed **Name** (0x01) header. This Name header must indicate the "Testing" downward (sub-directory) path change for the OBEX Server device, encoded as 16 bytes of null-terminated Unicode data. The length of this header is 19 bytes (0x0013).
- No other headers are included.

See **A.3.1** for a complete diagram of the event sequence chart.

4.3.1.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

4.3.1.4 *Expected Outcome*

4.3.1.4.1 Pass Verdict

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Success response code is returned.

The SetPath Operation is successful.

4.3.1.4.2 Fail Verdict

The OBEX Server does not transmit a SetPath Response.

The SetPath Response does not contain acceptable values.

The SetPath Operation is not successful.

4.3.1.5 *Notes*

Various other headers may be transmitted with the SetPath Response; these do not affect the result of the test.

4.3.2 Test S-SP-2: Backup SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation for an upward path change.

4.3.2.1 *Test Status*

Accepted

4.3.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a SetPath Request. The following must occur:

- A SetPath Request packet with the Final Bit set is sent.
- A SetPath Request packet with the Final Bit set is sent (0x85).
- The SetPath Request packet length is 5 bytes (0x0005).

- The Flags field has the **Backup** bit set. (0x01).
- The Constants field has no bits set (0x00).

See **A.3.1** for a complete diagram of the event sequence chart.

4.3.2.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

4.3.2.4 *Expected Outcome*

4.3.2.4.1 Pass Verdict

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Success response code is returned.

The SetPath Operation is successful.

4.3.2.4.2 Fail Verdict

The OBEX Server does not transmit a SetPath Response.

The SetPath Response does not contain acceptable values.

The SetPath Operation is not successful.

4.3.2.5 *Notes*

Various other headers may be transmitted with the SetPath Response; these do not affect the result of the test.

4.3.3 Test S-SP-3: Reset SetPath Operation

Demonstrates the transmission of an OBEX SetPath operation to reset to the default path.

4.3.3.1 *Test Status*

Accepted

4.3.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a SetPath Request. The following must occur:

- A SetPath Request packet with the Final Bit set is sent (0x85).
- The SetPath Request packet length is 8 bytes (0x0008).
- The Flags field has no bits set (0x00).
- The Constants field has no bits set (0x00).
- The SetPath Request contains a properly formed empty **Name** (0x01) header. This Name header indicates a path reset request. The length of the name header is 3 bytes (0x0003).

See **A.3.1** for a complete diagram of the event sequence chart.

4.3.3.3 *Test Condition*

If support for the SetPath PDU is not present, this test may be **SKIPPED**.

4.3.3.4 *Expected Outcome*

4.3.3.4.1 Pass Verdict

The OBEX Server receives a SetPath Request.

The OBEX Server transmits a SetPath Response. The following are true:

- The entire SetPath PDU does not exceed the default OBEX packet size of 255 bytes.
- The Final bit is set.
- The Success response code is returned.

The SetPath Operation is successful.

4.3.3.4.2 Fail Verdict

The OBEX Server does not transmit a SetPath Response.

The SetPath Response does not contain acceptable values.

The SetPath Operation is not successful.

4.3.3.5 *Notes*

Various other headers may be transmitted with the SetPath Response; these do not affect the result of the test.

4.4 *OBEX Get PDU*

4.4.1 Test S-G-1: Normal Get Operation

Demonstrates the transmission of a small 25-byte Get operation.

4.4.1.1 *Test Status*

Accepted

4.4.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.4.1.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.4.1.4 *Expected Outcome*

4.4.1.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- The Get Packet Length indicates the correct size.
- A Body and/or End Of Body header is sent with at least a 25-byte object.

- The Final Bit is set.
- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.

The Get Operation is successful.

4.4.1.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Responses do not contain at least 25 bytes of Body and/or End Of Body object data.

The Get Operation is not successful.

4.4.1.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.4.2 Test S-G-2: Maximum Get Operation

Demonstrate a successful Get of a maximum sized object supported by the device. If the maximum size object is arbitrarily large, an upper bound of 10 kilobytes may be used.

4.4.2.1 *Test Status*

Accepted

4.4.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.4.2.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.4.2.4 *Expected Outcome*

4.4.2.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- The Get Packet Length indicates the correct size.
- A Body and/or End Of Body header is sent with at least a 10 Kbyte object.
- The Final Bit is set.

- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.

The Get Operation is successful.

4.4.2.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Responses do not contain at least 10 Kbytes of Body and/or End Of Body object data.

The Get Operation is not successful.

4.4.2.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.4.3 Test S-G-3: Zero Byte Get Operation

Demonstrate a successful **GET** of a 0-byte object.

4.4.3.1 *Test Status*

Accepted

4.4.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.4.3.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.4.3.4 *Expected Outcome*

4.4.3.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- The Get Packet Length indicates the correct size.
- A Body and/or End Of Body header is sent with a 0-byte object.
- The Final Bit is set.
- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.

The Get Operation is successful.

4.4.3.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Responses do not contain exactly 0-bytes of Body and/or End Of Body object data.

The Get Operation is not successful.

4.4.3.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.4.4 Test S-G-4: Default Object Get Operation

Demonstrates the transmission of a Get operation for the default object. This test requires the retrieval of a 25-byte default business card.

4.4.4.1 *Test Status*

Accepted

4.4.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 22 bytes (0x0016) with the addition of a Name and Type headers.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must be an empty header indicating the retrieval of a default object. The length of this header is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Type** (0x42) header indicating the type of default object to be retrieved. In this case the value should be “text/x-vCard”, encoded as a 13-byte null-terminated ASCII string. The length of this header is 16 bytes (0x0010).
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.4.4.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.4.4.4 *Expected Outcome*

4.4.4.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the Server. The following are true:

- The Get Packet Length indicates the correct size.
- A Body and/or End Of Body header is sent with at least a 25-byte default business card object.
- The Final Bit is set.
- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.

The Get Operation is successful.

4.4.4.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Responses do not contain at least 25-bytes of Body and/or End Of Body default object data.

The Get Operation is not successful.

4.4.4.4.3 Inconclusive Verdict

No default business card is present, as indicated by an unsuccessful response code (e.g. Not Found) from the OBEX Server device.

4.4.4.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.4.5 Test S-G-5: Get Using Advertised Packet Size

Demonstrate that the test GS2 when performed after a successful OBEX **CONNECT** operation utilizes the larger OBEX packet size advertised in the **CONNECT** request.

4.4.5.1 *Test Status*

Accepted

4.4.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is seven bytes (0x0007).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5K bytes (0x1400).
- No headers are included.

The OBEX Client receives a Connect Response.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.3** for a complete diagram of the event sequence chart.

4.4.5.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.4.5.4 *Expected Outcome*

4.4.5.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response.

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- The Get Packet Length indicates the correct size. For the Get Responses including object data, the larger OBEX packet size (5K) supported by the OBEX Client should be utilized. If packet sizes greater than 1K are used, this is sufficient.
- A Body and/or End Of Body header is sent with at least a 10 Kbyte object.
- The Final Bit is set.
- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.

The Get Operation is successful.

4.4.5.4.2 Fail Verdict

The OBEX Server does not transmit a Connect Response.

The OBEX Server does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Responses do not contain at least 10 Kbytes of Body and/or End Of Body object data.

The Get Operation is not successful.

4.4.5.5 *Notes*

Various other headers may be transmitted with the Connect Response or the Get Responses; these do not affect the result of the test.

4.5 *OBEX Put PDU*

4.5.1 Test S-P-1: Small Put Operation

Demonstrates the transmission of a small 25-byte Put operation.

4.5.1.1 *Test Status*

Accepted

4.5.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The Put Request contains a properly formed **Length** (0xC3) header. The Length header is 5 bytes and will contain the length of the object to be transferred (25 bytes or 0x00000019).

- The Put Request contains a properly formed **Body** (0x48) header. This Body header will contain 12-bytes of random data forming the object being sent. The length of this header is 15 bytes (0x000F).
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 13-bytes of random data forming the rest of the object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 16 bytes (0x0010).
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

4.5.1.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

4.5.1.4 *Expected Outcome*

4.5.1.4.1 Pass Verdict

The OBEX Server receives a Put Request. The following are true:

- A 25-byte object should have been received and stored according to the name provided by the OBEX Client.

The OBEX Server transmits a Put Response. The following are true:

- The Put Packet Length indicates the correct size.
- The Final Bit is set.
- The Success response code is sent.

The Put Operation is successful.

4.5.1.4.2 Fail Verdict

The OBEX Server does not transmit a Put Response.

The Put Response does not contain acceptable values.

The Put Operation is not successful.

4.5.1.5 *Notes*

Various other headers may be transmitted with the Put Response; these do not affect the result of the test.

4.5.2 Test S-P-2: Maximum Put Operation

Demonstrates the transmission of a 10 Kbyte Put operation.

4.5.2.1 *Test Status*

Accepted

4.5.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Requests are transmitted by the OBEX Client. The following must occur:

- All Put packets must fit within the 255 minimum OBEX packet size. This ensures that all devices can interoperate.
- **51** total Put Request packets will be sent.
- **10240 bytes** of object data will be sent. The first packet contains 40 bytes of object data, while the next 50 packets contain 204 bytes each.
- The first 50 Put Request packets are sent as (0x02).
- The last Put Request packet has the Final Bit set (0x82).

- The first Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The first Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The first Put Request contains a properly formed **Length** (0xC3) header. The Length header is 5 bytes and will contain the length of the object to be transferred (10240 bytes or 0x00002800).
- The first Put Request contains a properly formed **Body** (0x48) header. This Body header will contain the first 40-bytes of random data forming the object being sent. The length of this header is 43 bytes (0x002B).
- The remaining Put Request packets will have a length of 210 bytes (0x00D2) with the addition of the Body/End-Of-Body header.
- The remaining Put Requests contain a properly formed **Body** (0x48) header with the exception of the last Put packet that contains an **End Of Body** (0x49) header instead. These Body headers will contain 204 bytes of random data forming the object being sent. The length of this header is 207 bytes (0x00CF).
- No other headers are included.

Put Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Put Request packet for every Put Response packet received until the Put operation is completed.

See **A.5.1** for a complete diagram of the event sequence chart.

4.5.2.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

4.5.2.4 *Expected Outcome*

4.5.2.4.1 Pass Verdict

Put Requests are received by the OBEX Server. The following are true:

- A 10 Kbyte object should have been received and stored according to the name provided by the OBEX Client.

Put Responses are transmitted by the OBEX Server. The following are true:

- The Put Packet Length indicates the correct size.
- The Final Bit is set.
- The Success response code is sent.

The Put Operation is successful.

4.5.2.4.2 Fail Verdict

The OBEX Server does not transmit the Put Responses.

The Put Responses do not contain acceptable values.

The Put Operation is not successful.

4.5.2.5 *Notes*

Various other headers may be transmitted with the Put Responses; these do not affect the result of the test.

4.5.3 Test S-P-3: Zero Byte Put Operation

Demonstrates the transmission of a 0-byte Put operation.

4.5.3.1 *Test Status*

Accepted

4.5.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The Put Request contains a properly formed **Length** (0xC3) header. The Length header is 5 bytes and will contain the length of the object to be transferred (0 bytes or 0x00000000).
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 0-bytes of object data. This is the last header in the packet and signifies the end of the object. The length of this header is 3 bytes (0x0003).
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

4.5.3.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

4.5.3.4 *Expected Outcome*

4.5.3.4.1 Pass Verdict

The OBEX Server receives a Put Request. The following are true:

- A 0-byte object should have been received and stored according to the name provided by the OBEX Client.

