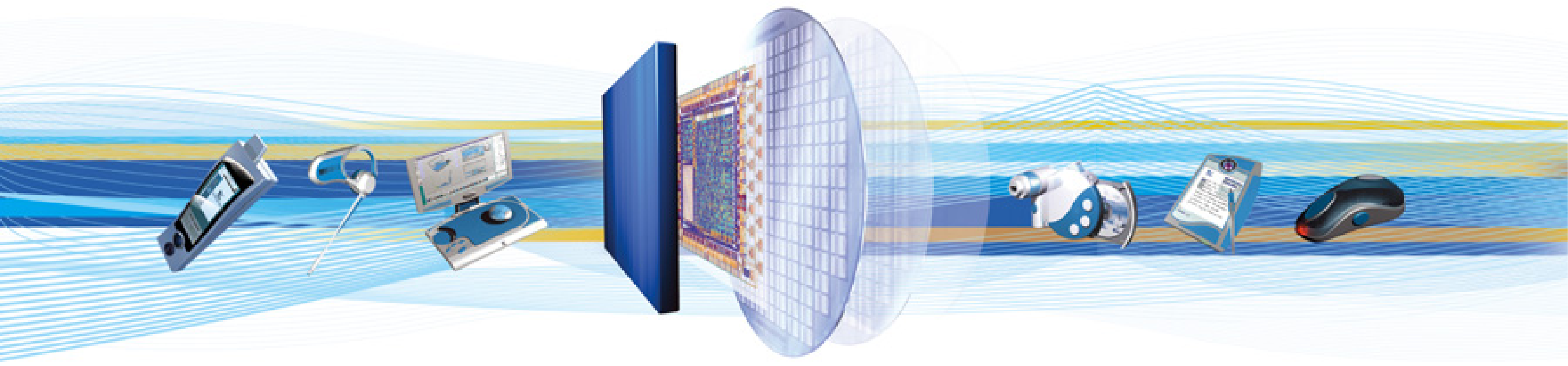
CSR Synergy®

CSR Synergy Bluetooth 18.2.0

PAC – Phone Book Access Client Profile

API Description

November 2011



Cambridge Silicon Radio Limited

Churchill House
Cambridge Business Park
Cowley Road
Cambridge CB4 0WZ
United Kingdom

Registered in England and Wales 3665875

Tel: +44 (0)1223 692000
Fax: +44 (0)1223 692001
www.csr.com

Bluetooth®

Contents

List of Figures

List of Tables

1 Introduction

1.1 Introduction and Scope

This document describes the message interface provided by the OBEX Phone Book Access Profile Client (PAC). The PAC conforms to the client side of the Phone Book Access Profile, ref. [PBAP].

1.2 Assumptions

The following assumptions and preconditions are made in the following:

- There is a secure and reliable transport between the profile part, i.e. PAC and the application
- The PAC shall only handle one request at the time
- Bonding (pairing) is NOT handled by the PAC

2 Description

2.1 Introduction

The scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a car-kit installed in car retrieves phone book objects from a mobile phone
- A second usage is a car-kit browsing the phone book virtual folder structure and copying single entries from the phone book

The PAC provides the following services to the application:

- Inquiry of devices
- Screening of those
- Connection handling
- OBEX protocol handling

The application is responsible for handling the requests and confirms from the PAC with correct data (object) as described in the IrOBEX specification. The PAC is not checking if the data is packed correctly with white spaces in the right places, for details see ref. [PBAP] and [OBEX].

2.2 Reference Model

The PAC interfaces to the connection management (CM).

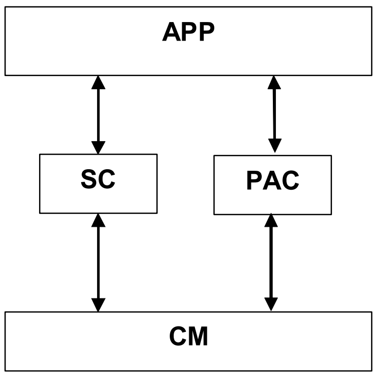


Figure 1: Reference model

2.3 Sequence Overview

If, in IDLE state, a connect request is received from the application, the PAC starts to connect to the specified device and the CONNECT state is entered, then the application receives a confirmation on the connect request. The application can issue a request to start the transfer for the object or folder browsing. The application can perform multiple requests (one at the time) before disconnecting the connection. When the application disconnects the service, the PAC re-enters IDLE state.

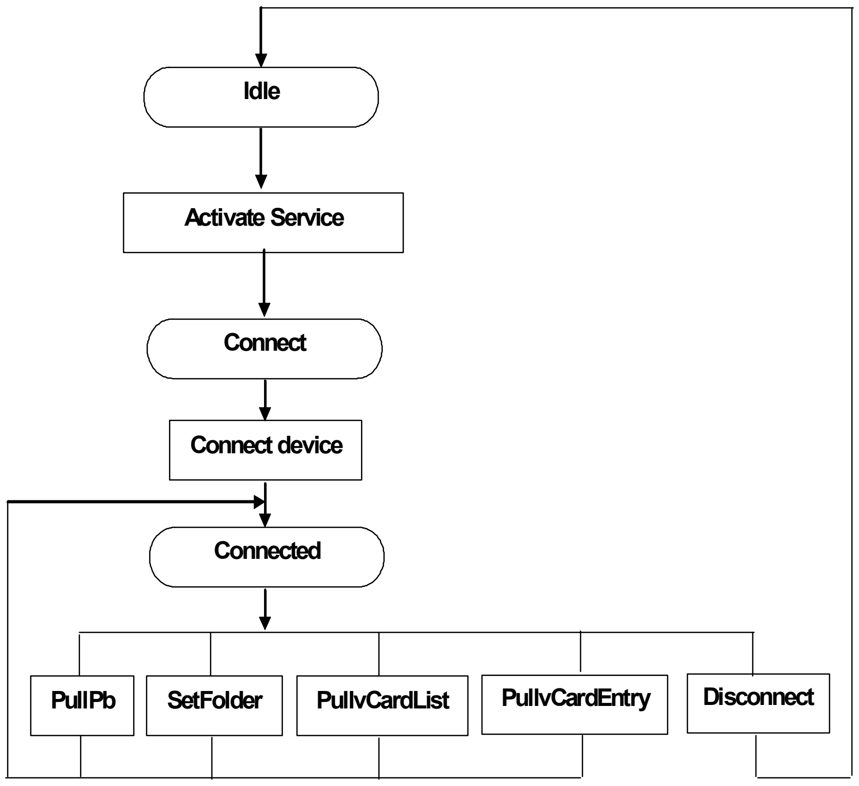


Figure 2: PAC state diagram

3 Interface Description

3.1 Connect

When the application wants to connect to a Phone Book Access Profile Server it has to send a CSR_BT_PAC_CONNECT_REQ to the PAC. In this message the application has to specify which device to connect to. The parameter 'authorize' is controlling if the PBAP client wants to OBEX-authenticate the PBAP server. If the parameter is TRUE the application has to specify the password parameter. This message also has a parameter called maxPacketSize, which indicates the maximum OBEX packet size, which the application wants to receive from the PBAP server side. The value can be between 255 bytes to 64Kbytes – 1, see definition in ref. [OBEX]. If the packet size is large it is optimizing for quick transfer, but the disadvantage will be use of big memory blocks.

