FOREWORD

This document, together with the Generic Object Exchange profile and the Generic Access profile form the File Transfer usage model.

Interoperability between devices from different manufacturers is provided for a specific service and usage model if the devices conform to a Bluetooth SIG-defined profile specification. A profile defines a selection of messages and procedures (generally termed *capabilities*) from the Bluetooth SIG specifications, and gives an unambiguous description of the air interface for specified service(s) and usage model(s).

All defined features are process-mandatory. This means that if a feature is used, it is used in a specified manner. Whether the provision of a feature is mandatory or optional is stated separately for both sides of the Bluetooth air interface.

1 INTRODUCTION

1.1 SCOPE

The File Transfer profile defines the requirements for the protocols and procedures that shall be used by the applications providing the File Transfer usage model. This profile uses the Generic Object Exchange profile (GOEP) as a base profile to define the interoperability requirements for the protocols needed by the applications. The most common devices using these usage models can be (but are not limited to) PCs, notebooks, and PDAs.

The scenarios covered by this profile are the following:

- Usage of a Bluetooth device (e.g. a notebook PC) to browse an object store (file system) of another Bluetooth device. Browsing involves viewing objects (files and folders) and navigating the folder hierarchy of another Bluetooth device. For example, one PC browsing the file system of another PC.
- A second usage is to transfer objects (files and folders) between two Bluetooth devices. For example, copying files from one PC to another PC.
- A third usage is for a Bluetooth device to manipulate objects (files and folders) on another Bluetooth device. This includes deleting objects, and creating new folders.

1.2 BLUETOOTH PROFILE STRUCTURE

In Figure 1.1, the Bluetooth profile structure and the dependencies of the profiles are depicted. A profile is dependent upon another profile if it re-uses parts of that profile, by implicitly or explicitly referencing it. Dependency is illustrated in the figure: a profile has dependencies on the profile(s) in which it is contained – directly and indirectly.

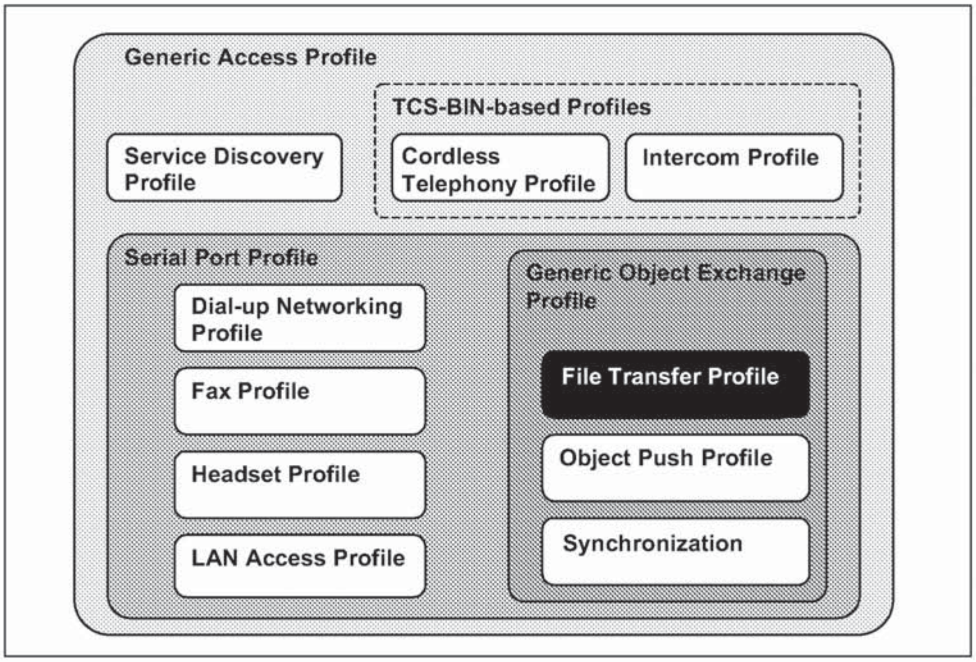

Figure 1.1: Bluetooth Profiles

1.3 BLUETOOTH OBEX-RELATED SPECIFICATIONS

Bluetooth Specification includes five separate specifications for OBEX and applications using OBEX.

1. Bluetooth IrDA Interoperability Specification [1].

- Defines how the applications can function over both Bluetooth and IrDA.
- Specifies how OBEX is mapped over RFCOMM and TCP.
- Defines the application profiles using OBEX over Bluetooth.

2. Bluetooth Generic Object Exchange Profile Specification [2]

- Generic interoperability specification for the application profiles using OBEX.
- Defines the interoperability requirements of the lower protocol layers (e.g. Baseband and LMP) for the application profiles.

3. Bluetooth Synchronization Profile Specification [3]

- Application Profile for Synchronization applications.
- Defines the interoperability requirements for the applications within the Synchronization application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

4. Bluetooth File Transfer Profile Specification (This Specification)

- Application Profile for File Transfer applications.
- Defines the interoperability requirements for the applications within the File Transfer application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

5. Bluetooth Object Push Profile Specification [4]

- Application Profile for Object Push applications.
- Defines the interoperability requirements for the applications within the Object Push application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

1.4 SYMBOLS AND CONVENTIONS

1.4.1 Requirement status symbols

In this document (especially in the profile requirements tables in Annex A), the following symbols are used:

'M' for mandatory to support (used for capabilities that shall be used in the profile);

'O' for optional to support (used for capabilities that can be used in the profile);

'C' for conditional support (used for capabilities that shall be used in case a certain other capability is supported);

'X' for excluded (used for capabilities that may be supported by the unit but shall never be used in the profile);

'N/A' for not applicable (in the given context it is impossible to use this capability).

Some excluded capabilities are capabilities that, according to the relevant Bluetooth specification, are mandatory. These are features that may degrade operation of devices following this profile. Therefore, these features shall never be activated while a unit is operating as a unit within this profile.

1.4.2 Signaling diagram conventions

The following arrows are used in diagrams describing procedures:

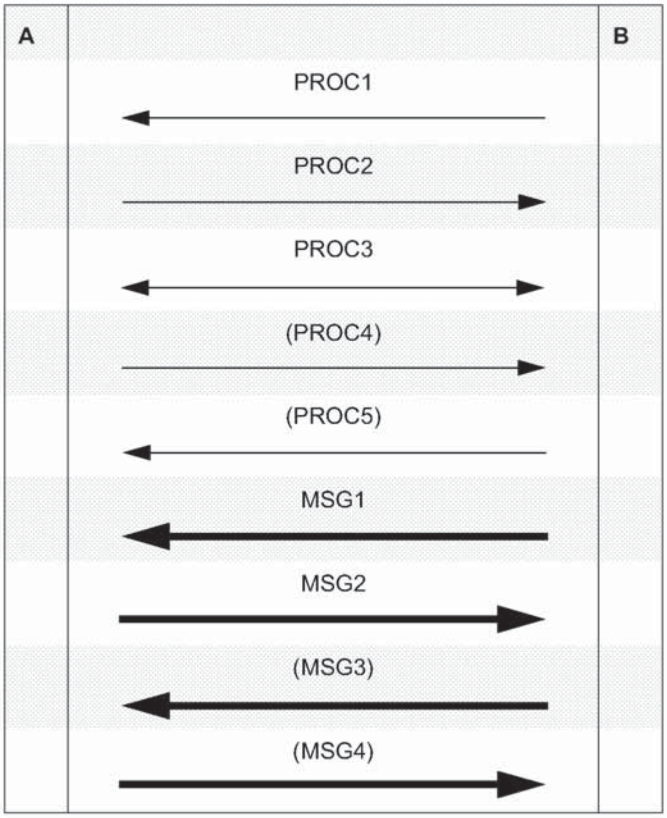

Table 1.1: Arrows used in signaling diagrams

In the table above, the following cases are shown: PROC1 is a sub-procedure initiated by B. PROC2 is a sub-procedure initiated by A. PROC3 is a sub-procedure where the initiating side is undefined (may be both A and B). PROC4 indicates an optional sub-procedure initiated by A, and PROC5 indicates an optional sub-procedure initiated by B.

MSG1 is a message sent from B to A. MSG2 is a message sent from A to B. MSG3 indicates an optional message from A to B, and MSG4 indicates an optional message from B to A.

2 PROFILE OVERVIEW

2.1 PROFILE STACK

The figure below shows the protocols and entities used in this profile.

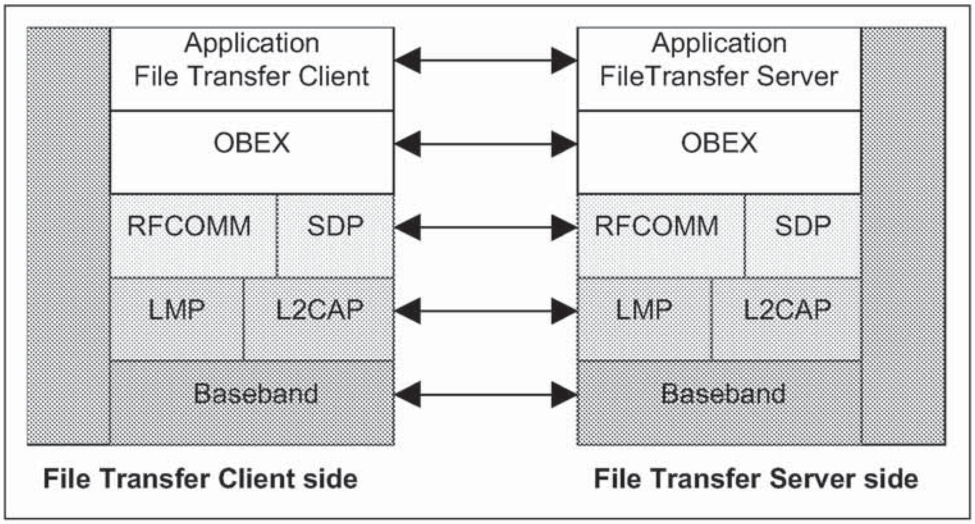

Figure 2.1: Protocol model

The Baseband [5], LMP [6] and L2CAP [7] are the OSI layer 1 and 2 Bluetooth protocols. RFCOMM [8] is the Bluetooth adaptation of GSM TS 07.10 [9]. SDP is the Bluetooth Service Discovery Protocol [10]. OBEX [1] is the Bluetooth adaptation of IrOBEX [11].