The OBEX Server transmits a Put Response. The following are true:

- The Put Packet Length indicates the correct size.
- The Final Bit is set.
- The Success response code is sent.

The Put Operation is successful.

4.5.3.4.2 Fail Verdict

The OBEX Server does not transmit a Put Response.

The Put Response does not contain acceptable values.

The Put Operation is not successful.

4.5.3.5 *Notes*

Various other headers may be transmitted with the Put Response; these do not affect the result of the test.

4.5.4 Test S-P-4: Put Delete Operation

Demonstrates the transmission of a Put operation to delete an object.

4.5.4.1 *Test Status*

Accepted

4.5.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. The Name header will indicate the name of a valid object on the OBEX Server. This object will be deleted from the OBEX Server during this operation. The name is encoded as null-terminated Unicode data and will included three byes of header description.
- No Body or End Of Body headers are sent.
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

4.5.4.3 *Test Condition*

If Put Delete operations are not supported, this test may be **SKIPPED**.

4.5.4.4 *Expected Outcome*

4.5.4.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following are true:

- The Put Packet Length indicates the correct size.
- The Final Bit is set.
- The Success response code is sent.

The Put Delete Operation is successful.

The OBEX Server deleted the requested file.

4.5.4.4.2 Fail Verdict

The OBEX Server does not transmit the Put Response.

The Put Response does not contain acceptable values.

The Put Delete Operation is not successful.

The OBEX Server did not delete the requested file.

4.5.4.5 *Notes*

Various other headers may be transmitted with the Put Response; these do not affect the result of the test.

4.6 *OBEX Abort PDU*

4.6.1 Test S-A-1: Simple Put Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Put operation.

4.6.1.1 *Test Status*

Accepted

4.6.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet without the Final Bit set is sent (0x02).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- No other headers are included.

The OBEX Client receives a Put Response.

The OBEX Client transmits an Abort Request. The following must occur:

- An Abort Request packet with the Final Bit set is sent (0xFF)
- The Abort Request packet length is 3 bytes (0x0003).
- No headers are included.

See **A.6.1** for a complete diagram of the event sequence chart.

4.6.1.3 *Test Condition*

If support for the Abort PDU is not present, this test may be **SKIPPED**.

4.6.1.4 *Expected Outcome*

4.6.1.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response.

The OBEX Server receives an Abort Request.

The OBEX Server transmits an Abort Response. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set
- The Success Response code is sent.

The Put operation is successfully canceled.

4.6.1.4.2 Fail Verdict

The OBEX Server does not transmit a Put Response.

The OBEX Server does not receive an Abort Request.

The OBEX Server does not transmit an Abort Response.

The Abort Response does not contain acceptable values.

The Put operation is not successfully canceled.

4.6.1.5 *Notes*

Various headers may be transmitted with the Put Response and Abort Response; these do not affect the result of the test.

4.6.2 Test S-A-2: Immediate Put Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Put operation. The Abort is issued immediately following a Put Request, without waiting for a Put Response packet.

4.6.2.1 *Test Status*

Accepted

4.6.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet without the Final Bit set is sent (0x02).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- No other headers are included.

The OBEX Client immediately transmits an Abort Request before the OBEX Server can respond to the previous Put Request. The following must occur:

- An Abort Request packet with the Final Bit set is sent (0xFF)
- The Abort Request packet length is 3 bytes (0x0003).
- No headers are included.

See **A.6.1** for a complete diagram of the event sequence chart.

4.6.2.3 *Test Condition*

If support for the Abort PDU is not present, this test may be **SKIPPED**.

4.6.2.4 *Expected Outcome*

4.6.2.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server receives an Abort Request.

The OBEX Server transmits a Put Response.

The OBEX Server transmits an Abort Response. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set
- The Success Response code is sent.

The Put operation is successfully canceled.

4.6.2.4.2 Fail Verdict

The OBEX Server does not receive an Abort Request.

The OBEX Server does not transmit a Put Response.

The OBEX Server does not transmit an Abort Response.

The Abort Response does not contain acceptable values.

The Put operation is not successfully canceled.

4.6.2.5 *Notes*

Various headers may be transmitted with the Put Response and Abort Response; these do not affect the result of the test.

4.6.3 Test S-A-3: Simple Get Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Get operation. At least a 256-byte object should be requested, in order to guarantee that the Get operation takes at least two packets to complete.

4.6.3.1 *Test Status*

Accepted

4.6.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Get Request. The following must occur:

- A Get Request packet without the Final Bit set is sent (0x03).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including three bytes of header description.
- No other headers are included.

The OBEX Client receives a Get Response.

The OBEX Client transmits an Abort Request. The following must occur:

- An Abort Request packet with the Final Bit set is sent (0xFF)
- The Abort Request packet length is 3 bytes (0x0003).
- No headers are included.

See **A.6.2** for a complete diagram of the event sequence chart.

4.6.3.3 *Test Condition*

If support for the Get or Abort PDU is not present, this test may be **SKIPPED**.

4.6.3.4 *Expected Outcome*

4.6.3.4.1 Pass Verdict

The OBEX Server receives a Get Request.

The OBEX Server transmits a Get Response.

The OBEX Server receives an Abort Request.

The OBEX Server transmits an Abort Response. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set
- The Success Response code is sent.

The Get operation is successfully canceled.

4.6.3.4.2 Fail Verdict

The OBEX Server does not transmit a Get Response.

The OBEX Server does not receive an Abort Request.

The OBEX Server does not transmit an Abort Response.

The Abort Response does not contain acceptable values.

The Get operation is not successfully canceled.

4.6.3.5 *Notes*

Various headers may be transmitted with the Get Response and Abort Response; these do not affect the result of the test.

4.6.4 Test S-A-4: Immediate Get Abort

Demonstrates the transmission of an OBEX Abort operation to cancel an existing Get operation. The Abort is issued immediately following a Get Request, without waiting for a Get Response packet. At least a 256-byte object should be transferred, in order to guarantee that the Get operation takes at least two packets to complete.

4.6.4.1 *Test Status*

Accepted

4.6.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Get Request. The following must occur:

- A Get Request packet without the Final Bit set is sent (0x03).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including the 3 bytes of header description.
- No other headers are included.

The OBEX Client immediately transmits an Abort Request before the OBEX Server can respond to the previous Get Request. The following must occur:

- An Abort Request packet with the Final Bit set is sent (0xFF)
- The Abort Request packet length is 3 bytes (0x0003).
- No headers are included.

See **A.6.2** for a complete diagram of the event sequence chart.

4.6.4.3 *Test Condition*

If support for the Get or Abort PDU is not present, this test may be **SKIPPED**.

4.6.4.4 *Expected Outcome*

4.6.4.4.1 Pass Verdict

The OBEX Server receives a Get Request.

The OBEX Server receives an Abort Request.

The OBEX Server transmits a Get Response.

The OBEX Server transmits an Abort Response. The following is true:

- The Abort Packet Length indicates the correct size.
- The Final Bit is set
- The Success Response code is sent.

The Get operation is successfully canceled.

4.6.4.4.2 Fail Verdict

The OBEX Server does not receive an Abort Request.

The OBEX Server does not transmit a Get Response.

The OBEX Server does not transmit an Abort Response.

The Abort Response does not contain acceptable values.

The Get operation is not successfully canceled.

4.6.4.5 *Notes*

Various headers may be transmitted with the Get Response and Abort Response; these do not affect the result of the test.

4.7 *OBEX Session PDU*

4.7.1 Test S-S-1: Create Session

Demonstrates the transmission of an OBEX Create Session operation.

4.7.1.1 *Test Status*

Accepted

4.7.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Create Session Request. The following must occur:

- The Create Session Request with the Final Bit set is sent (0x87).
- The Create Session Request packet length will be 33 bytes (0x0021).
- The Create Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 30 bytes (0x001E).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, and Nonce* (0x01) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Create Session (0x00) opcode as its value. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value").
- No other headers are sent.

See **A.7.1** for a complete diagram of the event sequence chart.

4.7.1.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.1.4 *Expected Outcome*

4.7.1.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

- The Session PDU contains a properly formed **Session Parameters** header with the *Session Opcode, Device Address, and Nonce* tag/length/value triplets.

The OBEX Server transmits a Create Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain the *Device Address, Nonce, and Session ID* tag/length/value triplets.

The *Session ID* transmitted in the Create Session Response should be verified against a manually formed *Session ID* from the values provided in the Create Session Request and Create

Session Response packets. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").

The Create Session Operation is successful.

4.7.1.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Response does not contain acceptable values.

The Create Session Operation is not successful.

4.7.1.5 *Notes*

Various other headers may be transmitted with the Create Session Response; these do not affect the result of the test.

4.7.2 Test S-S-2: Close Active Session

Demonstrates the transmission of an OBEX Close Session operation to close an active reliable session.

4.7.2.1 *Test Status*

Accepted

4.7.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Close Session Request.

The OBEX Client transmits a Close Session Request. The following must occur:

- The Close Session Request with the Final Bit set is sent (0x87).
- The Close Session Request packet length will be 27 bytes (0x001B).
- The Close Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 24 bytes (0x0018).
- The Session Parameters header must contain the *Session Opcode* (0x05) *and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Close Session (0x01) opcode as its value. The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being closed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

See **A.7.2** for a complete diagram of the event sequence chart.

4.7.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.2.4 *Expected Outcome*

4.7.2.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU does not contain a **Session Parameters** header.

The Close Session Operation is successful.

4.7.2.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Close Session Response.

The Close Session Response does not contain acceptable values.

The Close Session Operation is not successful.

4.7.2.5 *Notes*

Various other headers may be transmitted with the Create Session Response or Close Session Response; these do not affect the result of the test.

4.7.3 Test S-S-3: Close Suspended Session

Demonstrates the transmission of an OBEX Close Session operation to close a suspended reliable session.

4.7.3.1 *Test Status*

Accepted

4.7.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Close Session Request.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

The OBEX Client transmits a Close Session Request. The following must occur:

- The Close Session Request with the Final Bit set is sent (0x87).
- The Close Session Request packet length will be 27 bytes (0x001B).
- The Close Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 24 bytes (0x0018).
- The Session Parameters header must contain the *Session Opcode* (0x05) *and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Close Session (0x01) opcode as its value. The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being closed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

See **A.7.3** for a complete diagram of the event sequence chart.

4.7.3.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.3.4 *Expected Outcome*

4.7.3.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU does not contain a **Session Parameters** header.

The Close Session Operation is successful.

4.7.3.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Close Session Response.

The Close Session Response does not contain acceptable values.

The Close Session Operation is not successful.

4.7.3.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Suspend Session Response, or Close Session Response; these do not affect the result of the test.

4.7.4 Test S-S-4: Suspend Session

Demonstrates the transmission of an OBEX Suspend Session operation.

4.7.4.1 *Test Status*

Accepted

4.7.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

See **A.7.4** for a complete diagram of the event sequence chart.

4.7.4.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.4.4 *Expected Outcome*

4.7.4.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU may contain a properly formed **Session Parameters** header as the first header in the packet. (**OPTIONAL**)
- If the Session Parameters header exists, it must contain the *Timeout* tag/length/value triplet.

