17 December 2009

Status
Approved

Revision
V10r00

Document File Name
HDP Implementation
Guidance Whitepaper

Document Owner
Medical Devices WG

E-mail Address
med-feedback@bluetooth.org

HEALTH DEVICE PROFILE

Implementation Guidance Whitepaper

ABSTRACT: This whitepaper provides medical, healthcare and fitness device manufacturers an overview and guidance in implementing the *Bluetooth* Health Device Profile in their products.

Bluetooth SIG

Revision History		
Revision	**Date**	**Description**
V10r00	17 December 2009	Approved version.

Contributors	
Company	Name
A & D Medical	Jerry Wang
Broadcom	David Hughes Sen-Der Huang
CSR	Hermanni Suominen Nick Hunn
connectBlue	Mats Andersson
IBM	Michael Nidd
Intel Corp.	Robert D. Hughes (editor) Doug Bogia
Nokia	Paivi Ruuska
Nonin Medical, Inc.	Jayant Parthasarathy Josh Schilling Kurt Kermes
Kyocera	Teodor Dumitru
MindTree	Dennis Mathews Krishna Shingala
Roche	Joe Forler
STMicroelectronics	Andrea Cucchi
Samsung	Giriraj Goyal
Socket Mobile	Len Ott
Stollmann E+V GmbH	Karsten Aalders
Sybase / iAnywhere	David Suvak
Texas Instruments	Leonardo Estevez

DISCLAIMER AND COPYRIGHT NOTICE

THIS DOCUMENT IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Any liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Products equipped with the Bluetooth technology ("Bluetooth products") may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Bluetooth products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in this document provides any information or assistance in connection with securing such compliance, authorizations or licenses. NOTHING IN THIS DOCUMENT CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

This document is for comment only and is subject to change without notice.

Contents

1. Background

As countries, companies, and individuals around the world become aware of the growing percentage of the population needing medical care, both because of the changing demographics and the incidence of long-term chronic disease, they are looking for better ways to monitor health. This is reflected in a growing demand for connected health devices, where patient data can be collected, either by a medical institution, or by the patient themselves. As the use of these devices increases, along with the volume of data produced, it becomes increasingly important to ensure interoperability, so that similar devices connect and transfer data in a standard way.

Health device manufacturers already use Bluetooth wireless technology for a secure and reliable connection. Until now, Bluetooth technology provided the wireless link. The underlying data protocols and formats were proprietary. There was not even agreement over the best profile to base these on. Most used serial port profile (SPP) to emulate a standard RS-232 (EIA-232) serial cable, but DUN, FAX, PAN and HID have also been put to use. In order for a consumer mass market in health and fitness devices to evolve, manufacturers have realized that it requires them to adopt an interoperable wireless standard.

The Medical Devices Working Group (MED WG) formed to develop a profile that could provide this level of interoperability between health device Sources (such as blood pressure meters, weighing scales and thermometers) and health device Sinks (such as PCs, PDAs, mobile phones and displays) from different manufacturers. The work of the MED WG has resulted in the development of the Health Device Profile (HDP) [1] and the Multi-Channel Adaptation Protocol (MCAP) [2] which together fulfill this need.

The Bluetooth Health Device Profile defines the underlying wireless connection and protocol. It operates in conjunction with the ISO/IEEE 11073-20601 Personal Health Data Exchange Protocol and associated 11073-104xx device specialization specifications (where xx represents a specific document number) to provide application level interoperability for a wide variety of personal health devices.

The purpose of this Whitepaper is to explain how these fit together and provide examples and best practice regarding how they can be implemented. It also covers qualifications issues that must be observed for developers and manufacturers of medical and health devices. It is designed to be read alongside the relevant specifications, where it is hoped it will improve the understanding of the features of HDP and MCAP as designed by the MED WG.