The RFCOMM, L2CAP, LMP, and Baseband interoperability requirements are defined in GOEP.

2.2 CONFIGURATIONS AND ROLES

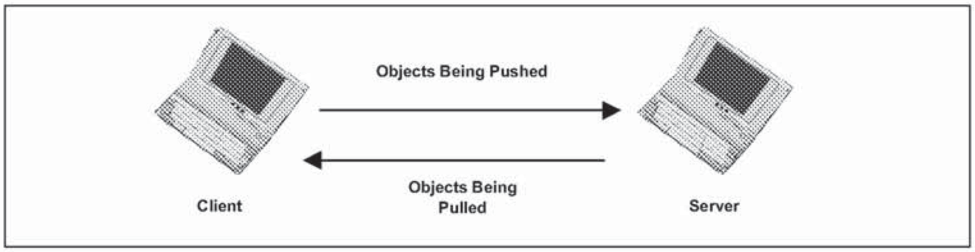

Figure 2.2: Bi-directional File Transfer Example between two Personal Computers

The following roles are defined for this profile:

Client – The Client device initiates the operation, which pushes and pulls objects to and from the *Server*. In addition to the interoperability requirements defined in this profile, the Client must also comply with the interoperability requirements for the Client of the GOEP if not defined in the contrary. The Client must be able to interpret the OBEX Folder Listing format and may display this information for the user.

Server – The Server device is the target remote Bluetooth device that provides an object exchange server and folder browsing capability using the OBEX Folder Listing format. In addition to the interoperability requirements defined in this profile, the Server must comply with the interoperability requirements for the Server of the GOEP if not defined in the contrary.

2.3 USER REQUIREMENTS AND SCENARIOS

The scenarios covered by this profile are the following:

- Usage of the Client to browse the object store of the Server. Clients are required to pull and understand Folder Listing Objects. Servers are required to respond to requests for Folder Listing Objects. Servers are required to have a root folder. Servers are not required to have a folder hierarchy below the root folder.
- Usage of the Client to transfer objects to and from the Server. The transfer of objects includes folders and files. Clients must support the ability to push or pull files from the Server. Clients are not required to push or pull folders. Servers are required to support file push, pull, or both. Servers are allowed to have read-only folders and files, which means they can restrict object pushes. Thus, Servers are not required to support folder push or pull.
- Usage of the Client to create folders and delete objects (folders and files) on the Server. Clients are not required to support folder/file deletion or folder creation. Servers are allowed to support read-only folders and files, which means they can restrict folder/file deletion and creation.

A device adhering to this profile must support Client capability, Server capability or both. The restrictions applying to this profile are the same as in the GOEP.

2.4 PROFILE FUNDAMENTALS

The profile fundamentals are the same as defined in Section 2.4 in GOEP [2]. Support for link level authentication and encryption is required but their use is optional.

Support for OBEX authentication is required but its use is optional.

This profile does not mandate the server or client to enter any discoverable or connectable modes automatically, even if they are able to do so.

On the Client side, end-user intervention is always needed to initiate file transfer (see Chapter 3).

Support of bonding is required but its use is optional.

3 USER INTERFACE ASPECTS

3.1 FILE TRANSFER MODE SELECTION, SERVERS

Servers must be placed in File Transfer mode. This mode enables a Client to perform file transfer operations with the Server. When entering this mode, File Transfer Servers should set the device in *Limited Discoverable* mode (see Generic Access Profile), ensure that the Object Transfer Bit is set in the CoD (see [15]), and register a service record in the SDDB (see section 6 on page 383).

It is recommended that this mode be set and unset by user interaction, when possible. Public devices, devices that want to be visible at all times, or devices that can not supply a user interface to enable File Transfer mode shall use *General Discoverable* mode (see Generic Access Profile) instead of *Limited Discoverable* mode.

3.2 FUNCTION SELECTION, CLIENTS

Clients provide file transfer functions to the user via a user interface. An example of a file transfer user interface is a file-tree viewer to browse folders and files. Using such a system file-tree viewer, the user can browse and manipulate files on another PC, which appears in the network view.

File Transfer Applications provide the following functions.

Select Server	Selecting the Server from a list of possible Servers, and setting up a connection to it.
Navigate Folders	Displaying the Server's folder hierarchy, including the files in the folders, and moving through the Server's folder hierarchy to select the current folder. The current folder is where items are pulled and/or pushed.
Pull Object	Copying a file or a folder from the Server to the Client.
Push Object	Copying a file or folder from the Client to the Server.
Delete Object	Deleting a file or folder on the Server.
Create Folder	Creating a new folder on the Server.

When the user selects the Select Server function, an inquiry procedure will be performed to produce a list of available devices in the vicinity. Requirements on inquiry procedures are discussed in Section 6.5.1 of the GOEP [2].

3.3 APPLICATION USAGE

In this section, the presented scenarios work as examples. Variations in the actual implementations are possible and allowed.

When the Client wants to select a Server the following user interaction can be followed:

Client	Server
	The user sets the device **into File Transfer mode**. A Server typically does not need to provide any other user interaction.
The user of the Client selects the **File Transfer Application** on the device.	
A list of Servers that may support the File Transfer service is displayed to the user.	
The user selects a Server in which to connect. The connection may require the user to enter a password for authentication. If both link level authentication and OBEX authentication is required, then the user will need to be prompted for two passwords.	If the Client requires authentication of the Server, then the Server will need to prompt the user for a password. If both link level authentication and OBEX authentication are required, then the user will need to be prompted for two passwords.
After the connection is complete, including any authentication, the contents of the Server's root folder are displayed.	

The following user interaction shows how the user of the Client performs file transfer functions. The operations assume a Server has already been selected as described above.

Client	Server
The user is presented with the folder hierarchy of the Server. The first presentation has the root folder selected as the current folder.	
The user chooses a folder to be the current folder. The contents of this folder are displayed.	
To push a file from the Client to the Server, the user selects a file on the Client and activates the **Push Object** function. The object is transferred to the current folder on the Server.	
To pull a file from the Server, the user selects a file in the current folder of the Server and activates the **Pull Object** function. The user is notified of the result of the operation.	
To delete a file on the Server, the user selects the file in the Server's current folder and activates the **Delete Object** function. The user is notified of the result of the operation.	
To create a new folder on the Server, the user activates the **Create Folder** function. This function requests a name from the user for the folder. When complete, a new folder is created in the Server's current folder.	

4 APPLICATION LAYER

This section describes the feature requirements on units active in the File Transfer use case.

4.1 FEATURE OVERVIEW

The File Transfer application is divided into three main features, as shown in the Table 4.1 below.

	Features	Support in File Transfer Client	Support in File Transfer Server
1.	Folder Browsing	M	M
2.	Object Transfer: File Transfer Folder Transfer	 M O	 M O*
3.	Object Manipulation	O	O*

Table 4.1: Application layer procedures

*. Optional, but the server must be able to respond with an appropriate error code, even if it doesn't support these capabilities.

4.2 FOLDER BROWSING

A file transfer session begins with the Client connecting to the Server and pulling the contents of the Server's root folder. When an OBEX connection is made, the Server starts out with its current folder set to the root folder. The contents of folders must be transferred in the Folder Listing format specified in [11].

Table 4.2 shows the application procedure required by the Client to connect to the Server and pull the contents of the root folder.

Client	Details
OBEX CONNECT.	Target Header must be set to the Folder Browsing UUID: F9EC7BC4-953C-11D2-984E-525400DC9E09. This UUID is sent in binary (16 bytes) with most significant byte sent first (0xF9 is sent first).
Pull the contents of the Server's root folder using GET.	The Type Header must be set to the MIME-type of the Folder Listing Object (x-obex/folder-listing). The Connect ID header must be set to the value returned in the Connect operation. A Name header is not used.

Table 4.2: Application layer procedure for File Transfer Connect

Browsing an object store involves displaying folder contents and setting the 'current folder'. The OBEX SETPATH command is used to set the current folder. To display a folder hierarchy starting with the root folder, the Client must read the root folder contents using GET. It must then retrieve the contents of all sub-folders using GET. If the sub-folders contain folders, then the Client must retrieve the contents of these folders and so on. To retrieve the contents of a folder, the Client must set the current folder to the sub-folder using SETPATH, then pull the sub-folder contents using GET. Table 4.3 shows the application procedure required for retrieving the contents of a sub-folder.

Client	Details
Set the current folder to the sub-folder using OBEX SETPATH.	Name header is set to the name of the sub-folder. Connect ID header is required.
Pull the contents of the sub-folder using GET.	No Name is sent, since the sub-folder is the current folder. The Type Header must be set to the MIME-type of the Folder Listing Object (x-obex/folder-listing). Connect ID header is required.
Set the current folder back to the root folder using OBEX SETPATH.	Name header is empty. Connect ID header is required.
If the parent of the sub-folder is not the root folder, then set the current folder to the parent folder using SETPATH.	The Backup flag is set and no Name header is sent. Connect ID header is required.

Table 4.3: Application layer procedure for Folder Browsing

4.3 OBJECT TRANSFER

Objects are transferred from the Client to the Server using OBEX PUT, and objects are transferred from the Server to the Client using OBEX GET. Transferring files requires a single PUT or GET operation per file. Transferring folders requires transferring all the items stored in a folder, including other folders. The process of transferring a folder may require that new folders be created. The SETPATH command is used to create folders.

Table 4.4 shows the application procedure for transferring a folder from the Client to the Server. If the folder contains other folders, then these other folders are transferred using the same method. The folder is transferred to the current folder on the Server.