The Suspend Session Operation is successful.

4.7.4.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Response does not contain acceptable values.

The Suspend Session Operation is not successful.

4.7.4.5 *Notes*

Various other headers may be transmitted with the Create Session Response or Suspend Session Response; these do not affect the result of the test.

4.7.5 Test S-S-5: Resume Session

Demonstrates the transmission of an OBEX Resume Session operation.

4.7.5.1 *Test Status*

Accepted

4.7.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Request.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

The OBEX Client transmits a Resume Session Request. The following must occur:

- The Resume Session Request with the Final Bit set is sent (0x87).
- The Resume Session Request packet length will be 51 bytes (0x33).
- The Resume Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x30).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, Nonce* (0x01)*, and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Resume Session (0x03) opcode as its value. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being resumed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

See **A.7.5** for a complete diagram of the event sequence chart.

4.7.5.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.5.4 *Expected Outcome*

4.7.5.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain at least the *Device Address, Nonce, Session ID and Next Sequence Number* tag/length/value triplets.

The *Session ID* transmitted in the Resume Session Response should be verified against the *Session ID* provided in the Resume Session Request packet.

The Resume Session Operation is successful.

4.7.5.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation is not successful.

4.7.5.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Suspend Session Response, or Resume Session Response; these do not affect the result of the test.

4.7.6 Test S-S-6: Set Session Timeout

Demonstrates the transmission of an OBEX Set Session Timeout operation.

4.7.6.1 *Test Status*

Accepted

4.7.6.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

The OBEX Client transmits a Set Session Timeout Request. The following must occur:

- The Set Session Timeout Request with the Final Bit set is sent (0x87).
- The Set Session Timeout Request packet length will be 9 bytes (0x0009).

- The Set Session Timeout Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Set Timeout (0x04) opcode as its value.
- No other headers are sent.

See **A.7.7** for a complete diagram of the event sequence chart.

4.7.6.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.6.4 *Expected Outcome*

4.7.6.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Set Session Timeout Request.

The OBEX Server transmits a Set Session Timeout Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU may contain a properly formed **Session Parameters** header as the first header in the packet. (**OPTIONAL**)
- If the Session Parameters header exists, it must contain the *Timeout* tag/length/value triplet.

The Set Session Timeout Operation is successful.

4.7.6.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Set Session Timeout Response.

The Set Session Timeout Response does not contain acceptable values.

The Set Session Timeout Operation is not successful.

4.7.6.5 *Notes*

Various other headers may be transmitted with the Create Session Response or Set Session Timeout Response; these do not affect the result of the test.

4.7.7 Test S-S-7: Set Session Timeout (Infinite Timeout)

Demonstrates the transmission of an OBEX Set Session Timeout operation specifying an infinite timeout value.

4.7.7.1 *Test Status*

Accepted

4.7.7.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Request.

The OBEX Client transmits a Set Session Timeout Request. The following must occur:

- The Set Session Timeout Request with the Final Bit set is sent (0x87).
- The Set Session Timeout Request packet length will be 9 bytes (0x0009).
- The Set Session Timeout Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Set Timeout (0x04) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Set Session Timeout Response.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

Wait 5 seconds assuming the infinite Suspend Session Timer will not expire.

The OBEX Client will resume the reliable OBEX session. See **4.7.5.2** for the full description of this process.

See **A.7.8** for a complete diagram of the event sequence chart.

4.7.7.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.7.4 *Expected Outcome*

4.7.7.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Set Session Timeout Request.

The OBEX Server transmits a Set Session Timeout Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU may contain a properly formed **Session Parameters** header as the first header in the packet. (**OPTIONAL**)
- If the Session Parameters header exists, it must contain the *Timeout* tag/length/value triplet with a value of 0xFFFFFFFF.

The Set Session Timeout Operation is successful.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Success** (0xA0) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Resume Session Operation is successful.

4.7.7.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Set Session Timeout Response.

The Set Session Timeout Response does not contain acceptable values.

The Set Session Timeout Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation is not successful.

4.7.7.4.3 Inconclusive Verdict

The OBEX Server transmits a Set Session Timeout Response specifying a non-infinite timeout value.

4.7.7.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Set Session Timeout Response, Suspend Session Response, or Resume Session Response; these do not affect the result of the test.

4.7.8 Test S-S-8: Set Session Timeout (1 second timeout)

Demonstrates the transmission of an OBEX Set Session Timeout operation and the expiration of the suspended reliable session when the suspend session timeout expires.

4.7.8.1 *Test Status*

Accepted

4.7.8.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Request.

The OBEX Client transmits a Set Session Timeout Request. The following must occur:

- The Set Session Timeout Request with the Final Bit set is sent (0x87).

- The Set Session Timeout Request packet length will be 15 bytes (0x000E).
- The Set Session Timeout Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 12 bytes (0x000C).
- The Session Parameters header must contain the *Session Opcode* (0x05) and *Timeout* (0x04) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Set Timeout (0x04) opcode as its value. The *Timeout* triplet must have a four-byte length (0x04) and 1 second as the timeout value (0x00000001).
- No other headers are sent.

The OBEX Client receives a Set Session Timeout Response.

- The Set Session Timeout Response might contain a **Session Parameters** (0x52) header. If it does, this header will contain the *Timeout* (0x04) tag/length/value triplet. The timeout value presented in this header might be larger than the 1 second timeout we requested in our request. However, since the smallest timeout value is negotiated, our test can always be guaranteed that the suspend timer should expire.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

Wait 1.5 seconds for the Suspend Session Timer to expire.

The OBEX Client will resume the reliable OBEX session. See **4.7.5.2** for the full description of this process.

See **A.7.9** for a complete diagram of the event sequence chart.

4.7.8.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.8.4 *Expected Outcome*

4.7.8.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Set Session Timeout Request.

The OBEX Server transmits a Set Session Timeout Response.

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU may contain a properly formed **Session Parameters** header as the first header in the packet. (**OPTIONAL**)
- If the Session Parameters header exists, it must contain the *Timeout* tag/length/value triplet with a value no larger than 0x00000001.

The Set Session Timeout Operation is successful.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Service Unavailable** (0xD3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Resume Session Operation is unsuccessful.

4.7.8.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Set Session Timeout Response.

The Set Session Timeout Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation is successful.

4.7.8.4.3 Inconclusive Verdict

The OBEX Server transmits a Set Session Timeout Response specifying a timeout value larger than one second.

4.7.8.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Set Session Timeout Response, Suspend Session Response, or Resume Session Response; these do not affect the result of the test.

4.7.9 Test S-S-9: Create Session (1 second timeout)

Demonstrates the transmission of an OBEX Create Session operation and the expiration of the suspended reliable session when the suspend session timeout expires.

4.7.9.1 *Test Status*

Accepted

4.7.9.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Create Session Request. The following must occur:

- The Create Session Request with the Final Bit set is sent (0x87).
- The Create Session Request packet length will be 39 bytes (0x0027).
- The Create Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 36 bytes (0x0024).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, Nonce* (0x01), and *Timeout* (0x04) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Create Session (0x00) opcode as its value. The *Device Address*

will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Timeout* triplet must have a four-byte length (0x04) and 1 second as the timeout value (0x00000001).

- No other headers are sent.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Request. This header might also contain the *Timeout* (0x04) tag/length/value triplet. The timeout value presented in this header might be larger than the 1 second timeout we requested in our request. However, since the smallest timeout value is negotiated, our test can always be guaranteed that the suspend timer should expire.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

Wait 1.5 seconds for the Suspend Session Timer to expire.

The OBEX Client will resume the reliable OBEX session. See **4.7.5.2** for the full description of this process.

See **A.7.10** for a complete diagram of the event sequence chart.

4.7.9.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.9.4 *Expected Outcome*

4.7.9.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The Success Response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.
- The Session PDU contains a properly formed **Session Parameters** header as the first header in the packet.
- The Session Parameters header must contain the *Device Address, Nonce, and Session ID* tag/length/value triplets.

The *Session ID* transmitted in the Create Session Response should be verified against a manually formed *Session ID* from the values provided in the Create Session Request and Create Session Response packets. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Service Unavailable** (0xD3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Resume Session Operation is unsuccessful.

4.7.9.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation is successful.

4.7.9.4.3 Inconclusive Verdict

The OBEX Server transmits a Create Session Response specifying a timeout value larger than one second.

4.7.9.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Suspend Session Response, or Resume Session Response; these do not affect the result of the test.

4.7.10 Test S-S-10: Multiple Sessions

Demonstrates the creation and closure of two OBEX reliable sessions.

4.7.10.1 *Test Status*

Accepted

4.7.10.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the first Close Session Request.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header

and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.

- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

The OBEX Client will create a reliable OBEX session. See **4.7.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the second Close Session Request.

The OBEX Client transmits a Close Session Request. The following must occur:

- This request will close the suspended session, which is also the first session that was created.
- The Close Session Request with the Final Bit set is sent (0x87).
- The Close Session Request packet length will be 27 bytes (0x001B).
- The Close Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 24 bytes (0x0018).
- The Session Parameters header must contain the *Session Opcode* (0x05) *and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Close Session (0x01) opcode as its value. The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the session being closed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

The OBEX Client receives a Close Session Response.

The OBEX Client transmits a Close Session Request to close the active reliable session. See the previous Close Session Request for details on the format of this request.

See **A.7.11** for a complete diagram of the event sequence chart.

4.7.10.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.7.10.4 *Expected Outcome*

4.7.10.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response.

The Close Session Operation is successful.

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response.

The Close Session Operation is successful.

4.7.10.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Operation is not successful.

The OBEX Server does not transmit a Create Session Response.

The Create Session Operation is not successful.

The OBEX Server does not transmit two Close Session Responses.

The Close Session Operations are not successful.

4.7.10.4.3 Inconclusive Verdict

The OBEX Server does not support one active and one suspended OBEX reliable session.

4.7.10.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Suspend Session Response, or Close Session Response; these do not affect the result of the test.

4.8 *Server Rejection Responses*

4.8.1 Test S-SR-1: Server Put Rejection

Demonstrates the transmission of a 10K Put operation rejected by the OBEX Server through an unsuccessful response code.

4.8.1.1 *Test Status*

Accepted

4.8.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Put Request(s) is/are transmitted by the OBEX Client. The following must occur:

- All Put packets must fit within the 255 minimum OBEX packet size. This ensures that all devices can interoperate.
- A maximum of **51** total Put Request packets will be sent, however, the Put Request packets will cease as soon as the Server device rejects the Put operation.
- **10240 bytes** of object data will be sent. The first packet contains 40 bytes of object data, while the next 50 packets contain 204 bytes each.
- Each Put Request will wait **500ms** before sending the next request in order to give the OBEX Server adequate time to abort the operation.
- The first 50 Put Request packets are sent as (0x02).
- The last Put Request packet has the Final Bit set (0x82).
- The first Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The first Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The Put Request contains a properly formed **Length** (0xC3) header. The Length header is 5 bytes and will contain the length of the object to be transferred (10240 bytes or 0x00002800).

- The first Put Request contains a properly formed **Body** (0x48) header. This Body header will contain the first 40-bytes of random data forming the object being sent. The length of this header is 43 bytes (0x002B).
- The remaining Put Request packets will have a length of 210 bytes (0x00D2) with the addition of the Body/End-Of-Body header.
- The remaining Put Requests contain a properly formed **Body** (0x48) header with the exception of the last Put packet that contains an **End Of Body** (0x49) header instead. These Body headers will contain 204 bytes of random data forming the object being sent. The length of this header is 207 bytes (0x00CF).
- No other headers are included.