Any feedback is appreciated and should be e-mailed to: med-feedback@bluetooth.org

2. Introduction to HDP

2.1 USE CASES

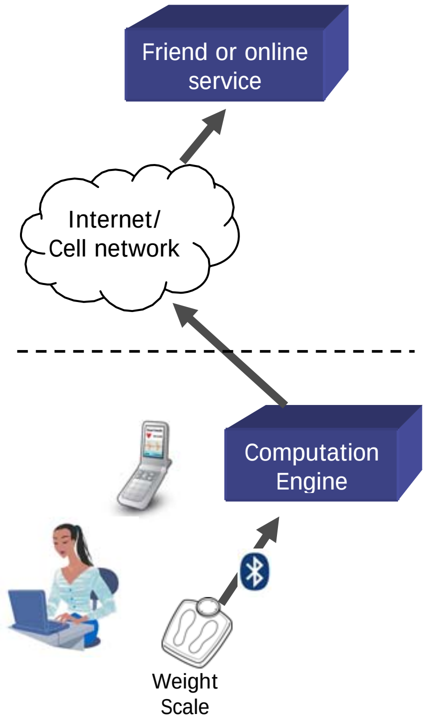
The purpose of HDP is to support a long and growing list of proposed applications for mobile, in-home, in-clinic and in-hospital uses. For example, in a clinical setting, Bluetooth transceivers using HDP may transmit waveforms from portable patient monitoring devices (Sources) such as ECG monitors and blood oximeters to fixed monitoring stations (Sinks) located within the clinic.

Exemplary home use applications include those in which the Sources might be pulse oximeters, glucose meters, weight scales, thermometers, or blood pressure monitors, and in which the Sink might be a cell phone, a PDA, telehealth station, or a personal computer. In such applications, the Sink might serve not only to display the data locally, but also to forward that data to a remote server for further sharing (as with the user's physician) and archiving.

The following examples highlight other use cases and possible applications for HDP.

2.1.1 UC1 - FITNESS/WELLNESS MANAGEMENT

A fitness enthusiast using wireless biosensors is monitoring progress to his/her workout and fitness goals.



2.1.1.1 ASSUMPTIONS

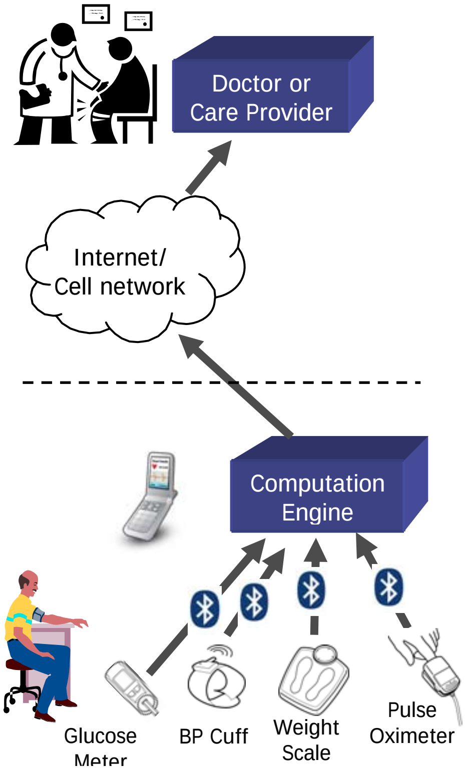
- Devices can be used at home or on the go.
- Devices are likely from multiple manufacturers.
- Multiple data types can be transmitted to the Computation Engine on multiple Bluetooth links in a piconet.

2.1.1.2 USAGE STEPS

1. A user (e.g. fitness enthusiast) uses a Bluetooth-enabled biosensor (weight scale, etc) to collect health data.
2. These devices transmit data at a scheduled time, when initiated by the user or immediately after use to a Bluetooth-enabled Computation Engine (mobile phone, PC, home health station, set-top box, PDA, etc.).
3. From the Computation Engine, the user can view the sensor data and thereby track their progress over time and customize workout goals
4. The user may optionally send their data over a secure network (cellular network, internet, Plain Old Telephone Service (POTS), etc.) such that a friend or fitness coach can acquire access via an online service.

2.1.2 UC2 - CHRONIC CONDITION MANAGEMENT / PATIENT RECOVERY

A patient with a chronic condition (short-term or long-term) and their caregiver monitor the status of the patient's health using wireless biosensors.