Client	Details
Create a new folder (if it does not already exist) in the Server's current folder using SETPATH. The current folder is changed to this new folder.	Name header is set to the name of the new folder. Connect ID header is required.
Push all files to the new folder using a PUT command for each file.	The Name header is set to the name of the file. Connect ID header is required.
Folders are created using SET-PATH.	Name header is set to folder name. This application procedure is applied recursively to each folder. Connect ID header is required.
Set the current folder back to the parent folder using SETPATH.	The Backup flag is set and no Name header is sent. Connect ID header is required.

Table 4.4: Application layer procedure for Pushing a Folder

Table 4.5 shows the application procedure for transferring a folder from the Server to the Client.

Client	Details
Set the current folder to the folder which is to be transferred using SETPATH.	The Name header is set to the name of the folder. Connect ID header is required.
Pull the contents of the folder using GET.	A Name header is not sent, and the Type Header must be set to the MIME-type of the Folder Listing Object (x-obex/folder-listing).
Pull all files to the new folder using a GET command for each file.	The Name header is set to the name of the file. Connect ID header is required.
Pull all Folders to the new folder using this application procedure.	This application procedure is applied recursively to each folder.
Set the current folder back to the parent folder, using SETPATH.	The Backup flag is set and no Name header is sent. Connect ID header is required.

Table 4.5: Application layer procedure for Pulling a Folder

4.4 OBJECT MANIPULATION

A Client can delete folders and files on a Server. It can also create new folders on a Server. A brief summary of these functions is shown below.

- A file is deleted by using a PUT command with the name of the file in a Name header and no Body header.
- An empty folder is deleted by using a PUT command with the name of the folder in a Name header and no Body header.
- A non-empty folder can be deleted in the same way as an empty folder but Servers may not allow this operation. If a Server refuses to delete a non-empty folder it must return the "Precondition Failed" (0xCC) response code. This response code tells the Client that it must first delete all the elements of the folder individually before deleting the folder.
- A new folder is created in the Server's current folder by using the SETPATH command with the name of the folder in a Name header. If a folder with that name already exists, then a new folder is not created. In both cases the current folder is set to the new folder.

5 OBEX

5.1 OBEX OPERATIONS USED

Table 5.1 shows the OBEX operations, which are required in the File Transfer profile.

Operation no.	OBEX Operation	Client	Server
1	Connect	M	M
2	Disconnect	M	M
3	Put	M	M
4	Get	M	M
5	Abort	M	M
6	SetPath	M	M

Table 5.1: OBEX Operations

5.2 OBEX HEADERS

Table 5.2 shows the specified OBEX headers, which are required in the File Transfer profile.

Header no.	OBEX Headers	Client	Server
1	Count	O	O
2	Name	M	M
3	Type	M	M
4	Length	M	M
5	Time	O	O
6	Description	O	O
7	Target	M	M
8	HTTP	O	O
9	Body	M	M
10	End of Body	M	M
11	Who	M	M
12	Connection ID	M	M
13	Authenticate Challenge	M	M
14	Authenticate Response	M	M
15	Application Parameters	X	X
16	Object Class	X	X

Table 5.2: OBEX Headers

5.3 INITIALIZATION OF OBEX

Devices implementing the File Transfer profile can optionally use OBEX authentication. The initialization procedure is defined in Section 5.3 of GOEP [2].

5.4 ESTABLISHMENT OF OBEX SESSION

The OBEX connection must use a Target header set to the File Browsing UUID, F9EC7BC4-953C-11D2-984E-525400DC9E09. This UUID is sent in binary (16 bytes) with 0xF9 sent first. OBEX authentication can optionally be used. This profile follows the procedures described in Section 5.4 of GOEP [2] with the Target, Connection ID, and Who headers being mandatory.

5.5 BROWSING FOLDERS

Browsing folders involves pulling Folder Listing objects and setting the current folder. Navigating a folder hierarchy requires moving forward and backward by changing the current folder. Upon completion of the OBEX Connect operation the Server's current folder is the root folder.

5.5.1 Pulling a Folder Listing Object

Pulling a Folder Listing object uses a GET operation and follows the procedure described in Section 5.6 of GOEP [2]. The Connection ID and Type headers are mandatory. A Name header containing the name of the folder is used to pull the listing of a folder. Sending the GET command without a name header is used to pull the contents of the current folder. Typically, a folder browsing application will pull the contents of the current folder, so a Name header is not used. The Type header must be set to 'x-obex/folder-listing'.

5.5.2 Setting the Current Folder (Forward)

Setting the current folder requires the SETPATH operation. The SETPATH request must include the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Opcode for SETPATH	0x82	M	-
Field	Packet Length	Varies	M	-
Field	Flags	0x02	M	'Backup level' flag is set to 0 and 'Don't Create' flag is set to 1.
Field	Constants	0x00	M	Constants are not used, and the field must be set to 0.
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This must be the first header.
Header	Name	Varies	M	Name of the folder.

Table 5.3: Fields and Headers in SETPATH Request for Setting Current Folder (Forward)

The response packet for the SETPATH request has the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Response code for SETPATH	0xA0 or 0xC4	M	0xA0 for success or 0xC4 if the folder does not exist.
Field	Packet Length	Varies	M	-

Table 5.4: Fields and Headers in SETPATH Response for Setting Current Folder (Forward)

Other headers such as Description can optionally be used.

5.5.3 Setting the Current Folder (Backward)

Setting the current folder back to the parent folder requires the SETPATH operation. The SETPATH request must include the following fields and headers (note that a Name header is not used):

Field/ Header	Name	Value	Status	Explanation
Field	Opcode for SET-PATH	0x82	M	-
Field	Packet Length	Varies	M	-
Field	Flags	0x03	M	'Backup level' flag is set to 1 and 'Don't Create' flag is set to 1.
Field	Constants	0x00	M	Constants are not used, and the field must be set to 0.
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This must be the first header.

Table 5.5: Fields and Headers in SETPATH Request for Setting Current Folder (Backward)

The response packet for the SETPATH request has the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Response code for SETPATH	0xA0 or 0xC4	M	0xA0 for success, or 0xC4 if the current folder is the root.
Field	Packet Length	Varies	M	-

Table 5.6: Fields and Headers in SETPATH Response for Setting Current Folder (Backward)

Other headers, such as Description, can optionally be used.

5.5.4 Setting the Current Folder (Root)

Setting the current folder to the root requires the SETPATH operation. The SETPATH request must include the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Opcode for SET-PATH	0x82	M	-
Field	Packet Length	Varies	M	-
Field	Flags	0x02	M	'Backup level' flag is set to 0 and 'Don't Create' flag is set to 1.
Field	Constants	0x00	M	Constants are not used, and the field must be set to 0.
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This must be the first header.
Header	Name	Empty	M	Name header is empty.

Table 5.7: Fields and Headers in SETPATH Request for Setting Current Folder (Root)

The response packet for the SETPATH request has the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Response code for SETPATH	0xA0	M	0xA0 for success.
Field	Packet Length	Varies	M	-

Table 5.8: Fields and Headers in SETPATH Response for Setting Current Folder (Root)

Other headers, such as Description, can optionally be used.

5.6 PUSHING OBJECTS

Pushing object involves pushing files and folders.

5.6.1 Pushing Files

Pushing files follows the procedure described in Section 5.5 of GOEP [2]. The Connection ID header is mandatory.

5.6.2 Pushing Folders

Pushing folders involves creating new folders and pushing files. It may also involve navigating through the folder hierarchy. Navigation is described in Section 5.5 on page 376. Pushing files is described in Section 5.6.1 on page 379.

5.6.2.1 Creating New Folders

Creating a new folder requires the SETPATH operation. The SETPATH request must include the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Opcode for SET-PATH	0x82	M	-
Field	Packet Length	Varies	M	-
Field	Flags	0x00	M	'Backup level' flag is set to 0 and 'Don't Create' flag is set to 0.
Field	Constants	0x00	M	Constants are not used, and the field must be set to 0.
Header	Connection ID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This must be the first header.
Header	Name	Varies	M	Name of the folder.

Table 5.9: Fields and Headers in SETPATH Request for Creating a Folder.

The response packet for the SETPATH request has the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Response code for SETPATH	0xA0 or 0xC1	M	0xA0 for success or 0xC1 if the current folder is read only and creation of a new folder is unauthorized.
Field	Packet Length	Varies	M	-

Table 5.10: Fields and Headers in SETPATH Response for Creating a Folder

Other headers such as Description can optionally be used.

5.7 PULLING OBJECTS

Pulling objects involves pulling files and folders.

5.7.1 Pulling Files

Pulling files follows the procedure described in Section 5.6 of GOEP [2]. The Connect ID header is mandatory.

5.7.2 Pulling Folders

Pulling folders involves navigating the folder hierarchy, pulling folder listing objects and pulling files. Navigating the folder hierarchy and pulling folder listing-objects is described in Section 5.5 on page 376. Pulling files is described in Section 5.7.1 on page 381.

5.8 MANIPULATING OBJECTS

Manipulating objects includes deleting objects and creating new folders. Creating new folders is described in Section 5.6.2.1 on page 380, Creating New Folders. Deleting objects involves deleting files and folders.

5.8.1 Deleting Files

Deleting a file requires the PUT operation. The PUT request must include the following fields and headers (note that no Body or End Body headers are sent):

Field/ Header	Name	Value	Status	Explanation
Field	Opcode for PUT	0x82	M	-
Field	Packet Length	Varies	M	-
Header	ConnectionID	Varies	M	Connection ID is set to the value returned by the Server during the OBEX Connect operation. This must be the first header.
Header	Name	Varies	M	The header value is the name of the object to delete.

Table 5.11: Fields and Headers in PUT Request for Delete

The response packet for the PUT request has the following fields and headers:

Field/ Header	Name	Value	Status	Explanation
Field	Response code for PUT	0xA0, 0xC1 or 0xC4	M	0xA0 for success, 0xC1 for unauthorized (e.g. read only) or 0xC4 if the file does not exist.
Field	Packet Length	Varies	M	-

Table 5.12: Fields and Headers in PUT Response for Delete

Other headers such as Description can optionally be used.