See **A.9.1** for a complete diagram of the event sequence chart.

4.8.1.3 *Test Condition*

The Put PDU is **REQUIRED** by the OBEX protocol.

4.8.1.4 *Expected Outcome*

4.8.1.4.1 Pass Verdict

Put Request(s) is/are received by the OBEX Server.

Put Response(s) is/are transmitted by the OBEX Server.

The following are true for the last Put Response sent:

- The Put Packet Length indicates the correct size.
- The Final Bit is set.
- An unsuccessful response code is sent; for example Forbidden (0xC3).

The Put Operation is successfully aborted.

4.8.1.4.2 Fail Verdict

The OBEX Server does not transmit the Put Responses.

The Final Put Response does not contain acceptable values.

The Put Operation is not successfully aborted.

4.8.1.4.3 Inconclusive Verdict

The Put operation completes without the OBEX Server device sending an unsuccessful response code.

4.8.1.5 *Notes*

Various other headers may be transmitted with the Put Responses; these do not affect the result of the test.

4.8.2 Test S-SR-2: Server Get Rejection

Demonstrates the transmission of a Get operation rejected by the OBEX Server through an unsuccessful response code. At least a 256-byte object should be transferred, in order to guarantee that the Get operation takes at least two packets to complete. However, larger objects will have a better chance of being aborted, since they will give the OBEX Server more time to reject the operation.

4.8.2.1 *Test Status*

Accepted

4.8.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).

- Each Get Request will wait **500ms** before sending the next request in order to give the OBEX Server adequate time to abort the operation. Another Get request will be issued provided the Get operation has not been aborted (unsuccessful response code) or completed (success response code).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

See **A.9.2** for a complete diagram of the event sequence chart.

4.8.2.3 *Test Condition*

If support for the Get PDU is not present, this test may be **SKIPPED**.

4.8.2.4 *Expected Outcome*

4.8.2.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server.

The following are true for the last Get Response sent:

- The Get Packet Length indicates the correct size.
- The Final Bit is set.
- An unsuccessful response code is sent; for example Forbidden (0xC3).

The Get Operation is successfully aborted.

4.8.2.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Final Get Response does not contain acceptable values.

The Get Operation is not successfully aborted.

4.8.2.4.3 Inconclusive Verdict

The Get operation completes without the OBEX Server device sending an unsuccessful response code.

4.8.2.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.9 *OBEX Session PDU Error Checks*

4.9.1 Test S-E-1: Create Session Fails (No Sessions Available)

Demonstrates the transmission of a Session PDU to create a reliable OBEX session, but no OBEX sessions are available. As a result, no reliable OBEX Session should be created.

4.9.1.1 *Test Status*

Accepted

4.9.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Create Session Request. The following must occur:

- The Create Session Request with the Final Bit set is sent (0x87).
- The Create Session Request packet length will be 33 bytes (0x0021).
- The Create Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 30 bytes (0x001E).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, and Nonce* (0x01) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Create Session (0x00) opcode as its value. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value").
- No other headers are sent.

The OBEX Client receives a Create Session Response.

For every successful Create Session Response received, the OBEX Client will transmit a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

The sequence of creating and suspending sessions will be repeated until a create session request fails.

See **A.8.1** for a complete diagram of the event sequence chart.

4.9.1.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.1.4 *Expected Outcome*

4.9.1.4.1 Pass Verdict

Create Session Request(s) is/are received by the OBEX Server.

Create Session Responses(s) is/are transmitted by the OBEX Server

Suspend Session Requests(s) is/are received by the OBEX Server.

Suspend Session Response(s) is/are transmitted by the OBEX Server.

The OBEX Server receives the final Create Session Request.

The OBEX Server transmits the final Create Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Service Unavailable** (0xD3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The final Create Session Operation is unsuccessful.

4.9.1.4.2 Fail Verdict

The OBEX Server does not transmit the final Create Session Response.

The Create Session Response does not contain acceptable values.

The final Create Session Operation does not fail.

4.9.1.5 *Notes*

Various other headers may be transmitted with the Create Session Response; these do not affect the result of the test.

4.9.2 Test S-E-2: Create Session Fails (Session Already Active)

Demonstrates the transmission of a Session PDU to create a reliable OBEX session, but a reliable OBEX sessions already exists. As a result, the OBEX Create Session request is rejected.

4.9.2.1 *Test Status*

Accepted

4.9.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.9.1.2** for the full description of this process.

The OBEX Client transmits an additional Create Session Request. See **4.9.1.2** for the full description of this process.

The OBEX Client receives an additional Create Session Response.

See **A.8.2** for a complete diagram of the event sequence chart.

4.9.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.2.4 *Expected Outcome*

4.9.2.4.1 Pass Verdict

The OBEX Server receives the Create Session Requests.

The OBEX Server transmits the Create Session Responses.

The following are true for the final Create Session Response sent:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Forbidden** (0xC3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The first Create Session Operation is successful, while the second operation fails.

4.9.2.4.2 Fail Verdict

The OBEX Server does not transmit the Create Session Responses.

The final Create Session Response does not contain acceptable values.

The first Create Session Operation is not successful, or the second operation does not fail.

4.9.2.5 *Notes*

Various other headers may be transmitted with the Create Session Responses; these do not affect the result of the test.

4.9.3 Test S-E-3: Resume Session Fails (Bad Session)

Demonstrates the transmission of a Session PDU to resume a reliable OBEX session, but the Session information provided in the resume request is invalid. As a result, the reliable OBEX Session is not resumed.

4.9.3.1 *Test Status*

Accepted

4.9.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Resume Session Request. The following must occur:

- The Resume Session Request with the Final Bit set is sent (0x87).
- The Resume Session Request packet length will be 51 bytes (0x33).
- The Resume Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x30).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, Nonce* (0x01)*, and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Resume Session (0x03) opcode as its value. The *Device Address* will have a four-byte length (0x04) and contain an **invalid** 32-bit IrDA address. The 32-bit value used for this test will be 0x12345678. The *Nonce* will have a sixteen-byte length (0x10) and contain an **invalid** 16-byte value. The 16-byte value used for this test will be all 0xAB values. The *Session ID* triplet will have a 16-byte length (0x10) and contain an **invalid** 16-byte value. The 16-byte value used for this test will be all 0xCD values.
- No other headers are sent.

The OBEX Client receives a Resume Session Response.

See **A.8.3** for a complete diagram of the event sequence chart.

4.9.3.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.3.4 *Expected Outcome*

4.9.3.4.1 Pass Verdict

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Service Unavailable** (0xD3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Resume Session Operation is unsuccessful.

4.9.3.4.2 Fail Verdict

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation does not fail.

4.9.3.5 *Notes*

Various other headers may be transmitted with the Resume Session Response; these do not affect the result of the test.

4.9.4 Test S-E-4: Resume Session Fails (Session Already Active)

Demonstrates the transmission of a Session PDU to resume a reliable OBEX session, but a reliable OBEX sessions already exists. As a result, the OBEX Resume Session request is rejected.

4.9.4.1 *Test Status*

Accepted

4.9.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.9.1.2** for the full description of this process.

The OBEX Client receives a Create Session Response.

- The Create Session Response will contain a **Session Parameters** (0x52) header. This header will contain the *Device Address* (0x00), *Nonce* (0x01), and *Session ID* (0x02) tag/length/value triplets. The device address and nonce fields will be used when creating the Session ID in the Resume Session Request.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

The OBEX Client will create a reliable OBEX session. See **4.9.1.2** for the full description of this process.

The OBEX Client transmits a Resume Session Request. The following must occur:

- The Resume operation is attempting to resume the suspended session.
- The Resume Session Request with the Final Bit set is sent (0x87).
- The Resume Session Request packet length will be 51 bytes (0x33).
- The Resume Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 48 bytes (0x30).
- The Session Parameters header must contain the *Session Opcode* (0x05)*, Device Address* (0x00)*, Nonce* (0x01)*, and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Resume Session (0x03) opcode as its value. The *Device Address* will have a four-byte length (0x04) and contain the 32-bit IrDA address of the local device as its value. The *Nonce* will have a sixteen-byte length (0x10) and contain a sixteen-byte unique value. One possible way to create the *Nonce* is as follows: MD5("Random Number" "Time Value"). The *Session ID* triplet will have a 16-byte length (0x10) and contain a 16-byte value describing the suspended session being resumed. The *Session ID* is created as follows: MD5("Client Device Address" "Client Nonce" "Server Device Address" "Server Nonce").
- No other headers are sent.

The OBEX Client receives a Resume Session Response.

See **A.8.4** for a complete diagram of the event sequence chart.

4.9.4.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.4.4 *Expected Outcome*

4.9.4.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response.

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

The OBEX Server receives a Resume Session Request.

The OBEX Server transmits a Resume Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Forbidden** (0xC3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Resume Session Operation is unsuccessful.

4.9.4.4.2 Fail Verdict

The OBEX Server does not transmit a Resume Session Response.

The Resume Session Response does not contain acceptable values.

The Resume Session Operation does not fail.

4.9.4.5 *Notes*

Various other headers may be transmitted with the Create Session Response, Suspend Session Response, or Resume Session Response; these do not affect the result of the test.

4.9.5 Test S-E-5: Suspend Session Fails (No Reliable Session)

Demonstrates the transmission of a Session PDU to suspend a reliable OBEX session, but no reliable OBEX Session exists at the time the request is issued. As a result, the OBEX Suspend Session request is rejected.

4.9.5.1 *Test Status*

Accepted

4.9.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Suspend Session Request. The following must occur:

- The Suspend Session Request with the Final Bit set is sent (0x87).
- The Suspend Session Request packet length will be 9 bytes (0x0009).
- The Suspend Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Suspend Session (0x02) opcode as its value.

- No other headers are sent.

The OBEX Client receives a Suspend Session Response.

See **A.8.5** for a complete diagram of the event sequence chart.

4.9.5.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.5.4 *Expected Outcome*

4.9.5.4.1 Pass Verdict

The OBEX Server receives a Suspend Session Request.

The OBEX Server transmits a Suspend Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Forbidden** (0xC3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Suspend Session Operation is unsuccessful.

4.9.5.4.2 Fail Verdict

The OBEX Server does not transmit a Suspend Session Response.

The Suspend Session Response does not contain acceptable values.

The Suspend Session Operation does not fail.

4.9.5.5 *Notes*

Various other headers may be transmitted with the Suspend Session Response; these do not affect the result of the test.

4.9.6 Test S-E-6: Close Session Fails (No Reliable Session)

Demonstrates the transmission of a Session PDU to close a reliable OBEX session, but no reliable OBEX Session exists at the time the request is issued. As a result, the OBEX Close Session request is rejected.

4.9.6.1 *Test Status*

Accepted

4.9.6.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Close Session Request. The following must occur:

- The Close Session Request with the Final Bit set is sent (0x87).
- The Close Session Request packet length will be 27 bytes (0x001B).
- The Close Session Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 24 bytes (0x0018).
- The Session Parameters header must contain the *Session Opcode* (0x05) *and Session ID* (0x02) tag/length/value triplets. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Close Session (0x01) opcode as its value. The *Session ID* triplet will have a 16-byte length (0x10) and contain an **invalid** 16-byte value. The 16-byte value used for this test will be all 0xAB values.
- No other headers are sent.