2.1.2.1 ASSUMPTIONS

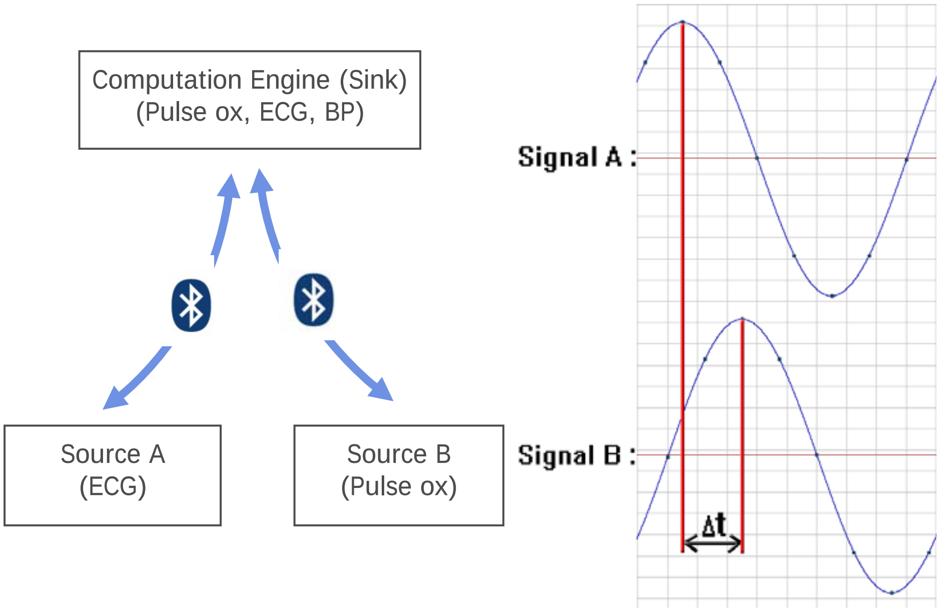
- Devices can be used at home or on the go.
- Devices are likely from the multiple manufacturers.
- Multiple data types can be transmitted to the Computation Engine on multiple Bluetooth links in a piconet.
- Source devices can send streaming data or event-driven data.

2.1.2.2 USAGE STEPS

1. A patient with a chronic condition (e.g. diabetes, congestive obstructive pulmonary disorder (COPD), congestive heart failure (CHF), asthma, etc,) or one who is recovering from a condition, uses one or more Bluetooth-enabled biosensors (blood pressure cuff, weight scale, glucose meter, thermometer, pulse oximeter, etc.) to collect vital sign data at various times during the day or week.
2. One or more devices transmit a combination of real-time streaming data and episodic data to a Bluetooth-enabled Computation Engine. This may occur at a scheduled time, as initiated by the user or immediately after use. The Bluetooth-enabled Computation Engine (mobile phone, PC, home health station, set-top box, PDA, telehealth device, etc.) then transmits the data within a secure network (cellular network, internet, POTS, etc.) to a central database where the patient's care provider acquires access.
3. The care provider receives this information and uses it to help the patient manage their condition by assisting them to live healthier or for a more rapid recovery while helping to lower the cost of healthcare. Both the management of and access to data allows for a better quality of life.

2.1.3 UC3 - PRECISE TIME SYNCHRONIZATION BETWEEN WIRELESS SENSORS

A patient with a long-term chronic condition wears a Bluetooth-enabled Computation Engine to monitor the status of their health with multiple Bluetooth enabled biosensors.



2.1.3.1 ASSUMPTIONS

- Source devices share Bluetooth clock information with the assistance of the Sink device.
- Both devices support the Clock Synchronization Protocol (CSP).
- To achieve the time synchronization, an application would require access to the Bluetooth clock of the Master and the clock offsets of the Slave devices in the piconet.