5.8.2 Deleting Folders

A folder can be deleted using the same procedure used to delete a file (see Section 5.8.1 on page 381). Deleting a non-empty folder will delete all its contents, including other folders. Some Servers may not allow this operation and will return the “Precondition Failed” (0xCC) response code, indicating that the folder is not empty. In this case the Client will need to delete the contents before deleting the folder.

5.9 DISCONNECTION

See Section 5.7 in GOEP [2].

6 SERVICE DISCOVERY

6.1 SD SERVICE RECORDS

The service belonging to the File Transfer profile is a server, which enables bi-directional generic file transfer. OBEX is used as a session protocol for this service. The following service records must be put into the SDDB.

Item	Definition:	Type/ Size:	Value:*	AttrID	Status	Default Value
Service Class ID List				See [15]	M	
Service Class #0		UUID	OBEX-File Transfer		M	
Protocol Descriptor list				See [15]	M	
Protocol ID #0		UUID	L2CAP		M	
Protocol ID #1		UUID	RFCOMM		M	
Param #0	CHANNEL	Uint8	Varies		M	
Protocol ID #2		UUID	OBEX		M	
Service name	Displayable Text name	String	Varies	See [15]	O	"OBEX File Transfer"
BluetoothProfileDescriptorList				See [15]	O	
Profile ID #0	Supported profile	UUID	OBEX File-Transfer			OBEX File Transfer [15]
Param #0	Profile version	uint16	0x100			0x100

Table 6.1: File Transfer Service Record

* UUID values are defined in the *Assigned Numbers* document.

6.2 SDP PROTOCOL DATA UNITS

Table 19 shows the specified SDP PDUs (Protocol Data Units) which are required in the File Transfer profile.

PDU no.	SDP PDU	Server	Client
1	SdpErrorResponse	M	M
2	SdpServiceSearch AttributeRequest	M	M
3	SdpServiceSearch AttributeResponse	M	M

Table 6.2: SDP PDUs Minimal Requirements

7 REFERENCES

7.1 NORMATIVE REFERENCES

[1] Bluetooth Special Interest Group, IrDA Interoperability

[2] Bluetooth Special Interest Group, Generic Object Exchange Profile

[3] Bluetooth Special Interest Group, Synchronization Profile

[4] Bluetooth Special Interest Group, Object Push Profile

[5] Bluetooth Special Interest Group, Baseband Specification

[6] Bluetooth Special Interest Group, LMP Specification

[7] Bluetooth Special Interest Group, L2CAP Specification

[8] Bluetooth Special Interest Group, RFCOMM with TS 07.10

[9] ETSI, TS 07.10, Version 6.3.0

[10] Bluetooth Special Interest Group, SDP Specification

[11] Infrared Data Association, IrDA Object Exchange Protocol (IrOBEX) with Published Errata, Version 1.2, April 1999.

[12] Infrared Data Association, IrMC (Ir Mobile Communications) Specification with Published Errata, Version 1.1, February 1999.

[13] The Internet Mail Consortium, vCard – The Electronic Business Card Exchange Format, Version 2.1, September 1996.

[14] The Internet Mail Consortium, vCalendar – The Electronic Calendaring and Scheduling Exchange Format, Version 1.0, September 1996.

[15] Bluetooth Special Interest Group, Assigned Numbers specification

[16] Bluetooth Special Interest Group, Bluetooth Generic Access Profile Specification

Part K:13

SYNCHRONIZATION PROFILE

This application profile defines the application requirements for Bluetooth devices necessary for the support of the Synchronization usage model. The requirements are expressed in terms of end-user services, and by defining the features and procedures that are required for interoperability between Bluetooth devices in the Synchronization usage model.

CONTENTS

FOREWORD

This document, together with the Generic Object Exchange profile and the Generic Access profile forms the Synchronization usage model.

Interoperability between devices from different manufacturers is provided for a specific service and usage model if the devices conform to a Bluetooth-SIG defined profile specification. A profile defines a selection of messages and procedures (generally termed *capabilities*) from the Bluetooth SIG specifications and gives an unambiguous description of the air interface for specified service(s) and usage model(s).

All defined features are process-mandatory. This means that if a feature is used, it is used in a specified manner. Whether the provision of a feature is mandatory or optional is stated separately for both sides of the Bluetooth air interface.

1 INTRODUCTION

1.1 SCOPE

The Synchronization profile defines the requirements for the protocols and procedures that shall be used by the applications providing the Synchronization usage model. This profile makes use of the Generic Object Exchange profile (GOEP) to define the interoperability requirements for the protocols needed by applications. The most common devices using these usage models might be notebook PCs, PDAs, and mobile phones.

The scenarios covered by this profile are:

- Usage of a mobile phone or PDA by a computer to exchange PIM (Personal Information Management) data, including a necessary log information to ensure that the data contained within their respective Object Stores is made identical. Types of the PIM data are, for example, phonebook and calendar items.
- Use of a computer by a mobile phone or PDA to initiate the previous scenario (Sync Command Feature).
- Use of a mobile phone or PDA by a computer to automatically start synchronization when a mobile phone or PDA enters the RF proximity of the computer

1.2 BLUETOOTH PROFILE STRUCTURE

In Figure 1.1, the Bluetooth profile structure and the dependencies of the profiles are depicted. A profile is dependent upon another profile if it re-uses parts of that profile, by implicitly or explicitly referencing it. Dependency is illustrated in the figure: a profile has dependencies on the profile(s) in which it is contained – directly and indirectly.

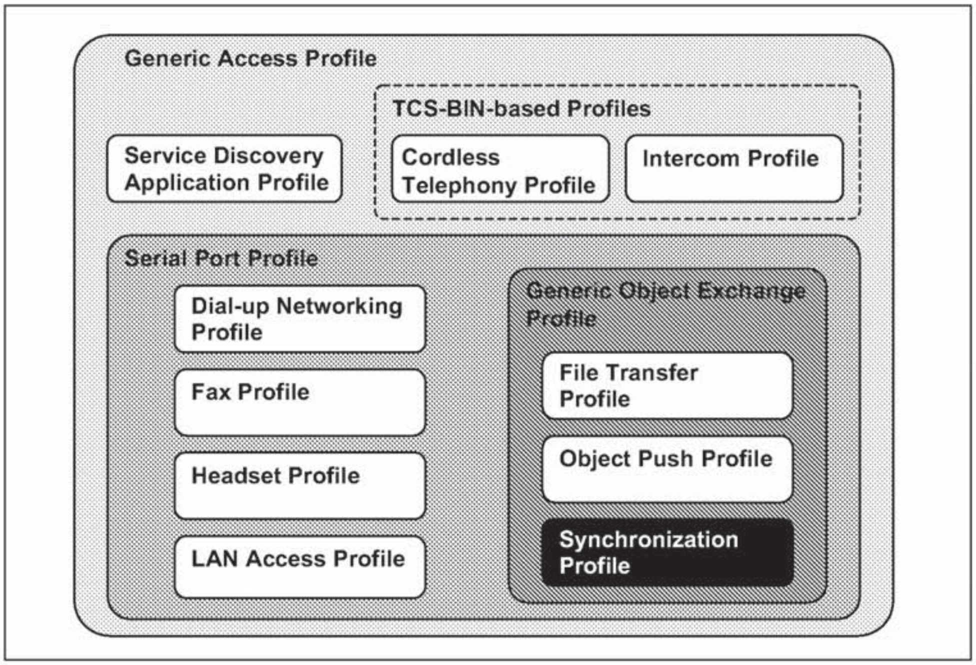

Figure 1.1: Bluetooth Profiles

1.3 BLUETOOTH OBEX RELATED SPECIFICATIONS

Bluetooth Specification includes five separate specifications for OBEX and applications using OBEX.

1. Bluetooth IrDA Interoperability Specification [1].

- Defines how the applications can function over both Bluetooth and IrDA.
- Specifies how OBEX is mapped over RFCOMM and TCP.
- Defines the application profiles using OBEX over Bluetooth.

2. Bluetooth Generic Object Exchange Profile Specification [2]

- Generic interoperability specification for the application profiles using OBEX.
- Defines the interoperability requirements of the lower protocol layers (e.g. Baseband and LMP) for the application profiles

3. Bluetooth Synchronization Profile Specification (This Specification)

- Application Profile for Synchronization applications.
- Defines the interoperability requirements for the applications within the Synchronization application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

4. Bluetooth File Transfer Profile Specification [3]

- Application Profile for File Transfer applications.
- Defines the interoperability requirements for the applications within the File Transfer application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

5. Bluetooth Object Push Profile Specification [4]

- Application Profile for Object Push applications.
- Defines the interoperability requirements for the applications within the Object Push application profile.
- Does not define the requirements for the Baseband, LMP, L2CAP, or RFCOMM.

1.4 SYMBOLS AND CONVENTIONS

1.4.1 Requirement status symbols

In this document, the following symbols are used:

'M' for mandatory to support (used for capabilities that shall be used in the profile);

'O' for optional to support (used for capabilities that can be used in the profile);

'C' for conditional support (used for capabilities that shall be used in case a certain other capability is supported);

'X' for excluded (used for capabilities that may be supported by the unit but shall never be used in the profile);

'N/A' for not applicable (in the given context it is impossible to use this capability).

Some excluded capabilities are capabilities that, according to the relevant Bluetooth specification, are mandatory. These are features that may degrade operation of devices following this profile. Therefore, these features shall never be activated while a unit is operating as a unit within this profile.