The PAC sends a CSR_BT_PAC_CONNECT_CFM message to the application, which has the status of the connection establishment - this is the parameter result code. For success in the request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE, any other response code indicates a failure in the connection.

Once the OBEX connection is established, the PAC will, transparently for the application layer, make use of low power modes. This implies use of sniff if supported by the PBAP Server side. Low power modes are enabled using a supervision timer. If no data is received within the specified time interval, the PAC manager will attempt a change to low power mode if possible. The value of the timer is determined by the CSR_BT_PAC_LP_SUPERVISION_TIMEOUT and is defined in the csr_bt_pac_handler.h file.

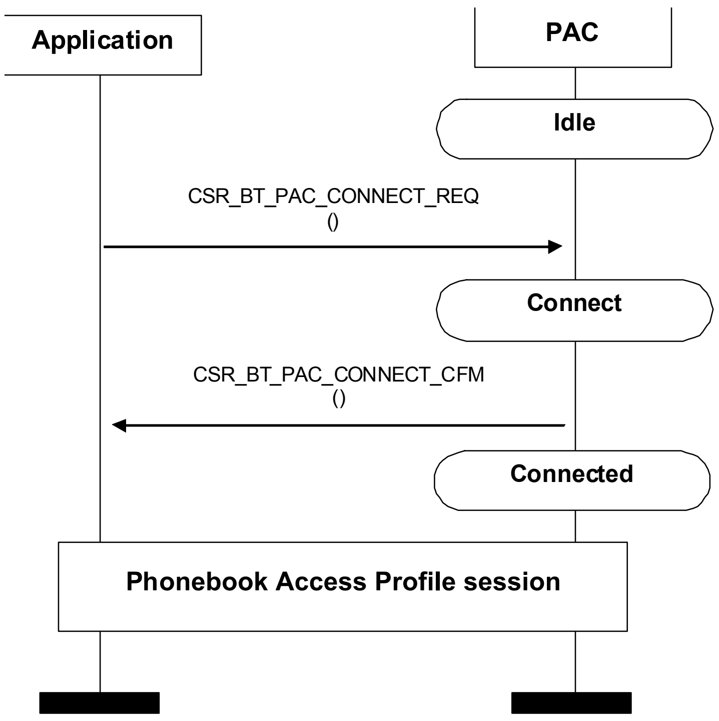


Figure 3: Connection handling

3.2 Browsing

Browsing an object store involves getting the folder contents and changing the ‘current folder’ forwards and backwards. The CSR_BT_PAC_SET_FOLDER_REQ is used for changing the current folder forward in the folder hierarchy. There are two ways to go back in the folder hierarchy – the first way is the CSR_BT_PAC_SET_ROOT_FOLDER_REQ changing the current folder to the root folder. The second way is the CSR_BT_PAC_SET_FOLDER_BACK_REQ changing the current folder back to the parent folder of the current one. To display a folder hierarchy starting with the root folder, the client must read the folder content using the message CSR_BT_PAC_PULL_VCARD_LIST_REQ. The folder listing object can be split over multiply packets using CSR_BT_PAC_PULL_VCARD_LIST_IND/RES messages. When the last part of the folder listing object has been received by the application then it will get a CSR_BT_PAC_PULL_VCARD_LIST_CFM message.

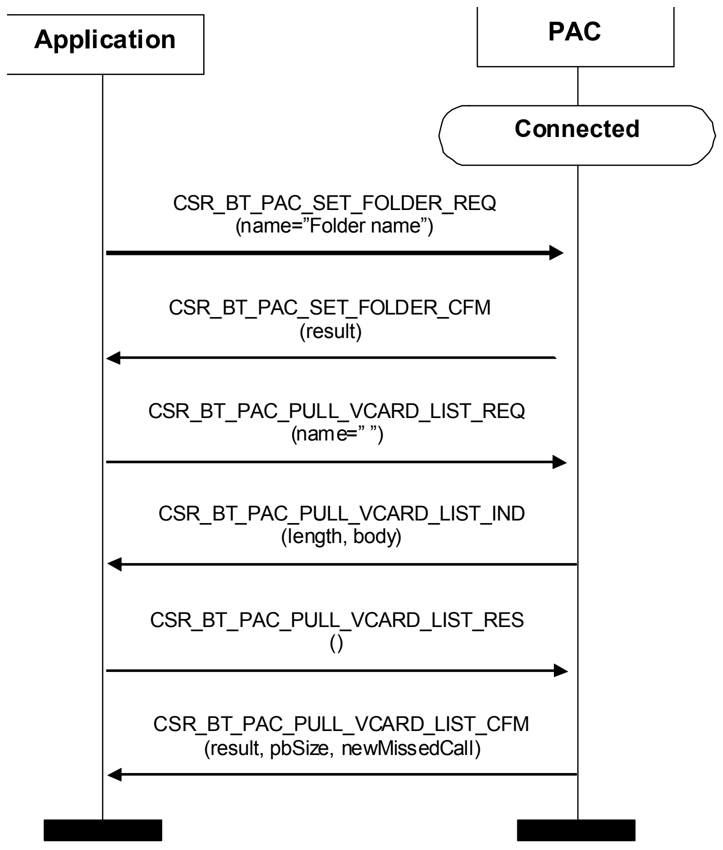

Application
PAC
Connected
CSR_BT_PAC_SET_FOLDER_REQ
(name="Folder name")
CSR_BT_PAC_SET_FOLDER_CFM
(result)
CSR_BT_PAC_PULL_VCARD_LIST_REQ
(name=" ")
CSR_BT_PAC_PULL_VCARD_LIST_IND
(length, body)
CSR_BT_PAC_PULL_VCARD_LIST_RES
()
CSR_BT_PAC_PULL_VCARD_LIST_CFM
(result, pbSize, newMissedCall)

Figure 4: Folder browsing handling

3.3 Object Transfer

An entire phone book (ph. ich, och, mcf or cch) can be pulled from the server by issuing a CSR_BT_PAC_PULL_PB_REQ. The server will respond with CSR_BT_PAC_PULL_VCARD_LIST_CFM or CSR_BT_PAC_PULL_VCARD_LIST_IND depending on whether the response has a body object. In case there is a body object the server will first respond with a CSR_BT_PAC_PULL_VCARD_LIST_IND containing the object (if the object is too big to fit in one packet then it is split in multiple packets). The application must respond with a CSR_BT_PAC_PULL_VCARD_LIST_RES (unless the application wants to terminate the operation. In this case it can use a CSR_BT_PAC_ABORT_REQ, see section 3.4) message to tell the PAC it is ready to receive the next packet. When the PAC has transferred the last part object it will end the operation by replying the next (and last) CSR_BT_PAC_PULL_VCARD_LIST_RES message with a CSR_BT_PAC_PULL_VCARD_LIST_CFM.