2.1.3.2 USAGE STEPS

1. A patient with a long-term chronic condition wears multiple Bluetooth enabled biosensors (pulse oximeter with a pulse rate monitor, ECG, etc.) to collect vital sign data continuously or periodically.
2. These Bluetooth-enabled sensors communicate to a common Computation Engine (e.g. embedded device, belt clip, mobile phone, display).
3. The various streams of data are time-synchronized to ~1 ms using the CSP. (The need for ~1 ms arises from the application that Pulse Transit Time and blood pressure can be calculated using the relative time difference between the SpO2 and ECG waveforms).
4. The Computation Engine then collects data from these biosensors and combines the precise time-stamped sensor data for analysis.

2.2 ARCHITECTURAL OVERVIEW

Figure 2.1 shows the protocols and entities used by the Health Device Profile.

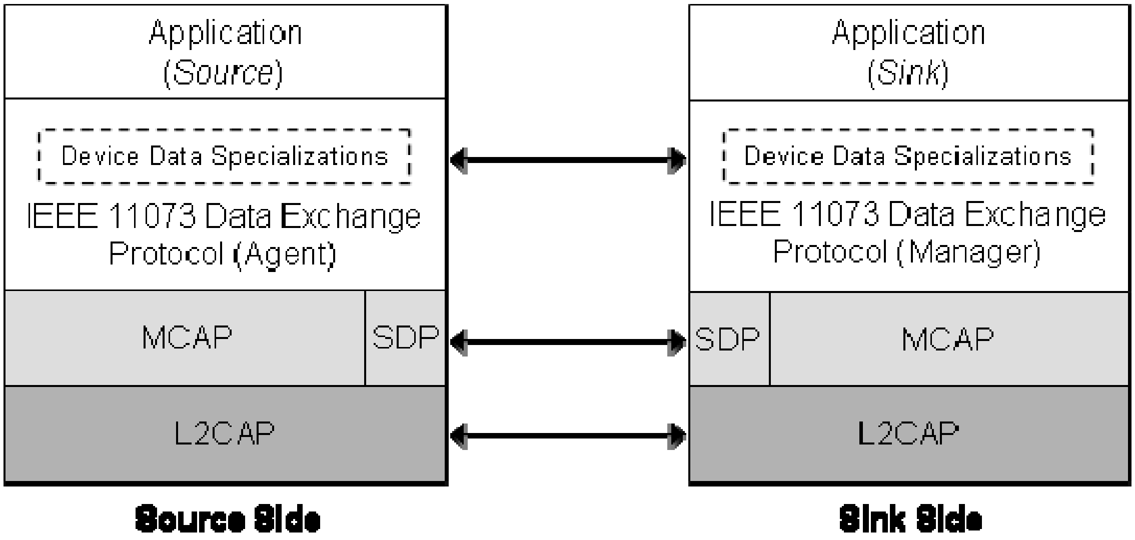


Figure 2.1: HDP Protocol Model

In Figure 2.1, *Source* refers to the source of data, while a *Sink* is the receiver of the data. The Source may generate data, or may relay data collected by one or more separate devices that wirelessly link the Sink to a device or entity outside of Bluetooth technology The Sink may be a display unit, or it may be any other consumer of data. It is possible for a device to operate as a Sink for one transaction, and a Source for another (an example is a dual-mode intermediary conduit where data is collected from a Source and provides store-and-forward to another Sink device).

Together, HDP and MCAP enable the establishment of one or more L2CAP Data Channels to support the exchange of device-specific data between a Sink and a Source. HDP uses the Multi-Channel Adaptation Protocol (MCAP) [2] for managing connection, disconnection and reconnection of Logical Link Control and Adaptation Protocol (L2CAP) [3] channels between these devices. MCAP in turn uses the L2CAP as the fundamental underlying Bluetooth protocol between devices. HDP adds further L2CAP channel configuration requirements by mandating

- Enhanced Retransmission Mode for any L2CAP channels needing to be ‘reliable’ and
- Streaming Mode for any L2CAP channels used for ‘streaming’ applications.

These new L2CAP modes are defined in Volume 3, Part A of Core Specification Addendum 1 and Volume 3 Part E in Bluetooth Core Specification 3.0 + HS [3].