The OBEX Client receives a Close Session Response.

See **A.8.6** for a complete diagram of the event sequence chart.

4.9.6.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.6.4 *Expected Outcome*

4.9.6.4.1 Pass Verdict

The OBEX Server receives a Close Session Request.

The OBEX Server transmits a Close Session Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Forbidden** (0xC3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Close Session Operation is unsuccessful.

4.9.6.4.2 Fail Verdict

The OBEX Server does not transmit a Close Session Response.

The Close Session Response does not contain acceptable values.

The Close Session Operation does not fail.

4.9.6.5 *Notes*

Various other headers may be transmitted with the Close Session Response; these do not affect the result of the test.

4.9.7 Test S-E-7: Set Session Timeout Fails (No Reliable Session)

Demonstrates the transmission of a Session PDU to set the timeout of the reliable OBEX session, but no reliable OBEX Session exists at the time the request is issued. As a result, the OBEX Set Session Timeout request is rejected.

4.9.7.1 *Test Status*

Accepted

4.9.7.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Set Session Timeout Request. The following must occur:

- The Set Session Timeout Request with the Final Bit set is sent (0x87).
- The Set Session Timeout Request packet length will be 9 bytes (0x0009).
- The Set Session Timeout Request contains a properly formed **Session Parameters** (0x52) header as the first header in the packet. The length of this header will be 6 bytes (0x0006).
- The Session Parameters header must contain the *Session Opcode* (0x05) tag/length/value triplet. The *Session Opcode* triplet must be the first triplet in the header and must have a one-byte length (0x01) and the Set Timeout (0x04) opcode as its value.
- No other headers are sent.

The OBEX Client receives a Set Session Timeout Response.

See **A.8.7** for a complete diagram of the event sequence chart.

4.9.7.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.9.7.4 *Expected Outcome*

4.9.7.4.1 Pass Verdict

The OBEX Server receives a Set Session Timeout Request.

The OBEX Server transmits a Set Session Timeout Response. The following are true:

- The entire Session PDU does not exceed the default OBEX packet size of 255 bytes.
- The **Forbidden** (0xC3) response code is sent.
- The Final bit is set.
- The Session Packet Length indicates the correct size.

The Set Session Timeout Operation is unsuccessful.

4.9.7.4.2 Fail Verdict

The OBEX Server does not transmit a Set Session Timeout Response.

The Set Session Timeout Response does not contain acceptable values.

The Set Session Timeout Operation does not fail.

4.9.7.5 *Notes*

Various other headers may be transmitted with the Set Session Timeout Response; these do not affect the result of the test.

4.10 *OBEX Headers*

4.10.1 Test S-H-1: Tiny TP Split Header

Demonstrate that OBEX headers broken over Tiny TP packet boundaries are properly handled. This should be demonstrated by sending a **GET** response with the **Body/End Of Body** header split into two TinyTP packets. The device should properly receive the complete object.

4.10.1.1 *Test Status*

Accepted

4.10.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

During the transport connection, during the SNRM frame, a 64-byte packet size should be requested. This will ensure that the TinyTP packet size is restricted to 64 bytes. This limits the OBEX Server data per TinyTP packet to 58 bytes (64 bytes – 2 (IrLMP) – 1 (TinyTP) – 3 (OBEX) = 58 bytes).

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The size of the object being requested must be at least 59 bytes in length. The length of this header is a two-byte value including 3 bytes of header description.
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.10.1.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

4.10.1.4 *Expected Outcome*

4.10.1.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- The Get Packet Length indicates the correct size.
- The Final Bit is set.
- Each Get Response should contain the Continue response code unless it is the last response packet. The final Get Response contains the Success response code instead.
- The Get Response PDU contains a properly formed **Body/End Of Body** header that is at least 59 bytes in length, as this will assure two TinyTP packets are used to transmit this header. This is guaranteed with the requirement that the TinyTP packet size is 64 bytes.

The Get Operation is successful.

4.10.1.4.2 Fail Verdict

The OBEX Client does not transmit the Get Responses.

The Get Responses do not contain acceptable values.

The Get Operation does not handle the use of the **Body/End Of Body** headers properly.

The Get Operation is not successful.

4.10.1.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.10.2 Test S-H-2: One-Byte Headers

Demonstrate that one-byte style OBEX headers are properly formed when transmitted and properly handled when received. Since OBEX defines only one one-byte header, the Session-Sequence-Number header, the only way to guarantee that both the Client and Server will send a one-byte header is to establish a reliable OBEX session and then perform an operation. After the reliable session has been created, an OBEX Connect operation will be issued.

4.10.2.1 *Test Status*

Accepted

4.10.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client will create a reliable OBEX session. See **4.9.1.2** for the full description of this process.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is nine bytes (0x0009).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Request contains a properly formed **Session-Sequence-Number** (0x93) header as the first and only header in the packet. The length of this header is two bytes. The value of this header will include the current sequence number (0x00).

- No other headers are included.

See **A.10.1** for a complete diagram of the event sequence chart.

4.10.2.3 *Test Condition*

If support for the Session PDU is not present, this test may be **SKIPPED**.

4.10.2.4 *Expected Outcome*

4.10.2.4.1 Pass Verdict

The OBEX Server receives a Create Session Request.

The OBEX Server transmits a Create Session Response.

Create Session operation is successful.

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The Connect Response contains a properly formed **Session-Sequence-Number** (0x93) header as the first header in the packet. The length of this header is two bytes. The value of this header will include the next sequence number (0x01).
- OBEX Connect operation is successful.The Connect operation successfully handled sending and receiving one-byte **Session-Sequence-Number** headers.

4.10.2.4.2 Fail Verdict

The OBEX Server does not transmit a Create Session Response.

Create Session operation is not successful.

The OBEX Server does not transmit a Connect Response with a properly formed Session-Sequence-Number header.

OBEX Connect operation is not successful.

4.10.2.5 *Notes*

Various other headers may be transmitted with the Create Session Response or Connect Response; these do not affect the result of the test.

4.10.3 Test S-H-3: Four-Byte Headers

Demonstrate that four-byte style OBEX headers are properly formed when transmitted and properly handled when received. This should be demonstrated by receiving a **GET** request with a **Count** header and sending a **GET** response with a **Length** header.

4.10.3.1 *Test Status*

Accepted

4.10.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.
- The first Get Request also contains a four-byte **Count** (0xC0) header. The length of the header is five bytes (0x0005). Its contents will be 0x00000001.

- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.10.3.3 *Test Condition*

Support for four-byte headers is **REQUIRED**.

4.10.3.4 *Expected Outcome*

4.10.3.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- A Get Response PDU contains a properly formed four-byte **Length** header.

The Get Operation is successful.

- The Get operation successfully handled sending a four-byte **Length** header and receiving a four-byte **Count** header.

4.10.3.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The Get Operation is not successful

The Get operation does not handle the use of the **Length** and **Count** headers properly.

4.10.3.4.3 Inconclusive Verdict

A Get Response does not include a **Length** header. Some devices might not include a Length header, as it is an optional header for a normal Get operation.

4.10.3.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.10.4 Test S-H-4: Byte Sequence Headers

Demonstrate that byte sequence style OBEX headers are properly formed when transmitted and properly handled when received. This should be demonstrated by receiving a **GET** request with an **HTTP** header and sending a **GET** response with an **End Of Body** header.

4.10.4.1 *Test Status*

Accepted

4.10.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

Get Request(s) is/are transmitted by the OBEX Client. The following must occur:

- A Get Request packet with the Final Bit set is sent (0x83).
- The Get Request packet length for the first request sent will be 255 bytes or less, but will vary based on the size of the Name header.
- The Get Request packet length for all subsequent requests is 3 bytes (0x0003).
- The first Get Request contains a properly formed **Name** (0x01) header. This Name header must indicate a valid object on the OBEX Server device, encoded as null-terminated Unicode data. The length of this header is a two-byte value including 3 bytes of header description.

- The first Get Request also contains an **HTTP** (0x47) byte sequence header. The length of the header is eleven bytes (0x000B). Its contents will be “Testing” (with null-termination) displayed in hexadecimal as follows:

  54657374 696E6700
- No other headers are included.

Get Response(s) is/are received by the OBEX Client. The OBEX Client will respond with a Get Request packet for every Get Response packet received with the Continue response code (0x90).

See **A.4.2** for a complete diagram of the event sequence chart.

4.10.4.3 *Test Condition*

Support for Byte Sequence headers is **REQUIRED**.

4.10.4.4 *Expected Outcome*

4.10.4.4.1 Pass Verdict

Get Request(s) is/are received by the OBEX Server.

Get Response(s) is/are transmitted by the OBEX Server. The following are true:

- A Get Response PDU contains a properly formed **End Of Body** byte sequence header.

The Get Operation is successful.

- The Get operation successfully handled sending an **End Of Body** byte sequence header and receiving an **HTTP** byte sequence header.

4.10.4.4.2 Fail Verdict

The OBEX Server does not transmit the Get Responses.

The OBEX Server does not transmit a Get Response with an **End Of Body** byte sequence header.

The Get Operation is not successful

The Get operation does not handle the use of the **End Of Body** and **HTTP** headers properly.

4.10.4.5 *Notes*

Various other headers may be transmitted with the Get Responses; these do not affect the result of the test.

4.10.5 Test S-H-5: Unicode Headers

Demonstrate that UNICODE style OBEX headers are properly handled when received. This includes verification that the header data is null-terminated. This should be demonstrated by receiving a **PUT** request with a **Name** header. The OBEX Server cannot be forced into sending a UNICODE header, so the proper formation of transmitted UNICODE headers cannot be verified.

4.10.5.1 *Test Status*

Accepted

4.10.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename

acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.

- The Put Request contains a properly formed **Length** (0xC3) header. The Length header is 5 bytes and will contain the length of the object to be transferred (25 bytes or 0x00000019).
- The Put Request contains a properly formed **Body** (0x48) header. This Body header will contain 12-bytes of random data forming the object being sent. The length of this header is 15 bytes (0x000F).
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 13-bytes of random data forming the rest of the object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 16 bytes (0x0010).
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

4.10.5.3 *Test Condition*

Support for Unicode headers is **REQUIRED**.

4.10.5.4 *Expected Outcome*

4.10.5.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response.

The Put Operation is successful.

- The Put operation successfully handled receiving a null-terminated, Unicode **Name** header.

4.10.5.4.2 Fail Verdict

The OBEX Server does not transmit a Put Response.

The Put Operation is not successful

The Put operation does not handle the use of the **Name** header properly.

4.10.5.5 *Notes*

Various other headers may be transmitted with the Put Response; these do not affect the result of the test.

4.11 *OBEX Authentication*

4.11.1 Test S-AU-1: Authenticate Client Connection

Demonstrates that the OBEX Client can successfully authenticate the OBEX Server during an OBEX Connect operation.

4.11.1.1 *Test Status*

Accepted

4.11.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is 31 bytes (0x001F).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).

- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Request will contain an **Authenticate Challenge** (0x4D) byte sequence header. The length of the header is 24 bytes (0x0018).

  The contents of this header will include the following two tag/length/value triplets:

  Nonce – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. A 4-byte unique time-stamp, followed by a colon, and a private key known only to the sender are the values passed into the MD5 algorithm to generate the nonce (i.e. Nonce = MD5(time-stamp ":" private-key)).