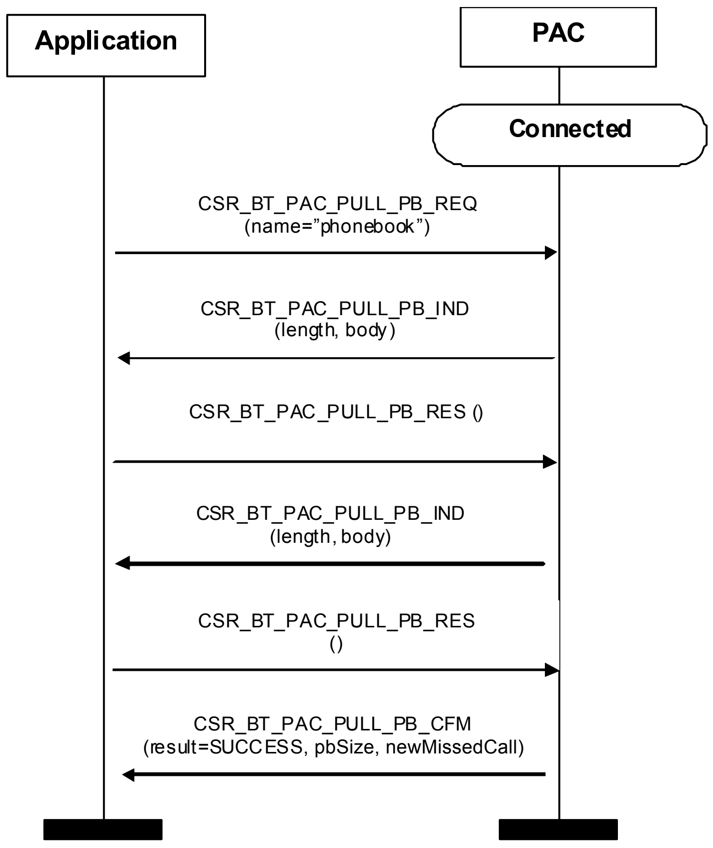

Application
PAC
Connected
CSR_BT_PAC_PULL_PB_REQ (name="phonebook")
CSR_BT_PAC_PULL_PB_IND (length, body)
CSR_BT_PAC_PULL_PB_RES ()
CSR_BT_PAC_PULL_PB_IND (length, body)
CSR_BT_PAC_PULL_PB_RES ()
CSR_BT_PAC_PULL_PB_CFM (result=SUCCESS, pbSize, newMissedCall)

Figure 5: Pull phone book from server

3.4 Abort Operation

The abort request (CSR_BT_PAC_ABORT_REQ) is used when the application decides to terminate a multi-packet operation (such as PULL_PB or PULL_VCARD_LIST) before it ends. The response (CSR_BT_PAC_ABORT_CFM) is received indicating that the abort request is a success. It is also indicating that the abort request is received and the PBAP server is now resynchronized with the client. If anything else is returned the PAC will disconnect the link and send CSR_BT_PAC_DISCONNECT_IND to the application.

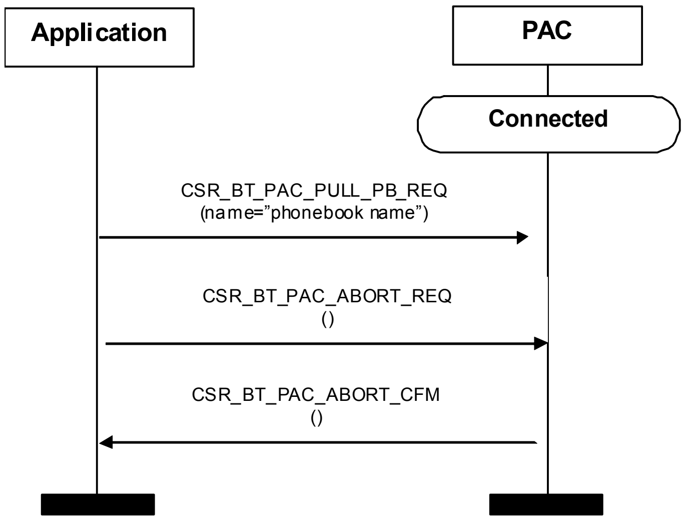


Figure 6: Abort operation

3.5 OBEX Authentication

The application can enable authentication of the PBAP server. This is done using the parameter's authorization and password in the CSR_BT_PAC_CONNECT_REQ message. The PAC will then authenticate the PBAP server under the OBEX connection establishment. An example of the authenticate sequence is illustrated below.

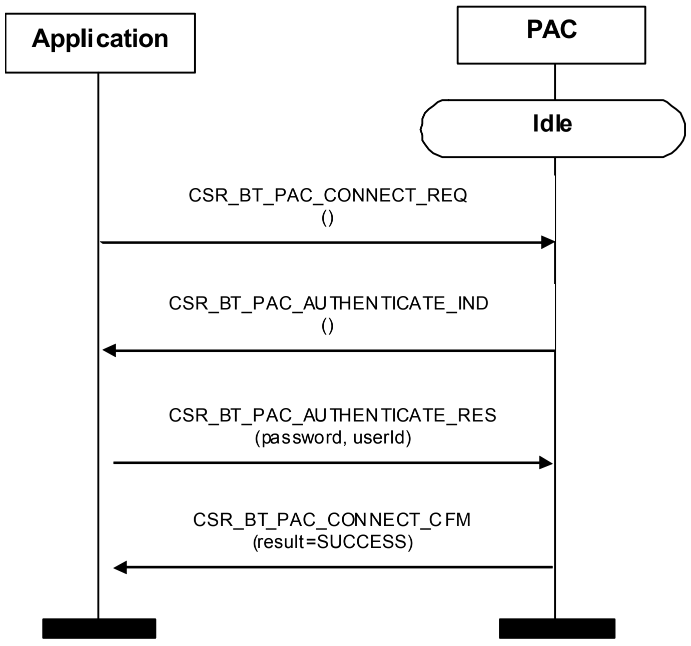


Figure 7: Authenticate connect

3.6 Disconnect

Sending a CSR_BT_PAC_DISCONNECT_REQ to the PAC disconnects the current connection (if any). Disconnecting may take some time and is confirmed with a CSR_BT_PAC_DISCONNECT_CFM signal.