1.4.2 Signaling diagram conventions

The following arrows are used in diagrams describing procedures:

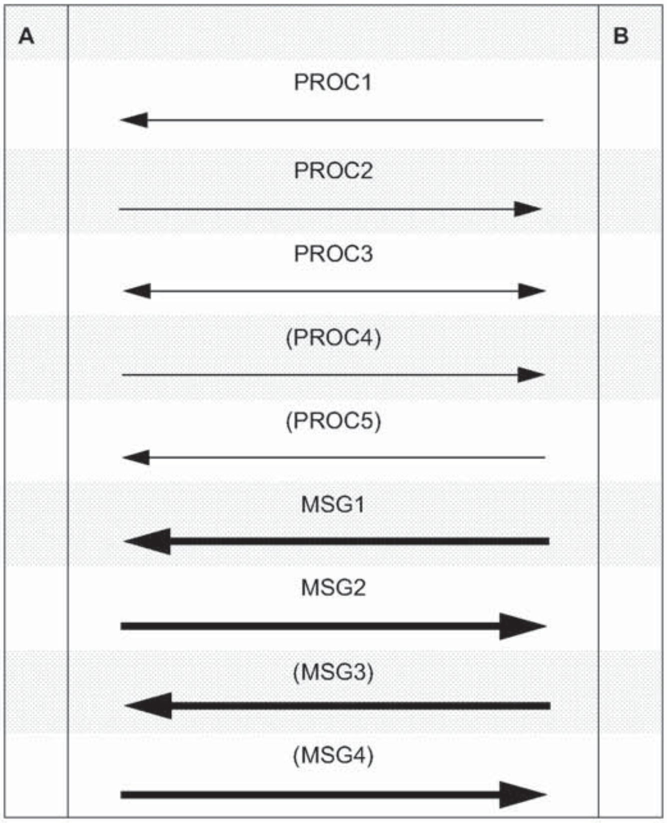

Table 1.1: Arrows used in signaling diagrams

In the table above, the following cases are shown: PROC1 is a sub-procedure initiated by B. PROC2 is a sub-procedure initiated by A. PROC3 is a sub-procedure where the initiating side is undefined (may be both A and B). PROC4 indicates an optional sub-procedure initiated by A, and PROC5 indicates an optional sub-procedure initiated by B.

MSG1 is a message sent from B to A. MSG2 is a message sent from A to B. MSG3 indicates an optional message from A to B, and MSG4 indicates an optional message from B to A.

2 PROFILE OVERVIEW

2.1 PROFILE STACK

The figure below shows the protocols and entities used in this profile.

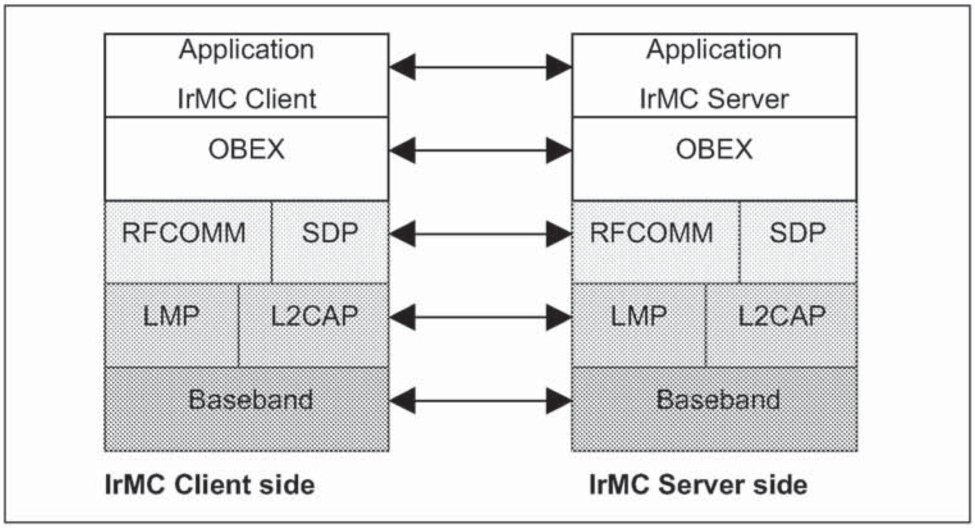

Figure 2.1: Protocol model

The Baseband [5], LMP [6] and L2CAP [7] are the OSI layer 1 and 2 Bluetooth protocols. RFCOMM [8] is the Bluetooth adaptation of GSM TS 07.10 [9]. SDP is the Bluetooth Service Discovery Protocol [10]. OBEX [1] is the Bluetooth adaptation of IrOBEX [11].

The IrMC Client layer shown in Figure 2.1 is the entity processing the synchronization according to the IrMC specification [12], and the IrMC server is the server software compliant to the IrMC specification.

The RFCOMM, L2CAP, LMP, and Baseband interoperability requirements are defined in Section 6 in GOEP[2].

2.2 CONFIGURATIONS AND ROLES

Figure 2.2 depicts a synchronization example in which a mobile phone acts as an IrMC server and a PC notebook as an IrMC Client. The IrMC Client (PC) pulls the PIM data from the IrMC server and synchronizes this data with data stored in the IrMC client. After that, the IrMC client puts this synchronized data back to the IrMC server.

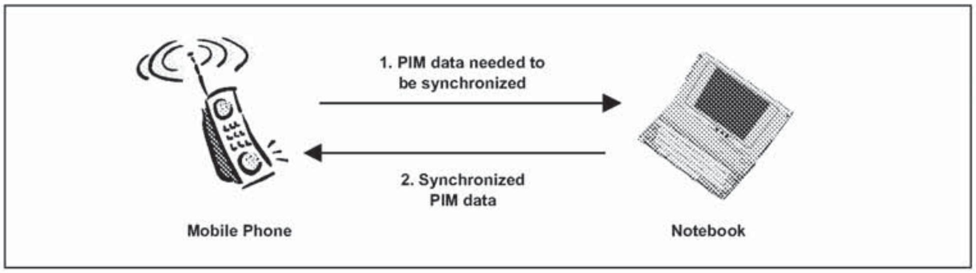

Figure 2.2: Synchronization Example with Mobile Phone and Computer

The following roles are defined for this profile:

IrMC Server – This is the IrMC server device that provides an object exchange server. Typically, this device is a mobile phone or PDA. In addition to the interoperability requirements defined in this profile, the IrMC server must comply with the interoperability requirements for the server of the GOEP, if not defined to the contrary.

If the IrMC Server also provides the functionality to initiate the synchronization, then it must act as a client temporarily. In this case, it must also comply with the requirements with the client of the GOEP if not defined in the contrary.

IrMC Client – This is the IrMC client device, which contains a sync engine and pulls and pushes the PIM data from and to the IrMC Server. Usually, the IrMC Client device is a PC. Because the IrMC Client must also provide functionality to receive the initialization command for synchronization, sometimes it must temporarily act as a server. In addition to the interoperability requirements defined in this profile, the IrMC server must also comply with the interoperability requirements for the server and client of the GOEP if not defined to the contrary.

2.3 USER REQUIREMENTS AND SCENARIOS

The scenarios covered by this profile are:

- Usage of an IrMC Server by an IrMC Client to pull the PIM data needed to be synchronized from the IrMC Server, to synchronize this data with the data on the IrMC Client, and to push this synchronized data back to the IrMC Server.
- Usage of an IrMC Client by an IrMC Server to initiate the previous scenario by sending a sync command to the IrMC Client.
- Automatic synchronization initiated by the IrMC client.

The restrictions applying to this profile are the same as in the GOEP. In addition to these restrictions, the peer-to-peer synchronization is not supported by the BT synchronization.

2.4 PROFILE FUNDAMENTALS

The profile fundamentals are the same as defined in Section 2.4 in GOEP [2], with the addition of the requirements that bonding, link level authentication, and encryption (Fundamentals 1 and 3 in GOEP) must always be used for this profile. The OBEX authentication (Fundamental 2 in GOEP) as an application-level security mechanism must be supported by the devices providing this profile, but this profile does not mandate that it must be used.

In this profile, because both the IrMC Client and IrMC Server can act as a client (IrMC Server temporarily), both can initiate link and channel establishments; i.e. create a physical link between these two devices.

This profile does not mandate the IrMC server or client to enter any discoverable or connectable modes automatically, even if they are able to do so. This means that the end-user intervention may be needed on both the devices when, for example, the synchronization is initiated on the IrMC client device.

3 USER INTERFACE ASPECTS

3.1 MODE SELECTION

There are two modes associated with the Synchronization profile.

- Initialization Sync mode
- General Sync mode

In the **Initialization Sync** mode, the IrMC Server is in the Limited discoverable (or the General discoverable mode, see Section 6.5.1 in GOEP [2]), Connectable, and Pairable modes (See Section 4 in GAP [16]). The IrMC Client does not enter this mode in this profile. It is recommended that the Limited Inquiry procedure (Section 6.2 in GAP[16]) is used by the IrMC Client when discovering the IrMC server. Requirements on inquiry procedures are discussed in Section 6.5.1 of the GOEP [2].

In the **General Sync** mode, the device is in the Connectable mode. Both the IrMC Client and Server can enter this mode. For the IrMC Server, this mode is used when the IrMC Client connects the server and starts the synchronization at the subsequent times after pairing. For the IrMC Client, the mode is used when the synchronization is initiated by the IrMC server.

The devices are not required to enter these modes automatically without user intervention, even if they can do so. When entering either of these modes, IrMC Server and Client must ensure that the Object Transfer bit is set in the CoD (See [15]), and register a service record in the SDDB (See Section 7).

3.2 APPLICATION USAGE EVENTS

In the following sections (Section 3.2.1-3.2.3), the presented scenarios work as examples and variations in the actual implementations are possible and allowed.

3.2.1 Synchronization Scenario

When an IrMC Client wants to synchronize with an IrMC Server for the first time, the following scenario (Table 3.1) can be followed:

Step	IrMC Client	IrMC Server
1		The IrMC server device must be in the General Sync mode. If the device is not in this mode, the user must activate this mode on the device.
2	The user activates an application for synchronization.	
3	A list of devices in the RF proximity of the IrMC client is displayed to the user.	
4	The user selects a device to be connected and synchronized.	
5	The user is alerted if the device does not support the Synchronization feature, and the user may select another possible device (Step 4).	
6	The Bluetooth PIN code is requested from the user and entered on both devices.	
7	If OBEX authentication is used, the user enters the password for the OBEX authentication on both devices.	
8	The first synchronization is processed.	
9	The user may be notified of the result of the operation.	