  The format for this field is as follows:

  ```
  00 10 <nonce>                    - Hexadecimal values
  ```

  Options – this value indicates that neither of the two defined authentication options is required.

  The format for this field is as follows:

  ```
  01 01 00                         - Hexadecimal values
  ```
- No other headers are included.

See **A.1.1** for a complete diagram of the event sequence chart.

4.11.1.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

4.11.1.4 *Expected Outcome*

4.11.1.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The Success response code is sent.
- The OBEX Server device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Response header.
- The Connect Response contains an **Authenticate Response** byte sequence header. This header must contain a properly formed **Request Digest** field and can optionally contain the **UserId** and **Nonce** fields.

The Connect Operation succeeds.

The OBEX Client's 16-byte Nonce, followed by a colon, and the password used on the OBEX Server should be passed into the MD5 algorithm and compared against the request digest transmitted in the OBEX Server's Authentication Response header. If the request digest matches the MD5 result, the OBEX Client Authentication is a success.

4.11.1.4.2 Fail Verdict

The OBEX Server does not transmit a Connect Response.

The Connect Response does not contain acceptable values.

The Connect Operation does not succeed.

If the request digest does not match the MD5 result, as described above, the OBEX Client Authentication is a failure.

4.11.1.4.3 Inconclusive Verdict

4.11.1.5 *The Connect Operation cannot respond to the OBEX Authentication request.Notes*

Various other headers may be transmitted with the Connect Response; these do not affect the result of the test.

4.11.2 Test S-AU-2: Authenticate Client Operation

Demonstrates that the OBEX Client can successfully authenticate the OBEX Server for an OBEX Put operation.

4.11.2.1 *Test Status*

Accepted

4.11.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet without the Final Bit set is sent (0x02).
- The Put Request packet length will be 27 bytes (0x001B).
- The Put Request will contain an **Authenticate Challenge** (0x4D) byte sequence header. The length of the header is 24 bytes (0x0018).

  The contents of this header will include the following two tag/length/value triplets:

  Nonce – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. A 4-byte unique time-stamp, followed by a colon, and a private key known only to the sender are the values passed into the MD5 algorithm to generate the nonce (i.e. Nonce = MD5(time-stamp ":" private-key)).

  The format for this field is as follows:

  ```
  00 10 <nonce>                    - Hexadecimal values
  ```

  Options – this value indicates that neither of the two defined authentication options is required.

  The format for this field is as follows:

  ```
  01 01 00                         - Hexadecimal values
  ```

- No other headers are included.

The OBEX Client receives a Put Response. The following must occur:

- The Put operation contains the Continue (0x90) response code.
- The Put Response contains an **Authenticate Response** byte sequence header.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The Put Request contains a properly formed **Body** (0x48) header. This Body header will contain 12-bytes of random data forming the object being sent. The length of this header is 15 bytes (0x000F).
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 13-bytes of random data forming the rest of the object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 16 bytes (0x0010).
- No other headers are included.

See **A.5.1** for a complete diagram of the event sequence chart.

4.11.2.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

4.11.2.4 *Expected Outcome*

4.11.2.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following are true:

- The Continue response code is sent.
- The OBEX Server device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Response header.
- The Put Response contains an **Authenticate Response** byte sequence header. This header must contain a properly formed **Request Digest** field and can optionally contain the **UserId** and **Nonce** fields.

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following are true:

- The Success response code is sent.

The Put Operation succeeds.

The OBEX Client's 16-byte Nonce, followed by a colon, and the password used on the OBEX Server should be passed into the MD5 algorithm and compared against the request digest transmitted in the OBEX Server's Authentication Response header. If the request digest matches the MD5 result, the OBEX Client Authentication is a success.

4.11.2.4.2 Fail Verdict

The OBEX Server does not transmit the Put Responses.

The Put Responses do not contain acceptable values.

The Put Operation does not succeed.

If the request digest does not match the MD5 result, as described above, the OBEX Client Authentication is a failure.

4.11.2.4.3 Inconclusive Verdict

The Put Operation cannot respond to the OBEX Authentication request.

4.11.2.5 *Notes*

Various other headers may be transmitted with the Put Responses; these do not affect the result of the test.

4.11.3 Test S-AU-3: Authenticate Server Connection

Demonstrates that the OBEX Server can successfully authenticate the OBEX Client during an OBEX Connect operation.

4.11.3.1 *Test Status*

Accepted

4.11.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length is 7 bytes (0x0007).
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).

The OBEX Client receives a Connect Response. The following must occur:

- The Connect operation is aborted with the receipt of the Unauthorized (0xC1) response code.
- An **Authentication Challenge** (0x4D) header is received containing at least the 16-byte Nonce value.

The OBEX Client transmits a Connect Request. The following must occur:

- The Connect Request has the Final Bit is set (0x80).
- The Connect Request packet length will be 255 bytes or less, but will vary based on the size of the Authentication Response header.
- The Protocol field is OBEX version 1.0 (0x10).
- The Flags field is (0x00).
- The Maximum OBEX Packet Len is 5120 (5K) bytes (0x1400).
- The Connect Request will contain an **Authenticate Response** (0x4E) byte sequence header in response to the Authentication Challenge header received from the OBEX Server. The length of the header will vary based on the tag/length/value triplets being returned.

  The contents of this header will include the following:

  Request Digest – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. The 16-byte Nonce from the Authentication Challenge header in the Connect Response, followed by a colon, and the password used on the OBEX Server are the values passed into the MD5 algorithm to generate the request digest (i.e. Request Digest = MD5(Nonce ":" Password)).

  The format for this field is as follows:

00 10 <Request-Digest>	- Hexadecimal values

  UserId – this value is optionally included based on the options field received in the Authentication Challenge header in the Connect Response. If the options field has the first bit set, the userId is required. In order to send this value, you will need to know the userId used on the OBEX Server and provide it in this field.

  The format for this field is as follows:

01 <varies, up to 20 bytes> <UserId>	- Hexadecimal values

  Nonce – this value is the 16-byte nonce received in Authentication Challenge header in the Connect Response.

  The format for this field is as follows:

02 10 <Nonce>	- Hexadecimal values

- No other headers are included.

See **A.1.2** for a complete diagram of the event sequence chart.

4.11.3.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

4.11.3.4 *Expected Outcome*

4.11.3.4.1 Pass Verdict

The OBEX Server receives a Connect Request.

The OBEX Server transmits a Connect Response. The following are true:

- The Unauthorized reason code is returned, since the OBEX Server requires OBEX Authentication.
- The OBEX Server device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Challenge header.
- The Connect Response contains an **Authenticate Challenge** byte sequence header. This header must contain a properly formed **Nonce** field and can optionally contain the **Options** and **Realm** fields.

The OBEX Server receives a Connect Request. The following are true:

- An **Authentication Response** header is received with a request digest.

The OBEX Server transmits a Connect Response. The following are true:

- The Success response code is returned.

The Connect Operation is successful.

The OBEX Server Authentication procedure succeeds.

4.11.3.4.2 Fail Verdict

The OBEX Server does not transmit the Connect Responses.

The Connect Responses do not contain acceptable values.

The Connect Operation is not successful.

The OBEX Server Authentication procedure does not succeed.

4.11.3.4.3 Inconclusive Verdict

The OBEX Server cannot initiate OBEX Authentication during the Connect operation.

4.11.3.5 *Notes*

Various other headers may be transmitted with the Connect Responses; these do not affect the result of the test.

4.11.4 Test S-AU-4: Authenticate Server Operation

Demonstrates that the OBEX Server can successfully authenticate the OBEX Client for an OBEX Put operation.

4.11.4.1 *Test Status*

Accepted

4.11.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet without the Final Bit set is sent (0x02).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- No other headers are included.

The OBEX Client receives a Put Response. The following must occur:

- The Put operation is aborted with the receipt of the Unauthorized (0xC1) response code.
- An **Authentication Challenge** (0x4D) header is received containing at least the 16-byte Nonce value.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet without the Final Bit set is sent (0x02).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name and Authentication Response headers.
- The Put Request will contain an **Authenticate Response** (0x4E) byte sequence header in response to the Authentication Challenge header received from the OBEX Server. The length of the header will vary based on the tag/length/value triplets being returned.

  The contents of this header will include the following:

Request Digest – this value is a 16-byte value generated from the MD5 algorithm described in the OBEX specification. The 16-byte Nonce from the Authentication Challenge header in the Put Response, followed by a colon, and the password used on the OBEX Server are the values passed into the MD5 algorithm to generate the request digest (i.e. Request Digest = MD5(Nonce ":" Password)).

The format for this field is as follows:

```
00 10 <Request-Digest>                    - Hexadecimal values
```

UserId – this value is optionally included based on the options field received in the Authentication Challenge header in the Put Response. If the options field has the first bit set, the userId is required. In order to send this value, you will need to know the userId used on the OBEX Server and provide it in this field.

The format for this field is as follows:

```
01 <varies, up to 20 bytes> <UserId>      - Hexadecimal values
```

Nonce – this value is the 16-byte nonce received in Authentication Challenge header in the Put Response.

The format for this field is as follows:

```
02 10 <Nonce>                             - Hexadecimal values
```

- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- No other headers are included.

The OBEX Client receives a Put Response. The following must occur:

- The Put operation contains the Continue (0x90) response code.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 34 bytes (0x0022).
- The Put Request contains a properly formed **Body** (0x48) header. This Body header will contain 12-bytes of random data forming the object being sent. The length of this header is 15 bytes (0x000F).
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 13-bytes of random data forming the rest of the object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 16 bytes (0x0010).
- No other headers are included.

See **A.5.2** for a complete diagram of the event sequence chart.

4.11.4.3 *Test Condition*

If support for OBEX Authentication is not present, this test may be **SKIPPED**.

4.11.4.4 *Expected Outcome*

4.11.4.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following are true:

- The Unauthorized reason code is returned, since the OBEX Server requires OBEX Authentication.
- The OBEX Server device should prompt the user to enter a password, or at least inform the user of the password being used. This password will be used in formulating the Authentication Challenge header.

- The Put Response contains an **Authenticate Challenge** byte sequence header. This header must contain a properly formed **Nonce** field and can optionally contain the **Options** and **Realm** fields.

The OBEX Server receives a Put Request. The following are true:

- An **Authentication Response** header is received with a request digest.

The OBEX Server transmits a Put Response. The following are true:

- The Continue response code is returned.

The OBEX Server receives a Put Request.

The OBEX Server transmits a Put Response. The following are true:

- The Success response code is returned.

The Put Operation is successful.

The OBEX Server Authentication procedure succeeds.

4.11.4.4.2 Fail Verdict

The OBEX Server does not transmit the Put Responses.

The Put Responses do not contain acceptable values.

The Put Operation is not successful.

The OBEX Server Authentication procedure does not succeed.

4.11.4.4.3 Inconclusive Verdict

The OBEX Server cannot initiate OBEX Authentication during the Put operation.

4.11.4.5 *Notes*

Various other headers may be transmitted with the Put Responses; these do not affect the result of the test.

4.12 *Miscellaneous Tests*

4.12.1 Test S-OP-1: Invalid OBEX Opcode

Invalid opcode test: Demonstrate that an unknown or user-defined operation request receives a “Not Implemented” (0xD1) response code from the server.