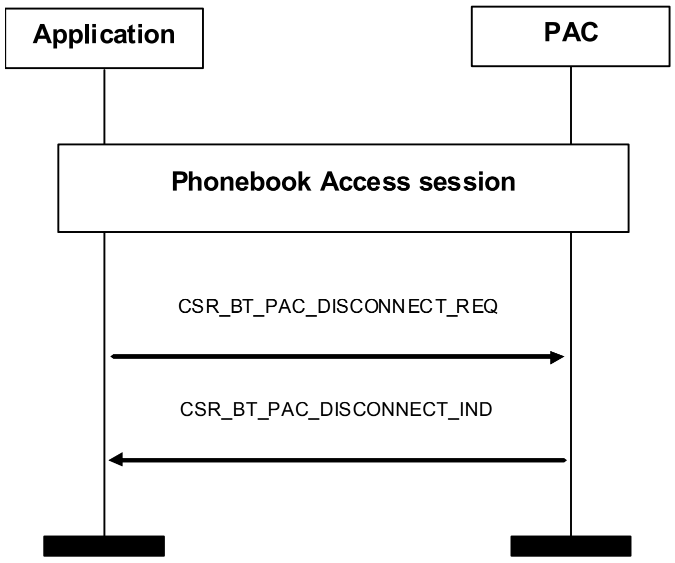


Figure 8: Normal disconnect

In case the peer side prematurely disconnects, the PAC sends a CSR_BT_PAC_DISCONNECT_IND to the application and enters IDLE state.

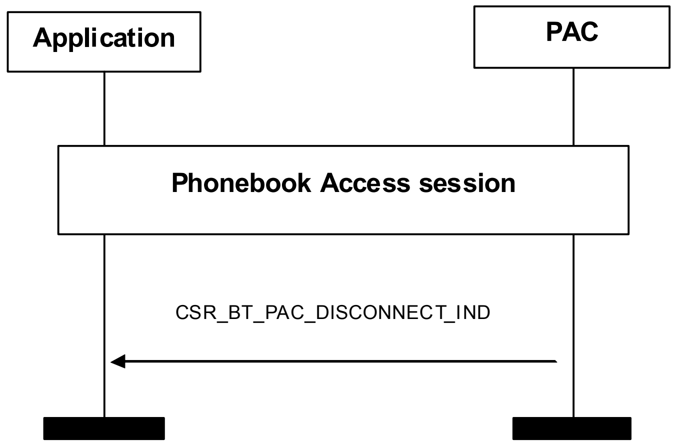


Figure 9: Abnormal disconnect

3.7 Payload Encapsulated Data

3.7.1 Using Offsets

As many OBEX messages contain multiple parameters with variable length, some of the parameters are based on *offsets* instead of standard pointers to the data. Signals with offset-based data can easily be recognized as they have both a *payload* and a *payloadLength* parameter. The *payload* contains the actual data, on which the offset is based. For example, a typical signal may contain the following:

```
CsrBtCommonPrim    type;
CsrUint8               result;
CsrUint16          ucs2nameOffset;
CsrUint16          bodyOffset;
CsrUint16          bodyLength;
CsrUint16          payloadLength;
CsrUint8               *payload;
```

In this example, two offset parameters can be found, namely *ucs2nameOffset* and *bodyOffset*. To obtain the actual data, the offset value is added to the *payload* pointer, which yields a pointer to the data, i.e.:

```
    CsrUint8  *ucs2name;
    ucs2name = (CsrUint8*)(primitive->payload + primitive->ucs2nameOffset);
```

As can be seen, the offset contains the number of bytes within the *payload* where the information begins. Similarly, the body data can be retrieved using the following:

```
CsrUint8 *body;
body = (CsrUint8*)(primitive->payload + primitive->bodyOffset);
```

And to illustrate the usage of the *length* parameter, which is also a common parameter, to copy the body one would typically use:

```
CsrMemCpy( copyOfBody, body, primitive->bodyLength );
```

Offset parameters will always have an “Offset” suffix on the name, and offsets are *always* relative to the “payload” parameter.

If the `bodyOffset` or the `bodyLength` is 0 (zero) this means that the signal does not contain any body. The same holds when the `payloadLength` is 0 (zero), which means that there is not payload.

3.7.2 Payload Memory

When the application receives a signal which has a *payload* parameter, the application must always free the payload pointer to avoid memory leaks, for example

```
pfree(primitive->payload);
pfree(primitive);
```

will free both the payload data and the message itself. Note that when the payload has been freed, offsets can not be used anymore, as the actual data is contained within the payload.

Signals that do not use the *payload* parameter must still have each of their pointer-based parameters freed.

4 OBEX Phone Book Access Profile Client Primitives

This section gives an overview of the primitives and parameters in the interface. Detailed information can be found in the corresponding csr_bt_pac_prim.h file.

4.1 List of All Primitives

Primitives:	Reference:
CSR_BT_PAC_CONNECT_REQ	See section 4.2
CSR_BT_PAC_CONNECT_CFM	See section 4.2
CSR_BT_PAC_AUTHENTICATE_IND	See section 4.3
CSR_BT_PAC_AUTHENTICATE_RES	See section 4.3
CSR_BT_PAC_PULL_PB_REQ	See section 4.4
CSR_BT_PAC_PULL_PB_CFM	See section 4.4
CSR_BT_PAC_PULL_PB_IND	See section 4.4
CSR_BT_PAC_PULL_PB_RES	See section 4.4
CSR_BT_PAC_GET_LIST_FOLDER_REQ	See section 4.4
CSR_BT_PAC_SET_FOLDER_REQ	See section 4.5
CSR_BT_PAC_SET_FOLDER_CFM	See section 4.5
CSR_BT_PAC_SET_BACK_FOLDER_REQ	See section 4.6
CSR_BT_PAC_SET_BACK_FOLDER_CFM	See section 4.6
CSR_BT_PAC_SET_ROOT_FOLDER_REQ	See section 4.7
CSR_BT_PAC_SET_ROOT_FOLDER_CFM	See section 4.7
CSR_BT_PAC_PULL_VCARD_LIST_REQ	See section 4.8
CSR_BT_PAC_PULL VCARD_LIST_CFM	See section 4.8
CSR_BT_PAC_PULL_VCARD_LIST_IND	See section 4.8
CSR_BT_PAC_PULL_VCARD_LIST_RES	See section 4.8
CSR_BT_PAC_PULL_VCARD_ENTRY_REQ	See section 4.9
CSR_BT_PAC_PULL_VCARD_ENTRY_CFM	See section 4.9
CSR_BT_PAC_PULL_VCARD_ENTRY_IND	See section 4.9
CSR_BT_PAC_PULL_VCARD_ENTRY_RES	See section 4.9
CSR_BT_PAC_ABORT_REQ	See section 4.10
CSR_BT_PAC_ABORT_CFM	See section 4.10
CSR_BT_PAC_DISCONNECT_REQ	See section 4.11
CSR_BT_PAC_DISCONNECT_IND	See section 4.11
CSR_BT_PAC_CANCEL_CONNECT_REQ	See Section 4.12
CSR_BT_PAC_SECURITY_OUT_REQ	See Section 4.13
CSR_BT_PAC_SECURITY_OUT_CFM	See Section 4.13