Table 3.1: Usage Events for First Time Synchronization

At subsequent times, when the bonding is done, the scenario below (Table 3.2) can be followed.:

Step	IrMC Client	IrMC Server
1		The IrMC server device must be in the General Sync mode. If the device is not in this mode, the user must activate this mode on the device.
2	The user of the IrMC Client selects the Synchronization feature on the device, or another event triggers the synchronization to start in the IrMC client.	
3	The synchronization is processed.	
4	The User may be notified of the result of the operation.	

Table 3.2: Usage Events after First Time Synchronization

3.2.2 Sync Command Scenario

When an IrMC Server wants to initiate synchronization, and when the bonding and the possible OBEX initialization are done, the scenario below (Table 3.3) can be followed:

Step	IrMC Client	IrMC Server
1	The IrMC Client should be in the General Sync mode, without user intervention. Otherwise this operation is not applicable.	
2		The user selects the Sync Command feature in the IrMC Server, and the synchronization is initiated with the IrMC client. On the IrMC Server device, the user has earlier configured the IrMC Client to which the sync command is sent.
3	The synchronization is processed.	
4		The User may be notified of the result of the operation.

Table 3.3: Usage Events of Sync Command Scenario

3.2.3 Automatic Synchronization Scenario

When it is desired that an IrMC Server and Client synchronize automatically, and when the bonding and (possible) OBEX initialization are done, the scenario below (Table 3.4) can be followed.

Step	IrMC Client	IrMC Server
1	The IrMC Server enters the RF proximity of the IrMC Client. The Client notices it, and starts the synchronization without any notification to the User. The IrMC Server must be constantly in the General Sync mode so that the IrMC Client can notice the presence of the server in its RF vicinity.	
2	The synchronization is processed.	
3	The User may be notified of the result of the operation on both the devices.	

Table 3.4: Usage Events of Automatic Synchronization Scenario

4 APPLICATION LAYER

This section describes the feature requirements on units active in the Synchronization use case.

4.1 FEATURE OVERVIEW

Table 4.1 shows the required services:

	Features	Support in IrMC Client	Support in IrMC Server
1.	Synchronization of one or more of the following cases:	M	M
	Synchronization of phonebooks		
	Synchronization of calendars		
	Synchronization of emails		
	Synchronization of notes		
2.	Sync Command	M	O
3.	Automatic Synchronization	O	M

Table 4.1: Application layer features

4.2 SYNCHRONIZATION FEATURE

The support of Synchronization with IrMC level 4 functionality is mandatory for both IrMC Clients and IrMC Servers. The requirements for IrMC Synchronization are defined in the IrMC spec (See also Section 5). Bluetooth Synchronization must support at least one of the following cases (i.e. the application classes):

1. Synchronization of phonebooks
2. Synchronization of calendars
3. Synchronization of messages
4. Synchronization of notes

To achieve application level interoperability, the content formats are defined for Bluetooth Synchronization. The content formats are dependent on the application classes, which are designed for the different purposes. The supported application classes must be identified in terms of the data stores in the SDDB of the IrMC Server (See Section 7.1.1). For the application classes the content format requirements are:

- Phone Book applications must support data exchange using the vCard 2.1 content format specified in [13]. Section 7 of IrMC Specification [12] includes extensions to vCard2.1, which must also be supported by the actual implementations.
- Calendar applications must support data exchange using the vCalendar 1.0 content format specified in [14].
- Messaging applications must support data exchange using the vMessage content format in Section 9 of [12].
- Notes applications must support data exchange using the vNote content format specified in Section 10 of [12].

The above requirements are the minimal requirements, and the application utilizing any of these classes may store its objects in any internal content format the implementer chooses.

The support for the various mandatory and optional fields of the content formats listed above shall be in accordance with the IrMC Specification [12].

4.3 SYNC COMMAND FEATURE

This feature means that the IrMC client device works temporarily as a server and is able to receive a Sync Command from the IrMC server, which in this case acts temporarily as a client. This Sync Command orders the IrMC client to start synchronization with the IrMC Server.

After sending the sync command and getting the response for it, the IrMC Server must terminate the OBEX session and the RFCOMM data link connection.

This feature must be supported by the IrMC Client and it can optionally be supported by the IrMC Server. The formal requirements for this feature are defined in Section 5.8 in [12].

4.4 AUTOMATIC SYNCHRONIZATION FEATURE

In this feature, the IrMC Client can start the synchronization when the IrMC Server enters the RF proximity of the IrMC Client. Basically, this means that, on the Baseband level, the IrMC Client pages the IrMC Server at intervals and, when it finds that the IrMC Server is in the range, the IrMC Client can begin synchronization.

The support of this feature is optional for the IrMC Client but mandatory for the IrMC Server. This means that the IrMC Server must offer a capability to put the server device into the General Sync mode so that it does not leave this mode automatically.

5 IRMC SYNCHRONIZATION REQUIREMENTS

The IrMC specification [12] specifies IrMC Synchronization, which is utilized by this profile. The sections of the IrMC specification, with which this profile complies, are defined in Table 5.1.

Chapter	Name	Informative Sections	Mandatory Sections	Optional Sections	Not Applicable Sections
1	Introduction	All	-	-	-
2	IrMC Framework	2.1-3, 2.5.1, and 2.6-7	2.8.1-2, 2.8.4, and 2.9 (except 2.9.2)	2.8.3, and 2.9.2	2.4 and 2.5.2-3
3	Data Transmissions Services	3.3	3.1	-	3.2
4	OBEX Information Access and Indexing	4.1, 4.4.2, and 4.7	4.1.2, 4.2-3, 4.6, and 4.8	4.1.1 and 4.5	4.4.1
5	Synchronization	5.1 and 5.7	5.2-6 (except 5.5.3), and 5.8	5.5.3	-
6	Device Information	-	6.1-2	-	
7	Phone Book	7.1	7.3, 7.5, 7.7.1, 7.7.3, 7.7.5, 7.8.1, and 7.8.2	7.4, 7.6, 7.7.4, 7.7.6, and 7.8.3-5	7.2 and 7.7.2
8	Calendar	8.1	8.3, 8.5, 8.6.1, 8.6.3, 8.6.5, and 8.7	8.4 and 8.6.4	8.2, and 8.6.2
9	Messaging	9.1	9.3, 9.5, 9.8.1, 9.8.3, 9.8.6, and 9.9-10	9.4, 9.6-7, 9.8.4, and 9.8.5	9.2, and 9.8.2
10	Notes	10.1	10.3, 10.5, 10.6.1, 10.6.3, 10.6.5, and 10.7	10.4 and 10.6.4	10.2, and 10.6.2

Table 5.1: IrMC Specification Dependencies

Chapter	Name	Informative Sections	Mandatory Sections*	Optional Sections	Not Applicable Sections
11	Call Control	-	-	-	ALL
12	Audio	-	-	-	ALL
13	IrMC Applications IAS Entry and Service Hint Bit	-	-	-	ALL

Table 5.1: IrMC Specification Dependencies

*. Some of these sections may not be mandatory if the applications do not support all of the applications classes

This profile does not mandate that the functionality of IrMC level 1 must be supported for the different personal data objects (vcard, vcal, vmessage and vnote), although the IrMC specification requires its support. However, it is worth mentioning that the Push command of IrMC requires the level1 functionality for a text message. Thus, the IrMC client must be able to receive this command into its Inbox and the IrMC server must be able to send this command, if support for the Sync Command feature is claimed. For Bluetooth, the object push functionality and requirements are defined in the Object Push profile.

6 OBEX

6.1 OBEX OPERATIONS USED

Table 6.1 shows the OBEX operations which are required in the Synchronization profile.

Operation no.	OBEX Operation	Ability to Send		Ability to Respond	
		IrMC Client	IrMC Server*	IrMC Client*	IrMC Server
1	Connect	M	O	M	M
2	Disconnect	M	O	M	M
3	Put	M	O	M	M
4	Get	M	X	X	M
5	Abort	M	O	M	M
6	SetPath	X	X	X	X

Table 6.1: OBEX Operations

The columns marked with '*' refer to the Sync Command feature for which support in the IrMC Server is optional.

6.2 OBEX HEADERS

Table 6.2 shows the specified OBEX headers which are required in the Synchronization profile.

Header No.	OBEX Headers	IrMC Client	IrMC Server
1	Count	X	X
2	Name	M	M
3	Type	X	X
4	Length	M	M
5	Time	O	O
6	Description	O	O
7	Target	M	M

Table 6.2: OBEX Headers

Header No.	OBEX Headers	IrMC Client	IrMC Server
8	HTTP	O	O
9	Body	M	M
10	End of Body	M	M
11	Who	M	M
12	Connection ID	M	M
13	Authenticate Challenge	M	M
14	Authenticate Response	M	M
15	Application Parameters	M	M
16	Object Class	X	X

Table 6.2: OBEX Headers

6.3 INITIALIZATION OF OBEX

OBEX authentication must be supported by the devices implementing the Synchronization profile. The initialization procedure for OBEX is defined in Section 5.3 in GOEP [2].

6.4 ESTABLISHMENT OF OBEX SESSION

The Target header must be used when the IrMC client establishes the connection (See Section 5.4 in GOEP [2]). The Target header value is 'IRMC-SYNC'.

6.5 PUSHING DATA

See Section 5.5 in GOEP [2].

6.6 PULLING DATA

See Section 5.6 in GOEP [2].

6.7 DISCONNECTION

See Section 5.7 in GOEP [2].

7 SERVICE DISCOVERY

7.1 SD SERVICE RECORDS

There are two separate services related to the Synchronization profile. The first is the actual synchronization server (i.e. IrMC server), and the second is the sync command server (i.e. IrMC Client).

7.1.1 Synchronization Service

In this case, the service is the IrMC server. The following information (i.e. service records) must be put into the SDDB.