4.12.1.1 *Test Status*

Accepted

4.12.1.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a request for a user-defined OBEX opcode (0x1A). The following must occur:

- A user-defined opcode request packet with the Final Bit set is sent (0x9A).
- The request packet length is 3 bytes (0x0003).

See **A.11.1** for a complete diagram of the event sequence chart.

4.12.1.3 *Test Condition*

Server Responses to invalid OBEX opcodes are **REQUIRED**.

4.12.1.4 *Expected Outcome*

4.12.1.4.1 Pass Verdict

The OBEX Server receives the user-defined OBEX opcode request.

The OBEX Server transmits a response. The following are true:

- The Not Implemented response code is returned.
- The packet length is three bytes.
- The Final Bit is set.

The user-defined OBEX opcode operation is unsuccessful.

4.12.1.4.2 Fail Verdict

The OBEX Server does not transmit the response.

The response does not contain acceptable values.

The user-defined OBEX opcode operation does not fail.

4.12.1.5 *Notes*

Various other headers may be transmitted with the response; these do not affect the result of the test.

4.12.2 Test S-IAS-1: Server IAS Query

Verify that the “OBEX” IAS entry is properly retrieved.

4.12.2.1 *Test Status*

Accepted

4.12.2.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client initiates a transport connection. This following will occur:

- Transmission of an IrLAP SNRM Command.
- Receipt of an IrLAP UA Response.
- Transmission of an IrLMP Connect Request. Source LSAP is the IAS Client LSAP (LSAP 1 will be used). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Receipt of an IrLMP Connect Response. Source LSAP is the IAS Server LSAP (LSAP 0). Destination LSAP is the IAS Client LSAP (LSAP 1 will be used).
- Transmission of an IrLMP Data packet with IAS query information. The following will occur:
  - Source LSAP is the IAS Client LSAP (LSAP 1 will be used).
  - Destination LSAP is the IAS Server (LSAP 0).
  - GetValueByClass IAS query is sent with the Last bit set (0x84).
  - “OBEX” IAS class name is sent in ASCII format. The length of the class name is 4 bytes (0x04).
  - “IrDA:TinyTP:LsapSel” IAS attribute is sent in ASCII format. The length of the attribute name is 19 bytes (0x13).

See **A.12.1** for a complete diagram of the event sequence chart.

4.12.2.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

4.12.2.4 *Expected Outcome*

4.12.2.4.1 Pass Verdict

The OBEX Server responds to the OBEX transport connection. The following will occur:

- Successful IrLAP and IrLMP connections.
- Receipt of an IrLMP Data packet containing an IrIAS query for the “OBEX” class name.

- Transmission of an IrLMP Data packet containing an IrIAS query response. The following are true:
  - Source LSAP is the IAS Server LSAP (LSAP 0)
  - Destination LSAP is the IAS Client LSAP (LSAP 1 will be used).
  - GetValueByClass IAS response is sent.
  - Success response code is returned.
  - Result count is returned.
  - Object ID value is returned.
  - An Integer value is returned.
  - A 32-bit signed OBEX LSAP value is returned. The value must be greater than 0x00 and less than 0x70.

The IAS query is successful with an OBEX LSAP value located.

4.12.2.4.2 Fail Verdict

The OBEX Server fails to transmit an IrLMP Data packet with the IAS query response.

The IAS query response contains unacceptable values.

The IAS query is unsuccessful.

4.12.2.5 *Notes*

N/A

4.12.3 Test S-TTP-1: Tiny TP Connect

Verify that the Tiny TP Connection is successful. The MaxSduSize parameter must not present in the Tiny TP Connect Response packet in order for the connection to be considered successful.

4.12.3.1 *Test Status*

Accepted

4.12.3.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client initiates a transport connection. This following will occur:

- Transmission of an IrLAP SNRM Command.
- Receipt of an IrLAP UA Response.
- Transmission of an IrLMP Connect Request. Source LSAP is the IAS Client LSAP (LSAP 1 will be used). Destination LSAP is the IAS Server LSAP (LSAP 0).
- Receipt of an IrLMP Connect Response. Source LSAP is the IAS Server LSAP (LSAP 0). Destination LSAP is the IAS Client LSAP (LSAP 1 will be used).
- Transmission of an IrLMP Data packet with IAS query information. The following will occur:
  - Source LSAP is the IAS Client LSAP (LSAP 1 will be used).
  - Destination LSAP is the IAS Server (LSAP 0).
  - GetValueByClass IAS query is sent with the Last bit set (0x84).
  - “OBEX” IAS class name is sent in ASCII format. The length of the class name is 4 bytes (0x04).
  - “IrDA:TinyTP:LsapSel” IAS attribute is sent in ASCII format. The length of the attribute name is 19 bytes (0x13).
- Receipt of an IrLMP Data packet with an IAS query response. Source LSAP is the IAS Server LSAP (LSAP 0). Destination LSAP is the IAS Client LSAP (LSAP 1 will be used).

- Transmission of an IrLMP Connect Request for the OBEX Service (TinyTP Connect Request). The following are true:
    - Source LSAP is the OBEX Client's OBEX LSAP (LSAP 2 will be used).
    - Destination LSAP is the OBEX Server's OBEX LSAP (varies).
    - The Parameter bit is clear.

See **A.12.2** for a complete diagram of the event sequence chart.

4.12.3.3 *Test Condition*

If OBEX does not use the IrDA transport, this test may be **SKIPPED**.

4.12.3.4 *Expected Outcome*

4.12.3.4.1 Pass Verdict

The OBEX Server responds to the OBEX transport connection. The following will occur:

- Successful IrLAP and IrLMP connections.
- Successful IAS Query for the "OBEX" class name.
- Receipt of an IrLMP Connect Request (TinyTP Connect Request).
- Transmission of an IrLMP Connect Response (TinyTP Connect Response). The following are true:
    - Source LSAP is the advertised OBEX LSAP (varies).
    - Destination LSAP is the OBEX Client's OBEX LSAP (LSAP 2 will be used).
    - The Parameter bit is clear.
    - No MaxSduSize parameter exists.

The TinyTP Connection to the OBEX Service is successful.

4.12.3.4.2 Fail Verdict

The OBEX Server fails to transmit an IrLMP Connect Response packet for the OBEX Service (TinyTP Connect Response).

The TinyTP Connect Response contains unacceptable values.

The TinyTP Connection is unsuccessful.

4.12.3.5 *Notes*

N/A

4.12.4 Test S-UP-1: No Response to Ultra Put

Demonstrate that the server does not respond to an Ultra Put request.

4.12.4.1 *Test Status*

Accepted

4.12.4.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.

- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 25-bytes of random data forming the entire object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 28 bytes (0x001C).
- No other headers are included.

See **A.5.4** for a complete diagram of the event sequence chart.

4.12.4.3 *Test Condition*

If support for Ultra is not present, this test may be **SKIPPED**.

4.12.4.4 *Expected Outcome*

4.12.4.4.1 Pass Verdict

The OBEX Server receives a Put Request.

The OBEX Server **does not** transmit a Put Response, as a Put operation over Ultra does not follow the standard request/response model used by a full OBEX implementation.

The Put Operation is successful.

4.12.4.4.2 Fail Verdict

The OBEX Server does not receive a Put Request.

The OBEX Server transmits a Put Response.

The Put Operation is not successful.

4.12.4.5 *Notes*

N/A

4.12.5 Test S-UP-2: Successful Ultra Put

Demonstrate that a server successfully receives an object sent via an Ultra Put.

4.12.5.1 *Test Status*

Accepted

4.12.5.2 *Test Procedure*

Initially, the OBEX Client and Server devices should be powered up and brought to the ready state. The ready state will encompass all devices that do not have an existing transport connection and are not currently processing any OBEX operations.

The OBEX Client transmits a Put Request. The following must occur:

- A Put Request packet with the Final Bit set is sent (0x82).
- The Put Request packet length will be 255 bytes or less, but will vary based on the size of the Name header.
- The Put Request contains a properly formed **Name** (0x01) header. This Name header will indicate the name of the object being transferred and must indicate a filename acceptable on the OBEX Server device. This header is encoded as null-terminated Unicode data and has a two-byte length value including 3 bytes of header description.
- The Put Request contains a properly formed **End Of Body** (0x49) header. This Body header will contain 25-bytes of random data forming the entire object being sent. This is the last header in the packet and signifies the end of the object. The length of this header is 28 bytes (0x001C).
- No other headers are included.

See **A.5.4** for a complete diagram of the event sequence chart.

4.12.5.3 *Test Condition*

If support for Ultra is not present, this test may be **SKIPPED**.

4.12.5.4 *Expected Outcome*

4.12.5.4.1 Pass Verdict

The OBEX Server receives a Put Request. The following are true:

- A 25-byte object should have been received and stored according to the name provided by the OBEX Client.

The Put Operation is successful.

4.12.5.4.2 Fail Verdict

The OBEX Server does not receive a Put Request.

The OBEX Server does not receive and store the 25-byte object sent by the OBEX Client.

The Put Operation is not successful.

4.12.5.5 *Notes*

N/A

A Diagrams

A.1 *OBEX Connect State Charts*

A.1.1 Simple Connect Operation

Transport connection established between Client and Server (see A.12.2)

Event	OBEX Client		OBEX Server
1	Connect Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test Complete

A.1.2 Authenticate Server Connect Operation

Transport connection established between Client and Server (see A.12.2)

Event	OBEX Client		OBEX Server
1	Connect Req + Final Bit	→	
2		←	Unauthorized Rsp + Final Bit
3	Connect Req + Final Bit	→	
4		←	Success Rsp + Final Bit

Test complete

A.2 *OBEX Disconnect State Charts*

A.2.1 Simple Disconnect Operation

Transport connection established between Client and Server (see A.12.2)

Event	OBEX Client		OBEX Server
1	Connect Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Disconnect Req + Final Bit	→	
4		←	Success Rsp + Final Bit

Test Complete

A.3 *OBEX SetPath State Charts*

A.3.1 Simple SetPath Operation

Transport connection established between Client and Server (see A.12.2)

Event	OBEX Client		OBEX Server
1	SetPath Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test Complete

A.4 *OBEX Get State Charts*

A.4.1 Simple Client Get Operation

Event	OBEX Client		OBEX Server
		Transport connection established between Client and Server (see A.12.2)	
		Events 1-2 repeat 0 to n times	
1	Get Req	→	
2		←	Continue Rsp + Final Bit
		Events 3-4 repeat 0 to n times	
3	Get Req + Final Bit	→	
4		←	Continue Rsp + Final Bit
5	Get Req + Final Bit	→	
6		←	Success Rsp + Final Bit
		Test complete	

A.4.2 Simple Server Get Operation

Event	OBEX Client		OBEX Server
		Transport connection established between Client and Server (see A.12.2)	
		Events 1-2 repeat 0 to n times	
1	Get Req + Final Bit	→	
2		←	Continue Rsp + Final Bit
3	Get Req + Final Bit	→	
4		←	Success Rsp + Final Bit
		Test complete	

A.4.3 Server Get Operation Specified Packet Size

Event	OBEX Client		OBEX Server
		Transport connection established between Client and Server (see A.12.2)	
1	Connect Req + Final Bit	→	
2		←	Success Rsp + Final Bit
		Events 1-2 repeat 0 to n times	
3	Get Req + Final Bit	→	
4		←	Continue Rsp + Final Bit
5	Get Req + Final Bit	→	
6		←	Success Rsp + Final Bit
		Test Complete	