Table 1: List of all primitives

4.2 CSR_BT_PAC_CONNECT

Primitives \ Parameters	type	appHandle	maxPacketSize	destination	authorize	resultCode	resultSupplier	obexPeerMaxPacketSize	supportedRepositories	realmLength	*realm	passwordLength	*password	*userId	length	count	btConnId	windowSize	srmEnable
CSR_BT_PAC_CONNECT_REQ	✓	✓	✓	✓	✓					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CSR_BT_PAC_CONNECT_CFM	✓					✓	✓	✓	✓										

Table 2: CSR_BT_PAC_CONNECT Primitives

Description

To start an OBEX Phone Book Access session against the Phone Book Access server (PAS), the application sends a CSR_BT_PAC_CONNECT_REQ with the wanted max packet size allowed to send from the server. The server responds with a CSR_BT_PAC_CONNECT_CFM. If the connection is established the result is CSR_BT_OBEX_SUCCESS_RESULT_CODE. Any other value indicates a failure in the connection.

The connect messages between the OBEX Phone Book Access client and Server is guarded by a timer, thus if for some reason the server do not reply to the OBEX connect request within a fixed time interval the Bluetooth connection is disconnected direct. The timeout functionality is per default set to five seconds. The timeout value can be disable, or change by changing CSR_BT_OBEX_CONNECT_TIMEOUT, which is define in csr_bt_user_config.default.h. Note if the value of CSR_BT_OBEX_CONNECT_TIMEOUT is change, it will influence all OBEX profiles.

The function:

```
CsrBtPacConnectExtReqSend (CsSchedrQid      appHandle,
                          CsrUint16         maxPacketSize,
                          CsrBtDeviceAddr   destination,
                          CsrBool           authorize,,
                          CsrUint16         realmLength,
                          data_ptr_t        *realm,
                          CsrUint16         passwordLength,
                          CsrUint8          *password,
                          CsrCharString     *userId,
                          CsrUint32         length,
                          CsrUint32         count);
```

defined in csr_bt_pac_lib.h, builds and sends the CSR_BT_PAC_CONNECT_REQ primitive to the PAC profile.

Parameters

type	Signal identity, CSR_BT_PAC_CONNECT_REQ/CFM.
appHandle	The identity of the calling process. It is possible to initiate the procedure by any higher layer process as the response is returned to appHandle.
maxPacketSize	The maximum OBEX packet size allowed sending to the client application.
destination	The Bluetooth® address of the device to connect to.

authorize	Is to control the OBEX authentication on connection to a PAS. If TRUE the PAC will initiate the authentication against the server.
resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. If the resultSupplier == CSR_BT_SUPPLIER_OBEX then the possible result codes can be found in csr_bt_obex.h. All values which are currently not specified in the respective prim.h files or csr_bt_obex.h are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h
obexPeerMaxPacketSize	Indicates the maximum size OBEX packet that is allowed to send to the server
supportedRepositories	Indicates the supported repositories in the server. The possible values are internal memory based phonebook (CSR_BT_PAC_SRC_PHONE) or the SIM (CSR_BT_PAC_SRC_SIM).
realmLength	Number of bytes in realm of type CsrUint16
*realm	A displayable string indicating for the user which userid and/or password to use. The first byte of the string is the character set of the string. The table below shows the different values for character set. Note that this pointer ca be set to NULL if the realm is not provided to the server.

Char set Code	Meaning
0	ASCII
1	ISO-8859-1
2	ISO-8859-2
3	ISO-8859-3
4	ISO-8859-4
5	ISO-8859-5
6	ISO-8859-6
7	ISO-8859-7
8	ISO-8859-8
9	ISO-8859-9
0xFF = 255	UNICODE

passwordLength	The length of the response password.
*password	Containing the challenge password of the OBEX authentication. This is a pointer which shall be allocated by the application.
*userId	Zero terminated string (ASCII) containing the userId for the authentication. This is a pointer which shall be allocated by the application.
length	Length is use to express the approximate total length of the bodies of all the objects in the transaction. If set to 0 this header will not be include.

count	Count is use to indicate the number of objects that will be sent during this connection. If set to 0 this header will not be include.
btConnId	Identifier used when moving the connection to another AMP controller, i.e. when calling the `CsrBtAmpmMoveReqSend`-function.
windowSize	Controls how many packets the OBEX profile (and lower protocol layers) are allowed to cache on the data receive side. A value of zero (0) will cause the system to auto-detect this value.
srmEnable	Enable local support for Single Response Mode.

4.3 CSR_BT_PAC_AUTHENTICATE

Primitives \ Parameters	type	options	realmLength	* realm	deviceAddr	*password	passwordLength	*userId
CSR_BT_PAC_AUTHENTICATE_IND	✓	✓	✓	✓	✓			
CSR_BT_PAC_AUTHENTICATE_RES	✓					✓	✓	✓

Table 3: CSR_BT_PAC_AUTHENTICATE Primitives

Description

The indication and response signal is used when the PAS wants to OBEX authenticate the PAC. The application has to response with the password or pin number in the password and userId for the server to identify the proper password.

Parameters

type	Signal identity, CSR_BT_PAC_AUTHENTICATE_IND/RES.
options	Challenge information of type CsrUint8. Bit 0 controls the responding of a valid user Id. If bit 0 is set it means that the application must response with a user Id in a CSR_BT_PAC_AUTHENTICATE_RES message. If bit 0 is not set the application can just set the userId to NULL. Bit 1 indicates the access mode being offered by the sender If bit 1 is set the access mode is read only. If bit 1 is not set the sender gives full access, e.g. both read and write. Bit 2 - 7 is reserved.
realmLength	Number of bytes in realm of type CsrUint16
* realm	A displayable string indicating for the user which userid and/or password to use. The first byte of the string is the character set of the string. The table below shows the different values for character set. Note that this pointer must be pfree by the application, and that this pointer can be NULL because the realm field is optional to set by the peer device.

Char set Code	Meaning
0	ASCII
1	ISO-8859-1
2	ISO-8859-2
3	ISO-8859-3
4	ISO-8859-4

5	ISO-8859-5
6	ISO-8859-6
7	ISO-8859-7
8	ISO-8859-8
9	ISO-8859-9
0xFF = 255	UNICODE

deviceAddr	The Bluetooth address of the device that has initiated the OBEX authentication procedure
*password	Containing the response password of the OBEX authentication. This is a pointer which shall be allocated by the application.
passwordLength	The length of the response password.
*userId	Zero terminated string (ASCII) containing the userId for the authentication. This is a pointer which shall be allocated by the application.