Item	Definition:	Type/ Size:	Value:*	AttrID:	Status:	Default Value:
Service Class ID List				See [15]	M	
Service Class #0		UUID	IrMCSync		M	
Protocol Descriptor list				See [15]	M	
Protocol ID #0		UUID	L2CAP		M	
Protocol ID #1		UUID	RFCOMM		M	
Param #0	CHANNEL	Uint8	Varies		M	
Protocol ID #2		UUID	OBEX		M	
Service name	Displayable Text name	String	Varies	See [15]	O	'IrMC Synchronization'
BluetoothProfileDescriptorList	Supported profiles and versions			See [15]	O	
Profile #0		UUID	IrMCSync			IrMC-Sync
Version #0		Uint16	Varies			0x0100
Supported Data Stores List	Data stores may be phonebook, calendar, notes, and messages.	Data Element Sequence of UInt8	Data stores: **0x01** = Phonebook **0x03** = Calendar **0x05** = Notes **0x06** = Messages	See [15]	M	

Table 7.1: Synchronization Service Record

*. Values that are of the type UUID are defined in the Assigned Numbers specification [15].

7.1.2 Sync Command Service

The Sync Command service is used for initiating the synchronization from the IrMC server device. The following service records must be put into the SDDB by the application which provides this service.

Item	Definition:	Type/ Size:	Value:*	AttrID:	Status:	Default Value:
Service Class ID List				See [15]	M	
Service Class #0		UUID	IrMCSync-Command		M	
Protocol Descriptor list				See [15]	M	
Protocol ID #0		UUID	L2CAP		M	
Protocol ID #1		UUID	RFCOMM	See [15]	M	
Param #0	CHANNEL	Uint8	Varies		M	
Protocol ID #2		UUID	OBEX		M	
Service name	Displayable Text name	String	Varies		O	'Sync Command Service'
BluetoothProfileDescriptorList	Supported profiles and versions			See [15]	O	
Profile #0		UUID	IrMCSync			IrMC-Sync
Version #0		Uint16	Varies			0x0100

Table 7.2: Sync Command Service Record

*. Values that are of the type UUID are defined in the Assigned Numbers specification [15].

7.2 SDP PROTOCOL DATA UNITS

Table 7.3 shows the specified SDP PDUs (Protocol Data Units) which are required in the Synchronization profile.

PDU no.	**SDP PDU**	**Ability to Send**		**Ability to Retrieve**	
		IrMC Client	**IrMC Server**	**IrMC Client**	**IrMC Server**
1	SdpErrorResponse	M*	M	M	O*
2	SdpServiceSearchAttribute-Request	M	O*	M*	M
3	SdpServiceSearchAttribute-Response	M*	M	M	O*

Table 7.3: SDP PDUs

The PDUs marked with '*' refer to the Sync Command feature, of which the support in the IrMC Server is optional.

8 REFERENCES

8.1 NORMATIVE REFERENCES

[1] Bluetooth Special Interest Group, IrDA Interoperability.

[2] Bluetooth Special Interest Group, Generic Object Exchange Profile.

[3] Bluetooth Special Interest Group, File Transfer Profile.

[4] Bluetooth Special Interest Group, Object Push Profile.

[5] Bluetooth Special Interest Group, Baseband Specification.

[6] Bluetooth Special Interest Group, LMP Specification.

[7] Bluetooth Special Interest Group, L2CAP Specification.

[8] Bluetooth Special Interest Group, RFCOMM with TS 07.10.

[9] ETSI, TS 07.10, Version 6.3.0.

[10] Bluetooth Special Interest Group, SDP Specification.

[11] Infrared Data Association, IrDA Object Exchange Protocol (IrOBEX) with Published Errata, Version 1.2, April 1999.

[12] Infrared Data Association, IrMC (Ir Mobile Communications) Specification with Published Errata, Version 1.1, February 1999.

[13] The Internet Mail Consortium, vCard – The Electronic Business Card Exchange Format, Version 2.1, September 1996.

[14] The Internet Mail Consortium, vCalendar – The Electronic Calendaring and Scheduling Exchange Format, Version 1.0, September 1996.

[15] Bluetooth Special Interest Group, Bluetooth Assigned Numbers.

[16] Bluetooth Special Interest Group, Bluetooth Generic Access Profile Specification.

Appendix I

REVISION HISTORY

REVISION HISTORY

Part K:1
Generic Access Profile

Rev	Date	Comments
1.0	June 20th 1999	• Released for final review. • Release for sign-off. • Updated based on received comments. • Final updated version. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:2
Service Discovery Application Profile

Rev	Date	Comments
1.0	June 20th 1999	• Aligned with GAP whenever necessary. • Emphasized that SDAP can be used as the basis for the service discovery portion of other profiles. • Added section 5.1 with SDP PDU exchange examples. • Emphasized that normal operation requires a LocDev to initiate and terminate L2CAP connections for SDP. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:3
Cordless Telephony Profile

Rev	Date	Comments
1.0	June 20th 1999	• Ftf preversion, some editorial updates and minor content changes - number of TLs 8 -> 7, Master-Slave Switch made conditional, restrictions in digits for Called&Calling party IEs, updates to CoD and SDP sections. • Updates after ftf. Added feature "Register recall", removed feature "service call" and redefined "Multi-terminal support" to reflect decisions on WUG status. Added description of Register recall to section 4.3. Removed emergency, service and ad-hoc call classes. Added description of piconet handling to 4.1.2. Updated and reworked SDP record. Additions to contributor list. Figure in section 8.2 removed. "Status" chapter removed. Added remark on security with respect to L2CAP connectionless. Editorial updates to section 4.4. • Updates to incorporate GAP and editorial guidelines for the specification • Errors in tables 3 and 4 and section 4.2. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:4
Intercom Profile

Rev	Date	Comments
1.0	June 20th 1999	• Update after F2F, incorporating technical issues only. • Editorial improvements. • Replaced bonding with authentication in Section 2.4. • Corrected references to LMP. • Removed PSM field from service record, and rephrased opening statement of SDP section. • Added chapter on GAP interoperability requirements. • Final GAP alignment. • Mandated call confirmation as SETUP confirmation. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:5
Serial Port Profile

Rev	Date	Comments
1.0	June 20th 1999	• Added more details on application layer procedures (chapter 3). First alignment with Generic Access Profile. • Added requirements on SDP procedures. More alignment with GAP. • Corrected some typos. Removed section 5.3.3 (Link Power Mode in L2CAP). Removed "Management entity" throughout document. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:6
Headset Profile

Rev	Date	Comments
1.0	June 20th 1999	• Update after F2F, incorporating outstanding issues as discussed (volume control and synchronisation, added AT command +VGS and +VGM, extended audio connection transfer description, authentication/ encryption optional to use, status change of outgoing audio connection, service record updated) and various editorial issues (amongst others update of contributors list). Removed status and history section. • Remote audio volume control: replaced may's with shall's to make it more consistent (if Remote audio volume control is supported, the entire procedure shall be supported as specified). • SDP - removed PSM for RFCOMM, added misplaced server channel. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:7
Dial-up Networking Profile

Rev	Date	Comments
1.0	June 20th 1999	• Some SDP values filled in, CoD updated after assigned numbers doc. • Updates after Tampere ftf: SDP record updated and reworked. Removed table from chapter 5.1 (now in RFCOMM). Updated contributors list. Figure removed from section 5.5.1. • Added profile structure section. Alignment with GAP (section 6) added. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:8
Fax Profile

Rev	Date	Comments
1.0	June 20th 1999	• Replaced bonding with authentication in Section 2.4. • Corrected references to LMP. • removed PSM field from service record, and rephrased opening statement of SDP section. • Added chapter on GAP interoperability requirements. • Updated Figure 1, Service discovery Profile to Service Discovery Application Profile. • Removed “ME” block from both sides of figure 2. • Removed paragraph discussing “ME” in section 2.1. • Renamed Section Heading 4 From Dialling and Control to Dialling and Control Interoperability requirements. • Re-worded section 4.1.2. • Removed the words “the” and “section” from the last sentence in section 5.3, paragraph 2. • Removed section 5.6. • Aligned section 6 with new changes from Dialup networking profile. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:9
LAN Access Profile

Rev	Date	Comments
1.0	June 20th 1999	• Updated Service records in line with “best practice”. • Removed Security section. • Editorial changes in Section 4.1. • Editorial changes in Section 5.1. • Editorial changes in Section 11.2. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view.

Part K:10
Generic Object Exchange Profile

Rev	Date	Comments
1.0	June 20th 1999	• Updated Chapter 1.2 and added reference to GAP regarding link and channel establishment. • Removed the security statement from the Profile fundamentals chapter, clarified the use of the limited discoverable mode in the Inquiry and Inquiry Scan chapter, and added the GAP requirement chapter. • Changed the 'initialization' wording to 'bonding', added some cross-references, and included the errata for IrOBEX in the reference list. • Management entity removed and the fall back procedure added if the Limited Inquiry procedure is supported. • Clarified that the fall back to the General inquiry is mandatory if Limited Inquiry is used. • Editorial changes and Chapter 7.3.1 (Bonding) updated to describe the result of Bonding. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view.

Part K:11
Object Push Profile

Rev	Date	Comments
1.0	June 20th 1999	• Removed PSM from SDP record. Updated text in Profile Structure 1.2. GAP alignment in Profile Fundamentals. • Editorial. • Renamed Initialization to Bonding. • Removed the ME section and references to ME. Stated in profile fundamentals that bonding is mandatory to support and optional to use. Removed "Notation for timers and counters". Changed wording in application procedure for object push feature. Minor update of SDP record. • Changed recommended inquiry procedure in chapter 3 to reference to the GOEP. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Part K:12
File Transfer Profile

Rev	Date	Comments
1.0	June 20th 1999	• SDP table changes, addition of references to doc [16]. • More SDP table changes, alignment with GAP, contributors update, and copyright notice. • Editorial and reference corrections. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view.