A.5 *OBEX Put State Charts*

A.5.1 Simple Put Operation

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

Events 1-2 repeat 0 to n times

1	Put Req	→	
2		←	Continue Rsp + Final Bit
3	Put Req + Final Bit	→	
4		←	Success Rsp + Final Bit

Test Complete

A.5.2 Authenticate Server Put Operation

Event	OBEX Client		OBEX Server
	Transport connection established between Client and Server (see A.12.2)		
1	Put Req	→	
2		←	Unauthorized Rsp + Final Bit
	Events 1-2 repeat 0 to n times		
3	Put Req	→	
4		←	Continue Rsp + Final Bit
5	Put Req + Final Bit	→	
6		←	Success Rsp + Final Bit

Test Complete

A.5.3 Put Operation Using Specified Packet Size

Event	OBEX Client		OBEX Server
	Transport connection established between Client and Server (see A.12.2)		
1	Connect Req + Final Bit	→	
2		←	Success Rsp + Final Bit
	Events 1-2 repeat 0 to n times		
3	Put Req	→	
4		←	Continue Rsp + Final Bit
5	Put Req + Final Bit	→	
6		←	Success Rsp + Final Bit

Test Complete

A.5.4 Ultra Put Operation

Event	OBEX Client		OBEX Server
	No Transport connection		
1	Put Req + Final Bit	→	

Test Complete

A.6 *OBEX Abort State Charts*

A.6.1 Put Abort

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

Cancel Option 1 or 2 ***MUST*** *occur. When a cancel option has been performed, the test should jump to Event 6 for indication of the Abort response packet.*

Events 1-4 repeat 1 to n times

1	Put Req	→	
2	Abort Req + Final Bit (Cancel Option 1)	→	
3		←	Continue Rsp + Final Bit
4	Abort Req + Final Bit (Cancel Option 2)	→	
5	Put Req + Final Bit	→	
6		←	Success Rsp + Final Bit

Test complete

A.6.2 Get Abort

Event	OBEX Client		OBEX Server
	Transport connection established between Client and Server (see A.12.2)		
	Cancel Option 1,2,3, or 4 ***MUST*** *occur. When a cancel option has been performed, the test should jump to Event 6 for indication of the Abort response packet.*		
	Events 1-4 repeat 0 to n times		
1	Get Req	→	
2	Abort Req + Final Bit (Cancel Option 1)	→	
3		←	Continue Rsp + Final Bit
4	Abort Req + Final Bit (Cancel Option 2)	→	
	Events 5-8 repeat 1 to n times		
5	Get Req + Final Bit	→	
6	Abort Req + Final Bit (Cancel Option 3)	→	
7		←	Continue Rsp + Final Bit
8	Abort Req + Final Bit (Cancel Option 4)	→	
9	Get Req + Final Bit	→	
10		←	Success Rsp + Final Bit

Test complete

A.7 *OBEX Session State Charts*

A.7.1 Create Session Operation

Event	OBEX Client		OBEX Server
	Transport connection established between Client and Server (see A.12.2)		
1	Create Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test complete

A.7.2 Close Session Operation (Active Session)

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Close Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test complete

A.7.3 Close Session Operation (Suspended Session)

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Event	**OBEX Client**		**OBEX Server**
1	Suspend Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Close Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit

Test complete

A.7.4 Suspend Session Operation

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Event	**OBEX Client**		**OBEX Server**
1	Suspend Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test complete

A.7.5 Resume Session Operation

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Event	**OBEX Client**		**OBEX Server**
1	Suspend Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Resume Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit

Test complete

A.7.6 Unexpected Resume Session Operation

Event	**OBEX Client**		**OBEX Server**

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Unexpected Transport disconnect occurs.
*OBEX Reliable Session **must** be automatically suspended.*

Transport connection is re-established between Client and Server

1	Resume Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test complete

A.7.7 Set Session Timeout Operation

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Set Timeout Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit

Test complete

A.7.8 Infinite Suspend Timer

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Set Timeout Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Suspend Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit
5	Resume Sesssion Req + Final Bit	→	
6		←	Success Rsp + Final Bit

Test complete

A.7.9 Suspend Session Timer Expiration (Set Session Timeout)

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Set Timeout Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Suspend Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit
5	Resume Sesssion Req + Final Bit	→	
6		←	Service Unavailable Rsp + Final Bit

Test complete

A.7.10 Suspend Session Timer Expiration (Create Session)

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Suspend Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Resume Sesssion Req + Final Bit	→	
4		←	Service Unavailable Rsp + Final Bit

Test complete

A.7.11 Multiple Sessions

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Event	OBEX Client		OBEX Server
1	Suspend Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Create Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit
5	Close Session Req + Final Bit	→	
6		←	Success Rsp + Final Bit
7	Close Session Req + Final Bit	→	
8		←	Success Rsp + Final Bit

Test complete

A.8 *OBEX Session State Charts*

A.8.1 Create Session (No Sessions Available)

Transport connection established between Client and Server (see A.12.2)

Events 1-4 repeat 0 to n times

Event	OBEX Client		OBEX Server
1	Create Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Suspend Session Req + Final Bit	→	
4		←	Success Rsp + Final Bit
5	Create Session Req + Final Bit	→	
6		←	Service Unavailable Rsp + Final Bit

Test complete

A.8.2 Create Session (Session Already Active)

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

1	Create Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Create Session Req + Final Bit	→	
4		←	Forbidden Rsp + Final Bit

Test complete

A.8.3 Resume Session (Bad Session ID)

Transport connection established between Client and Server (see A.12.2)

Event	**OBEX Client**		**OBEX Server**
1	Resume Session Req + Final Bit	→	
2		←	Service Unavailable Rsp + Final Bit

Test complete

A.8.4 Resume Session (Session Already Active)

Transport connection established between Client and Server (see A.12.2)

Event	**OBEX Client**		**OBEX Server**
1	Create Session Req + Final Bit	→	
2		←	Success Rsp + Final Bit
3	Suspend Sesson Req + Final Bit	→	
4		←	Success Rsp + Final Bit
5	Create Session Req + Final Bit	→	
6		←	Success Rsp + Final Bit
7	Resume Session Req + Final Bit	→	
8		←	Forbidden Rsp + Final Bit

Test complete

A.8.5 Suspend Session (No Active Session)

Transport connection established between Client and Server (see A.12.2)

Event	**OBEX Client**		**OBEX Server**
1	Suspend Session Req + Final Bit	→	
2		←	Forbidden Rsp + Final Bit

Test complete

A.8.6 Close Session (No Active Session)

Transport connection established between Client and Server (see A.12.2)

Event	**OBEX Client**		**OBEX Server**
1	Close Session Req + Final Bit	→	
2		←	Forbidden Rsp + Final Bit

Test complete

A.8.7 Set Session Timeout (No Active Session)

Event	**OBEX Client**		**OBEX Server**

Transport connection established between Client and Server (see A.12.2)

1	Set Session Timeout Req + Final Bit	→	
2		←	Forbidden Rsp + Final Bit

Test complete

A.9 *OBEX Server Abort State Charts*

A.9.1 Server Reject Put Operation

Transport connection established between Client and Server (see A.12.2)

Events 1-2 repeat 0 to 50 times

Event	OBEX Client		OBEX Server
1	Put Req	→	
2		←	Continue Rsp + Final Bit
3	Put Req	→	
4		←	Forbidden Rsp + Final Bit

Test complete

A.9.2 Server Reject Get Operation

Transport connection established between Client and Server (see A.12.2)

Events 1-2 repeat 0 to n times

Event	OBEX Client		OBEX Server
1	Get Req + Final Bit	→	
2		←	Continue Rsp + Final Bit
3	Get Req + Final Bit	→	
4		←	Forbidden Rsp + Final Bit

Test complete

A.10 *OBEX Header State Charts*

A.10.1 One-Byte Headers

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

Event	OBEX Client		OBEX Server
1	Connect Req + Final Bit	→	
2		←	Connect Rsp + Final Bit

Test complete

A.11 *Invalid OBEX Opcode State Charts*

A.11.1 Invalid OBEX Opcode

Event	OBEX Client		OBEX Server

Transport connection established between Client and Server (see A.12.2)

1	User Defined (0x1A) + Final Bit	→	
2		←	Not Implemented Rsp + Final Bit

Test complete

A.12 *OBEX Transport Connection State Charts*

A.12.1 OBEX IAS Query

Event	OBEX Client		OBEX Server
1	IrLAP Connect Req (SNRM Cmd)	→	
2		←	IrLAP Connect Rsp (UA Rsp)
3	IrLMP Connect Req	→	
4		←	IrLMP Connect Rsp
5	IrLMP Data (IAS Query)	→	
6		←	IrLMP Data (IAS Response)

Test complete

A.12.2 TinyTP Connection

Event	OBEX Client		OBEX Server
1	IrLAP Connect Req (SNRM Cmd)	→	
2		←	IrLAP Connect Rsp (UA Rsp)
3	IrLMP Connect Req	→	
4		←	IrLMP Connect Rsp
5	IrLMP Data (IAS Query)	→	
6		←	IrLMP Data (IAS Response)
7	TinyTP Connect Req	→	
8		←	TinyTP Connect Rsp

Test complete

A.13 *Spurious Transport Disconnect State Charts*

A.13.1 Transport Disconnect During Put Operation

Event	OBEX Client	OBEX Server

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Put Req (with or without Final Bit)	→	

Unexpected Transport disconnection (Events 2-4)

2		←	IrLAP Request Disconnect (RD Rsp)
3	IrLAP Disconnect Req (DISC Cmd)	→	
4		←	IrLAP Disconnect Rsp (UA Rsp)

Transport connection reestablished (see A.12.2)

OBEX Reliable Session resumed between OBEX Client and Server (Events 5-6)

5	Resume Session Req + Final Bit	→	
6		←	Success Rsp + Final Bit

OBEX Put operation Resumed (Events 7-10)
First Put Req packet sent is a retransmission

Events 7-8 repeat 0 to n times

7	Put Req	→	
8		←	Continue Rsp + Final Bit
9	Put Req + Final Bit	→	
10		←	Success Rsp + Final Bit

Test Complete

A.13.2 Transport Disconnect During Get Operation

Event **OBEX Client** **OBEX Server**

Transport connection established between Client and Server (see A.12.2)

OBEX Reliable Session established between OBEX Client and Server (see A.7.1)

1	Get Req (with or without Final Bit)	→	

Unexpected Transport disconnection (Events 2-4)

2		←	IrLAP Request Disconnect (RD Rsp)
3	IrLAP Disconnect Req (DISC Cmd)	→	
4		←	IrLAP Disconnect Rsp (UA Rsp)

Transport connection reestablished (see A.12.2)

OBEX Reliable Session resumed between OBEX Client and Server (Events 5-6)

5	Resume Session Req + Final Bit	→	
6		←	Success Rsp + Final Bit

OBEX Get operation Resumed (Events 7-12)
First Get Req packet sent is a retransmission

Events 7-8 repeat 0 to n times

7	Get Req	→	
8		←	Continue Rsp + Final Bit

Events 9-10 repeat 0 to n times

9	Get Req + Final Bit	→	
10		←	Continue Rsp + Final Bit
11	Get Req + Final Bit	→	
12		←	Success Rsp + Final Bit

Test Complete