4.4 CSR_BT_PAC_PULL_PB

Parameters / Primitives	type	*ucs2name	src	filter[8]	format	maxListCnt	listStartOffset	responseCode	pbSize	newMissedCall	bodyLength	bodyOffset	payloadLength	*payload	smpOn
CSR_BT_PAC_PULL_PB_REQ	✓	✓	✓	✓	✓	✓	✓								✓
CSR_BT_PAC_PULL_PB_CFM	✓							✓	✓	✓					
CSR_BT_PAC_PULL_PB_IND	✓										✓	✓	✓	✓	
CSR_BT_PAC_PULL_PB_RES	✓														✓

Table 4: CSR_BT_PAC_PULL_PB Primitives

Description

To retrieve a Phone book from the PAS the apps sends the CSR_BT_PAC_PULL_PB_REQ to the PAC and the name parameter specifies which phone book the PAC client wants. The server will as response sent a CSR_BT_PAC_PULL_PB_IND signal which contains the first part of phone book. The client must response with a CSR_BT_PAC_PULL_PB_RES to indicate that it is ready receive the next part of the phone book. The Client and server will continue the CSR_BT_PAC_PULL_PB_RES/IND session until the last packet have been transferred them will the server end the operation with a CSR_BT_PAC_PULL_PB_CFM signal.

In case the result is different from success, the other parameters are invalid and not used, then the result is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE, the release pointer must be pfreed in the application, and NOT the body pointer.

Parameters

type	Signal identity, CSR_BT_PAC_PULL_PB_REQ/CFM/IND/RES.
*ucs2name	A null terminated 16 bit Unicode big-endian text string (UCS-2BE) containing the (file) name of the object. The function “CsrUtf82Ucs2String” can be used for converting a null terminated UTF8 text string into a null terminated UCS2 text string.
src	Indicate whether phone book source should be the internal memory (CSR_BT_PAC_SRC_PHONE) or the SIM (CSR_BT_PAC_SRC_SIM).
filter[8]	Indicate the attribute contained in the requested vCard objects. If the PBAP server does not support this attribute it will be ignored. Only the attributes N and TEL are mandatory to be supported by the PBAP server, see the detail about the filter parameter in [PBAP]. The filter is a 64 bit long flag store in an array of 8 CsrUint8. The lower part of the flag; filter[0] contains bit 0 – 7, filter[1] contains bit 8 – 15, filter[2] contains bit 16 – 23, filter[3] contains bit 24 – 31, filter[4] contains bit 32 – 39, filter[5] contains bit 40 – 47, filter[6] contains bit 48 – 55 and filter[7] contains bit 56 – 63.
format	Indicate the requested format vCard 2.1 (CSR_BT_PAC_FORMAT_VCARD2_1) or vCard 3.0 (CSR_BT_PAC_FORMAT_VCARD3:0)
maxListCnt	Used for indicating the maximum number of entries of the phone book the application

	can receive. When the application set maxListCnt to zero it signifies that it wants to know the actual number of indexes used in the phone book of interest (i.e. indexes that correspond to non-null entries). When maxListCnt is zero the PBAP server ignores all other application parameters that may be present in the request. T0he response will not contain any body (phone book object).
listStartOffset	Used for indicating the offset of the first entry of the vCard-listing.
responseCode	The valid response codes are defined (in csr_bt_obex.h). For success the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the operation.
pbSize	The numbers of entries in the phone book of interest. This parameter is only used when maxListCnt is zero in the request, see maxListCnt.
newMissedCall	This parameter is only used when the phone book of interest is mch (mch.vcf). It indicates the numbers of missed calls that had not been checked yet since the PBAP session was started.
bodyLength	The length of the body (object).
bodyOffset	Payload relative offset for the object itself.
payloadLength	Number of bytes in payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
smpOn	Reserved for future use. Set to FALSE.

4.5 CSR_BT_PAC_SET_FOLDER

Primitives \ Parameters	type	* ucs2name	responseCode
CSR_BT_PAC_SET_FOLDER_ REQ	✓	✓	
CSR_BT_PAC_SET_FOLDER_CFM	✓		✓

Table 5: CSR_BT_PAC_SET_FOLDER Primitives

Description

This signal is used for changing the current folder on the server to a folder specified with the name parameter. This is used for navigating down in the directory hierarchy on the server. The result of the change folder operation is given in the confirm signal. The result can contain error codes corresponding to the reason for failure in case the folder does not exist or in case the server does not permit this operation.

Parameters

type	Signal identity, CSR_BT_PAC_SET_FOLDER_REQ/CFM.
* ucs2name	A null terminated 16 bit Unicode big-endian text string (UCS-2BE) containing the (file) name of the object. The function "CsrUtf82Ucs2String" can be used for converting a null terminated UTF8 text string into a null terminated UCS2 text string.
responseCode	The valid response codes are defined (in csr_bt_obex.h). For success in the confirm signal the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the confirm signal.

4.6 CSR_BT_PAC_SET_BACK_FOLDER

Parameters / Primitives	type	responseCode
CSR_BT_PAC_SET_BACK_FOLDER_REQ	✓	
CSR_BT_PAC_SET_BACK_FOLDER_CFM	✓	✓

Table 6: CSR_BT_PAC_SET_BACK_FOLDER Primitives

Description

This signal is used for setting the current folder back to the parent folder. The result of the operation is given in the confirm signal. If the current folder is the root folder the confirm signal will have the CSR_BT_OBEX_NOT_FOUND_RESPONSE_CODE result code.

Parameters

type	Signal identity, CSR_BT_PAC_SET_BACK_FOLDER_REQ/CFM.
responseCode	The valid result codes are defined (in csr_bt_obex.h). For success in the confirm signal the result is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the confirm signal.

4.7 CSR_BT_PAC_SET_ROOT_FOLDER

Primitives \ Parameters	type	responseCode
CSR_BT_PAC_SET_ROOT_FOLDER_REQ	✓	
CSR_BT_PAC_SET_ROOT_FOLDER_CFM	✓	✓

Table 7: CSR_BT_PAC_SET_ROOT_FOLDER Primitives

Description

This signal is used for setting the current folder back to the root folder. The result in the confirm message can contain error codes corresponding to the reason for the failure on the server, but normally this operation cannot fail.

Parameters

type	Signal identity, CSR_BT_PAC_SET_ROOT_FOLDER_REQ/CFM.
responseCode	The valid response codes are defined (in csr_bt_obex.h). For success in the confirm signal the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the confirm signal.