To enable the data to be based on international standards, HDP relies upon the following external specifications:

- ISO/IEEE 11073-20601 Data Exchange Protocol [4] to define the data exchange protocol
- ISO/IEEE 11073-104xx Device Specializations [5] (where xx represents a specific document number corresponding to a particular device type such as weight scale, thermometer, glucose meter, blood pressure monitor or pulse oximeter) to define the specific descriptions for each data type.

See Section 3.3 for further information on Device Data Specializations. These specifications fully define the contents of each Data Channel (beyond the framing bytes) and are managed outside of the Bluetooth SIG by the IEEE 11073 Personal Health Device Working Group.

- Although HDP is designed to support the use of other data exchange protocols [1], IEEE 11073-20601 is the only data exchange protocol planned for use with HDP as of this writing.

As shown in Figure 2.1, HDP utilizes the Service Discovery Protocol (SDP) [3] for the discovery of services and their attributes. In addition, the specified Device Identification Profile [6] is required and is essentially a set of additional SDP records that further describes a device.

The modular protocol approach, with well-defined interfaces between modules, gives HDP device manufacturers enough flexibility to use different layers (of the protocol stack) that may come from different providers. This flexibility allows the manufacturer to concentrate on the application hardware and software.

2.3 TERMINOLOGIES

This section describes the terminologies used to describe the profile.

Initiator / Acceptor

The device that initiates the Control Channel connection becomes the “Initiator” of the connection. The device addressed in a Control Channel connection request is the “Acceptor” of the connection.

Control Channel / Data Channel

The first L2CAP channel established between two implementations of MCAP is the “Control Channel”. This channel facilitates the creation of “Data Channels,” through which actual health device data, which the Data Exchange Protocol and Device Specializations define, is exchanged. There is only a single instance of a Control channel per HDP instance and all MCAP commands are sent over this channel. Control Channels are always configured to be reliable.

Streaming and Reliable Data Channels

Two fundamental types of Data Channels exist in HDP: Reliable Data Channels and Streaming Data Channels. Reliable Data Channels are appropriate for transmitting measurement or alert information where the confidence in the exchange is at its highest (e.g. event driven or store and forward measurement). Streaming Data Channels are useful when the timeliness of the delivery of each frame is more important than the reliable delivery of every frame (e.g. an ECG waveform, where low latency is critical). Support for Reliable Data Channels is mandatory for all Sources and Sinks, while support for Streaming Data Channels is mandatory for all Sinks in order to support Sources that may request it.

MCAP Data End Point (MDEP)

An MCAP Data End Point (MDEP) represents one logical function, i.e. each MDEP is described by a set of parameters and includes the following: MDEP ID, MDEP Role (Source/Sink), MDEP Data Type (Device Specialization), and MDEP Description. Note that a device may have multiple functions - for example, it could measure both blood pressure and temperature, in which case it could host two different MDEPs. Each MDEP establishes the context of the device function.

MCAP Data Link (MDL)

An MCAP Data Link (MDL) identifies a pair of MDEPs, one each for Source and Sink, and its explicit creation occurs by one of the two participating devices as result of a request on the Control Channel.

MCAP Communications Link (MCL)

An MCAP Communications Link (MCL) refers to the collection of L2CAP connections between two instances of MCAP and is comprised of a Control Channel and zero or more Data Channels.

[1] Although technically supported by the design, protocols that overlap with IEEE 11073-20601 applications would not be permitted. This is to avoid market confusion and fragmentation.

MCAP Instance

An MCAP instance can have one or more MCLs. If there are two MCAP instances, then there will be two HDP instances with separate SDP records and at least two instances of the IEEE data layer. Refer to the example in Section 2.6.3.

2.4 ADVANTAGES OF HDP

This section explores the principal advantages provided by HDP (and MCAP). Since HDP requires the incorporation of MCAP, a reference to HDP will in general include MCAP throughout this document.

➢ **Medical, Healthcare and Fitness Applicability**

HDP is a specialized profile, designed specifically to allow for interoperability between medical, healthcare and fitness applications from different vendors. This gives HDP a significant advantage over more generic profiles like Serial Port Profile, or others that only provide a base layer for proprietary protocols and data formats.