Part K:13
Synchronization Profile

Rev	Date	Comments
1.0	June 20th 1999	• Updated service records, IrMC chapter updated. • Chapter 1.2 updated, profile fundamentals clarified, recommended inquiry procedure added into Chapter 3 and service records updated. • Security issues clarified in Profile Fundamentals chapter and some editorial changes. • Change the 'Initialization' wording to 'Bonding', updated cross-references, and editorial changes. • Remove Management entity, removed statement that IrMC client must initiate the link establishment when bonding is not performed, and added a reference to the inquiry procedures of GOEP. • Editorial changes. • 1.0 Draft
1.0B	Dec 1st 1999	• Revised from a linguistic point of view. • Errata items previously published on the web has been included. These corrections and clarifications are marked with correction bars.

Appendix II

CONTRIBUTORS

CONTRIBUTORS

Part K:1
Generic Access Profile

Patric Lind (editor)	Ericsson Mobile Communications AB
Elg, Johannes	Ericsson Mobile Communications AB
Johan Sörensen	Ericsson Mobile Communications AB
Erik Slottboom	Ericsson Mobile Communications AB
Thomas Muller	Nokia Mobile Phones
Stephane Bouet	Nokia Mobile Phones
Martin Roter	Nokia Mobile Phones
Brian Redding	Motorola
Chatschik Bisdikian	IBM
Ken Morley	3COM
Jon Inouye	Intel

Part K:2
Service Discovery Application Profile

Elg, Johannes	Ericsson Mobile Communications AB
Bisdikian, Chatschik (editor)	IBM Corporation
Miller, Brent	IBM Corporation
Muller, Thomas	Nokia
Pascoe, Robert A.	XtraWorX

Part K:3
Cordless Telephony Profile

Olof Dellien (editor)	Ericsson Mobile Communications AB
Erik Slotboom	Ericsson Mobile Communications AB
Gert-jan van Lieshout	Ericsson Mobile Communications AB
Martin Roter	Nokia Mobile Phones
Christian Zechlin	Nokia Mobile Phones
Ken Morley	3COM
Richard Shaw	3COM
Brian Redding	Motorola
Alex Feinman	Motorola
Sridhar Rajagopal	Intel Corporation
Ramu Ramakesavan	Intel Corporation
Jun’ichi Yoshizawa	Toshiba

Part K:4
Intercom Profile

Erik Slotboom (editor)	Ericsson Mobile Communications AB
Olof Dellien	Ericsson Mobile Communications AB
Gert-jan van Lieshout	Ericsson Mobile Communications AB
Richard Shaw	3COM
Ken Morley	3COM
Shridar Rajagopal	Intel Corporation
Ramu Ramakeshavan	Intel Corporation
Brian Redding	Motorola
Thomas Müller	Nokia Mobile Phones
Christian Zechlin	Nokia Mobile Phones
Jun’ichi Yoshizawa	Toshiba

Part K:5
Serial Port Profile

Johan Sörensen (editor)	Ericsson Mobile Communications AB
Olof Dellien	Ericsson Mobile Communications AB
Riku Mettälä	Nokia Mobile Phones

Part K:6
Headset Profile

Richard Shaw	3Com
Ken Morley	3Com
Erik Slotboom (editor)	Ericsson Mobile Communications AB
Olof Dellien	Ericsson Mobile Communications AB
Bailey Cross	Intel Corporation
Shridar Rajagopal	Intel Corporation
Brian Redding	Motorola
Alex Feinman	Motorola
Thomas Müller	Nokia Mobile Phones
Christian Zechlin	Nokia Mobile Phones
Martin Roter	Nokia Mobile Phones
Jun'ichi Yoshizawa	Toshiba

Part K:7
Dial-up Networking Profile

Olof Dellien (editor)	Ericsson Mobile Communications AB
Erik Slotboom	Ericsson Mobile Communications AB
Riku Mettälä	Nokia Mobile Phones
Martin Roter	Nokia Mobile Phones
Christian Zechlin	Nokia Mobile Phones
Ken Morley	3COM
Richard Shaw	3COM
Brian Redding	Motorola
Alex Feinman	Motorola
Sridhar Rajagopal	Intel Corporation
Jun'ichi Yoshizawa	Toshiba

Part K:8
Fax Profile

Richard Shaw (editor)	3COM
Ken Morley	3COM
Olof Dellien	Ericsson Mobile Communications ABn
Erik Slotboom	Ericsson Mobile Communications AB
Gert-jan van Lieshout	Ericsson Mobile Communications AB
Riku Mettälä	Nokia Mobile Phones
Christian Zechlin	Nokia Mobile Phones
Thomas Müller	Nokia Mobile Phones
Martin Roter	Nokia Mobile Phones
Sridhar Rajagopal	Intel Corporation
Ramu Ramakeshavan	Intel Corporatio
Brian Redding	Motorola
Alex Feinman	Motorola
Jun'ichi Yoshizawa	Toshiba

Part K:9
LAN Access Profile

Jon Burgess	3COM
Paul J Moran (editor)	3COM
Phil Crooks	3COM
Johan Sorensen	Ericsson Mobile Communications AB
Chatschik Bisdikian	IBM Corporation
Marcia Peters	IBM Corporation
Pravin Bhagwat	IBM Corporation
Kris Fleming	Intel Corporation
Michael Camp	Nokia Mobile Phone
Shaun Astarabadi	Toshiba Corporation
Yosuke Tajika	Toshiba Corporation

Part K:10
Generic Object Exchange Profile

David Kammer	3COM
Patrik Olsson	Ericsson Mobile Communications AB
David Suvak	Extended Systems
Apratim Purakayastha	IBM Corporation
Aron Walker	IBM Corporation
Jon Inouye	Intel Corporation
Stephane Bouet	Nokia Mobile Phones
Riku Mettälä (editor)	Nokia Mobile Phone
James Scales	Nokia Mobile Phone
Steve Rybicki	Puma Technology
Shaun Astarabadi	Toshiba Corporation
Katsuhiro Kinoshita	Toshiba Corporation

Part K:11
Object Push Profile

David Kammer	3Com
Patrik Olsson (editor)	Ericsson Mobile Communications AB
David Suvak	Counterpoint Systems Foundry
Apratim Purakayastha	IBM Corporation
Aron Walker	IBM Corporation
Jon Inouye	Intel Corporation
Stephane Bouet	Nokia Mobile Phones
Riku Mettälä	Nokia Mobile Phone
James Scales	Nokia Mobile Phones
Steve Rybicki	Puma Technology
Shaun Astarabadi	Toshiba Corporation
Katsuhiro Kinoshita	Toshiba Corporation

Part K:12
File Transfer Profile

Shaun Astarabadi (editor)	Toshiba Corporation
David Suvak	Extended Systems
Patrik Olsson	Ericsson Mobile Communications AB
Apratim Purakayastha	IBM Corporation
Aron Walker	IBM Corporation
Jon Inouye	Intel Corporation
Stephane Bouet	Nokia Mobile Phones
Riku Mettälä	Nokia Mobile Phones
James Scales	Nokia Mobile Phones
Steve Rybicki	Puma Technology
Katsuhiro Kinoshita	Toshiba Corporation

Part K:13
Synchronization Profile

David Kammer	3COM
Patrik Olsson	Ericsson Mobile Communication AB
David Suvak	Extended Systems
Brent Miller	IBM Corporation
Apratim Purakayastha	IBM Corporation
Aron Walker	IBM Corporatio
Jon Inouye	Intel Corporation
Stephane Bouet	Nokia Mobile Phones
Riku Mettälä (editor)	Nokia Mobile Phone
James Scales	Nokia Mobile Phone
Steve Rybicki	Puma Technology
Shaun Astarabadi	Toshiba Corporation
Katsuhiro Kinoshita	Toshiba Corporation

The Bluetooth Specification was compiled and edited by Dan Sönnerstam, Pyramid Communication AB.

Appendix III

ACRONYMS AND ABBREVIATIONS

LIST OF ACRONYMS AND ABBREVIATIONS

Abbreviation or Acronym	Meaning
ACL	Asynchronous Connectionless
AG	Audio Gateway
AP	Access Point
B	
BB	Baseband
BD_ADDR	Bluetooth Device Address
C	
CC	Call Control
CL	Connectionless
CO	Connection-oriented
CoD	Class Of Device
CTP	Cordless Telephony Profile
D	
DAC	Device Access Code
DIAC	Dedicated Inquiry Access Code
DT	Data Terminal
DT	Data Terminal
F	
FHS	Frequency Hopping Synchronization
G	
GAP	Generic Access Profile
GIAC	General Inquiry Access Code
GM	Group Management
GOEP	Generic Object Exchange Profile
GW	Gateway
H	
HCI	Host Controller Interface
HS	Headset

I	
IP	Internet Protocol
IPX	Internet Protocol eXchange
IrDA	Infrared Data Association
IrMC	Ir Mobile Communications
L	
L2CA	Logical Link Control And Adaptation
L2CAP	Logical Link Control And Adaptation Protocol
LAN	Local Area Network
LAP	LAN Access Point
LC	Link Controller
LIAC	Limited Inquiry Access Code
LM	Link Manager
LMP	Link Manager Protocol
LocDev	Local Device
M	
ME	Management Entity
MM	Mobility Management
MSC	Message Sequence Chart
MTU	Maximum Transmission Unit
O	
OBEX	Object Exchange Protocol
P	
PC	Personal Computer
PDA	Personal Digital Assistant
PDU	Protocol Data Unit
PIM	Personal Information Management
PPP	Point-to-Point Protocol
PSTN	Public Switched Telephone Network
Q	
QoS	Quality Of Service

R	
RemDev	Remote Device
RFCOMM	Serial Cable Emulation Protocol
S	
SD	Service Discovery
SDDB	Service Discovery Database
SDP	Service Discovery Protocol
SeP	Serial Port
SIG	Special Interest Group
SrvDscApp	Service Discovery Application
T	
TCP	Transport Control Protocol
TCS	Telephony Control Specification
TL	Terminal
TL_O	Terminal Originating A Call
TL_T	Terminal Terminating A Call
U	
UDP	User Datagram Protocol
UI	User Interface
UIAC	Unlimited Inquiry Access Code
UUID	Universally Unique Identifier
W	
WUG	Wireless User Group

INDEX