4.8 CSR_BT_PAC_PULL_VCARD_LIST

Parameters / Primitives	type	* ucs2name	order	*searchVal	searchVallen	searchAtt	maxListCnt	listStartOffset	responseCode	pbSize	newMissedCall	bodyLength	bodyOffset	payloadLength	*payload	smpOn
CSR_BT_PAC_PULL_VCARD_LIST_REQ	✓	✓	✓	✓	✓	✓	✓	✓								✓
CSR_BT_PAC_PULL_VCARD_LIST_CFM	✓								✓	✓	✓					
CSR_BT_PAC_PULL_VCARD_LIST_IND	✓											✓	✓	✓	✓	
CSR_BT_PAC_PULL_VCARD_LIST_RES	✓															✓

Table 8: CSR_BT_PAC_PULL_VCARD_LIST Primitives

Description

This signal is used for pulling the content of the current folder or a folder specified by the name parameter. The folder structure must confirm to the virtual folder structure given in [PBAP]. The content of the folder must be sent in the Folder Listing format specified in ref. [PBAP]. The actual vCard-listing object can be split in multiply CSR_BT_PAC_PULL_VCARD_LIST_IND body packets.

The PBAP server will respond with a CSR_BT_PAC_PULL_VCARD_IND unless an error appear or maxListCnt = 0, in this case it will respond with a CSR_BT_PAC_PULL_VCARD_LIST_CFM and the parameter result will indicate the reasons why. maxListCnt signifies that the application want to know the numbers of indexes in the phone book of interest. The application must respond the CSR_BT_PAC_PULL_VCARD_LIST_IND with a CSR_BT_PAC_PULL_VCARD_LIST_RES indicating that it is ready to receive another packet.

The last signal in the operation is always the CSR_BT_PAC_PULL_VCARD_LIST_CFM signal.

The release pointer must be pfreed'ed in the application, and NOT the body pointer.

Parameters

type	Signal identity, CSR_BT_PAC_PULL_VCARD_LIST_REQ/CFM/IND/RES.
* ucs2name	A null terminated 16 bit Unicode big-endian text string (UCS-2BE) containing the (file) name of the object. The function "CsrUtf82Ucs2String" can be used for converting a null terminated UTF8 text string into a null terminated UCS2 text string.
order	Used for indicating which sorting order shall be used for the vCard-listing. The order is always ascendant. There are three possible sorting orders: Alphabetical (CSR_BT_PAC_ORDER_ALPHABETICAL) indexed (CSR_BT_PAC_SEARCH_ATT_NUMBER) or phonetic (CSR_BT_PAC_ORDER_PHONETICAL).
searchVal	Null terminated UTF-8 big-endian text string. The parameter indicates which vCards shall be contained in the vCard-listing; see detail in ref. [PBAP].
searchVallen	Length of the searchVal including zero termination.

searchAtt	Indicate which attributes should be used when searching. There are three possible attributes: Name (CSR_BT_PAC_SEARCH_ATT_NAME), Number (CSR_BT_PAC_SEARCH_ATT_NUMBER) or Sound (CSR_BT_PAC_SEARCH_ATT_SOUND).
maxListCnt	This parameter indicates the maximum number of entries of the vCard-listing. When the application sets maxListCnt to zero it signifies that it wants to know the actual used number of indexes in the phone book of interest (i.e. indexes that correspond to non-null entries). When maxListCnt is zero the PBAP server ignores all other application parameters that may be present in the request. The response will not contain any body (phone book object).
listStartOffset	Used for indicating the offset of the first entry of the vCard-listing.
responseCode	The valid response codes are defined (in csr_bt_obex.h). For success in the confirm signal the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the confirm signal.
pbSize	The numbers of entries in the phone book of interest see maxListCnt. This parameter is only used when maxListCnt is zero in the request, see maxListCnt.
newMissedCall	This parameter is used when the phone book of interest is mch. It indicates the numbers of missed calls that had not been checked yet since the PBAP session was started.
bodyLength	The length of the body (object).
bodyOffset	Offset relative to payload of the object itself.
payloadLength	Number of bytes in payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
smpOn	Reserved for future use. Set to FALSE.

4.9 CSR_BT_PAC_PULL_VCARD_ENTRY

Primitives \ Parameters	type	* ucs2name	filter[8]	format	responseCode	bodyLength	bodyOffset	payloadLength	*payload	smpOn
CSR_BT_PAC_PULL_VCARD_ENTRY_REQ	✓	✓	✓	✓						✓
CSR_BT_PAC_PULL_VCARD_ENTRY_CFM	✓				✓					
CSR_BT_PAC_PULL_VCARD_ENTRY_IND	✓					✓	✓	✓	✓	
CSR_BT_PAC_PULL_VCARD_ENTRY_RES	✓									✓

Table 9: CSR_BT_PAC_PULL_VCARD_ENTRY Primitives

Description

To retrieve a single phone book entry from the PBAP server the application sends the CSR_BT_PAC_PULL_VCARD_ENTRY_REQ and the parameter name specifies which entry the application wants. The PBAP server shall response with a CSR_BT_PAC_PULL_VCARD_IND which contains the first part of the entry. The client and server will continue sent and receive CSR_BT_PAC_PULL_VCARD_RES/IND until the entire entry has been transferred to the application. Then the PBAP server sends a CSR_BT_PAC_PULL_VCARD_ENTRY_CFM to confirm the end of the operation.

In case the result is different from success, the other parameters are invalid and not used, then the result is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE, the release pointer must be pfreed in the application, and NOT the body pointer.

Parameters

type	Signal identity, CSR_BT_PAC_PULL_VCARD_ENTRY_REQ/CFM/IND/RES.
*ucs2Name	A null terminated 16 bit Unicode big-endian text string (UCS-2BE) containing the (file) name of the object. The function “CsrUtf82Ucs2String” can be used for converting a null terminated UTF8 text string into a null terminated UCS2 text string.
filter[8]	Indicate the attribute contained in the requested vCard objects. The attributes not supported by the PBAP server will be ignored. Only the attributes N and TEL are mandatory to be supported by the PBAP server, see the detail about the filter parameter in [PBAP]. The filter is a 64 bit long flag store in an array of 8 CsrUint8. The lower part of the flag; filter[0] contains bit 0 – 7, filter[1] contains bit 8 – 15, filter[2] contains bit 16 – 23, filter[3] contains bit 24 – 31, filter[4] contains bit 32 – 39, filter[5] contains bit 40 – 47, filter[6] contains bit 48 – 55 and filter[7] contains bit 56 – 63.
format	The format parameter indicates the requested format vCard 2.1 (CSR_BT_PAC_FORMAT_VCARD2_1) or vCard 3.0 (CSR_BT_PAC_FORMAT_VCARD3:0)
responseCode	The valid response codes are defined (in csr_bt_obex.h). For success in the confirm signal the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure in the confirm signal.

bodyLength	The length of the body (object).
bodyOffset	Offset relative to payload of the object itself.
payloadLength	Number of bytes in payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
smpOn	Reserved for future use. Set to FALSE.

4.10 CSR_BT_PAC_ABORT

Primitives \ Parameters	type
CSR_BT_PAC_ABORT_REQ	✓
CSR_BT_PAC_ABORT_CFM	✓

Table 10: CSR_BT_PAC_ABORT Primitives

Description

The CSR_BT_PAC_ABORT_REQ is used when the apps decides to terminate a multi-packet operation (such as GET/PUT) before it normally ends. The CSR_BT_PAC_ABORT_CFM indicates that the server has received the abort response and the server is now resynchronized with the client. If the server does not respond the Abort Request or it response with a response code different from CSR_BT_OBEX_SUCCESS_RESPONSE_CODE, the profile will disconnect the Bluetooth connection and send a CSR_BT_DISCONNECT_IND to the application.

Parameters

type	Signal identity, CSR_BT_PAC_ABORT_REQ/CFM.

4.11 CSR_BT_PAC_DISCONNECT

Primitives \ Parameters	type	normalDisconnect	reasonCode	reasonSupplier
CSR_BT_PAC_DISCONNECT_REQ	✓	✓		
CSR_BT_PAC_DISCONNECT_IND	✓		✓	✓

Table 11: CSR_BT_PAC_DISCONNECT Primitives

Description

To disconnect a connection to a server (if any) send a CSR_BT_PAC_DISCONNECT_REQ to the PAC. When disconnected, the PAC will respond with a CSR_BT_PAC_DISCONNECT_IND. If the link is dropped in the middle of a session, the apps will receive a CSR_BT_PAC_DISCONNECT_IND indicating that the OBEX file transfer session is finished, and is ready for a new one.

The disconnect messages between the OBEX Phone Book Access client and Server is guarded by a timer, thus if for some reason the server do not reply to the OBEX disconnect request within a fixed time interval the Bluetooth connection is disconnected direct. The timeout functionality is per default set to five seconds. The timeout value can be disable, or change by changing CSR_BT_OBEX_DISCONNECT_TIMEOUT, which is define in csr_bt_user_config.default.h. Note if the value of CSR_BT_OBEX_DISCONNECT_TIMEOUT is change, it will influence all OBEX profiles.

Parameters

type	Signal identity, CSR_BT_PAC_DISCONNECT_REQ/IND.
normalDisconnect	FALSE defines an Abnormal disconnect sequence where the Bluetooth connection is release direct. TRUE defines a normal disconnect sequence where the OBEX connection is release before the Bluetooth connection.
reasonCode	The reason code of the operation. Possible values depend on the value of reasonSupplier. If e.g. the reasonSupplier == CSR_BT_SUPPLIER_CM then the possible reason codes can be found in csr_bt_cm_prim.h. If the reasonSupplier == CSR_BT_SUPPLIER_OBEX then the possible result codes can be found in csr_bt_obex.h. All values whichare currently not specified in the respective prim.h files or csr_bt_obex.h are regarded as reserved and the application should consider them as errors.
reasonSupplier	This parameter specifies the supplier of the reason given in reasonCode. Possible values can be found in csr_bt_result.h

4.12 CSR_BT_PAC_CANCEL_CONNECT

Primitives \ Parameters	type
CSR_BT_PAC_CANCEL_CONNECT_REQ	✓

Table 12: CSR_BT_PAC_CANCEL_CONNECT Primitives

Description

The CSR_BT_PAC_CANCEL_CONNECT_REQ can be used the cancel an ongoing connect procedure. If the PAC succeeds to cancel the ongoing connection attempt the application will receive a CSR_BT_PAC_CONNECT_CFM with a response code different from CSR_BT_OBEX_SUCCESS_RESPONSE_CODE.

Parameters

type	Signal identity, CSR_BT_PAC_CANCEL_CONNECT_REQ

4.13 CSR_BT_PAC_SECURITY_OUT

Parameters / Primitives	type	appHandle	secLevel	resultCode	resultSupplier
CSR_BT_PAC_SECURITY_OUT_REQ	✓	✓	✓		
CSR_BT_PAC_SECURITY_OUT_CFM	✓			✓	✓

Table 13: CSR_BT_PAC_SECURITY_OUT Primitives

Description

Applications that wish to change the enforcement to a specific profile security level, i.e. authentication, encryption and/or authorisation, can use this API to set up the security level for *new* connections. Note that this API is for the local device only and can be used from within any state.

The *CSR_BT_SECURITY_OUT_REQ* signal sets up the security level for new outgoing connections. Already established and pending connections are not altered. Note that *authorisation* should not be used for outgoing connections as that may be confusing for the user – there is really no point in requesting an outgoing connection and afterwards having to authorise as they are both locally-only decided procedures.

Note, that any attempts to set security to a less secure level than the mandatory security level will be rejected. See csr_bt_profiles.h for mandatory security settings. The default settings used by CSR Synergy Bluetooth are set to require authentication and encryption.

Note that if MITM protection is requested and the remote device does not have the required IO capabilities, pairing/bonding will fail and connections to the remote device *cannot* be made. See [SC] for further details.

Parameters

type | Signal identity CSR_BT_PAC_SECURITY_OUT_REQ/CFM.

appHandle | Application handle to which the confirm message is sent.

secLevel | The application must specify one of the following values:

- CSR_BT_SEC_DEFAULT : Use default security settings
- CSR_BT_SEC_MANDATORY : Use mandatory security settings
- CSR_BT_SEC_SPECIFY : Specify new security settings

If CSR_BT_SEC_SPECIFY is set the following values can be OR'ed additionally:

- CSR_BT_SEC_AUTHORISATION: Require authorisation
- CSR_BT_SEC_AUTHENTICATION: Require authentication
- CSR_BT_SEC_ SEC_ENCRYPTION: Require encryption (implies authentication)
- CSR_BT_SEC_MITM: Require MITM protection (implies encryption)

resultCode | The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the

	possible result codes can be found in csr_bt_cm_prim.h. If the resultSupplier == CSR_BT_SUPPLIER_OBEX then the possible result codes can be found in csr_bt_obex.h. All values which are currently not specified in the respective prim.h files or csr_bt_obex.h are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

5 Document References

Document	Reference
OBJECT PUSH PROFILE Version 1.1 Profile section K:11 22 February 2001	OPP
Phone Book Access Profile Version 1.0 27 April 2006	PBAP
IrDA Object Exchange Protocol - IrOBEX Version 1.3 3 January 2003	OBEX
Specifications for Ir Mobile Communications (IrMC) Version 1.1 01 March 1999	IRMC
CSR Synergy Bluetooth, SC – Security Controller API Description, Document no. api-0102-sc	SC

Terms and Definitions

BlueCore®	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth®	Set of technologies providing audio and data transfer over short-range radio connections
CSR	Cambridge Silicon Radio
PAS	OBEX Phone Book Access Profile Server
PAC	OBEX Phone Book Access Profile Client
OPS	OBEX Push Server
OPC	OBEX Push Client
SDS	Service Discovery Server
pb	Main Phone book
ich	Incoming calls History
och	Outgoing Calls History
mch	Missed Calls History
cch	Combined Call History
SIG	Special Interest Group
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0

TradeMarks, Patents and Licences

Life Support Policy and Use in Safety-critical Compliance

Performance and Conformance

Refer to www.csrsupport.com for compliance and conformance to standards information.