➢ **Wireless Service Discovery**

HDP provides a standard wireless discovery method where a device’s device-type and supported application data-type is determined. This discovery occurs by using the Generic Access Profile (GAP) discovery procedures and the Service Discovery Protocol (SDP). The SDP record for HDP is standardized.

➢ **Reliable Connection-oriented behavior**

HDP uses the connection oriented capability of the Bluetooth lower layers to ensure more reliable behavior when a Source moves out of range or disconnects (either inadvertently or intentionally). This allows both Source and Sink to recognize that the link has been broken, and to take appropriate actions to reestablish the connection.

Additionally, the Reliable Data Channel deals with the detection and retransmission of packets corrupted by interference on the radio link. The Frame Check Sequence (FCS) is particularly important when operating with high interference levels such as being near Wi-Fi or other ISM band devices.

➢ **Robust Control Channel**

The HDP Control Channel requires Enhanced Retransmission Mode and FCS for improved reliability desired for robust operation.

➢ **Support for Flexible Data Channel configurations**

HDP Data Channels allow for independent configuration as this provides flexibility to the applications. Data Channels configured as ‘reliable’ use Enhanced Retransmission Mode, while data channels configured as ‘streaming’ use Streaming Mode. The use of FCS is optional for Data Channels.

➢ **Application-level Interoperability specific to health applications**

HDP, along with the ISO/IEEE 11073-20601 Data Exchange Protocol, provides a structured approach for the establishment of Data Channels to allow for information exchange between the communicating health devices. As device specializations are added in ISO/IEEE 11073-104xx and adopted by the MED WG, the Bluetooth Assigned Numbers allow for the addition of device data specializations without having to update the HDP specifications. Data traffic on Data Channels is completely defined by these IEEE specifications.

➢ **Efficient Reconnection mechanism**

HDP allows devices to retain the state of the system and eliminates redundant configuration steps upon reconnection. This procedure allows devices to disconnect while there is no receiving or transmitting of data, and then reconnect as data becomes available to be sent. This method reduces average power consumption.

➢ **Precise clock synchronization**

HDP also defines an optional Clock Synchronization Protocol (CSP) that allows for precise timing synchronization (theoretically in the microsecond range) between health devices. This feature is for health devices, such as high-speed sensors, that require close synchronization.

➢ **Optimized for devices with low resources**

HDP has a small set of simple control commands and makes it relatively inexpensive to implement. It is also possible for devices to support an even smaller subset of the available commands depending on the role of the device and individual application requirements. This is helpful for product requirements defining limited code and memory space.

2.5 SDP OVERVIEW

The Service Discovery Protocol (SDP) provides a means for applications to discover which services are available, and to determine their characteristics. SDP involves communication between an SDP Server and an SDP Client. Additional information about SDP and the SDP process is available in the Bluetooth whitepaper “Discovery Whitepaper: Service Discovery Applications” [14]. The SDP server maintains a list of service records that describe the characteristics of services associated with the server. The attributes of a service include the type or class of service offered and the mechanism or protocol information needed to utilize the service. Each service record contains information about a single service and each service can contain multiple features. An SDP Client can retrieve information from a service record maintained by the SDP Server by issuing an SDP request. If the client, or an application associated with the client, determines to use a service, then it opens a separate connection to the service provider in order to utilize the service.

HDP utilizes SDP to determine the appropriate L2CAP PSM values for the Control and Data Channel connections, as well as for the exchange of other identifying characteristics of the devices undergoing a connection. Refer to HDP and Bluetooth Assigned Numbers for specific service record attribute details related to HDP including: Service Class IDs, Protocol Descriptor, Profile Descriptor, Service Name, Service Description, Provider Name, Supported Features, etc. The HDP service discovery record also specifies the Data Exchange Protocol (e.g. IEEE 11073-20601 [4]) through the ’Data Exchange Specification’ field, and the Device Specialization (e.g. IEEE 11073-104xx [5]) through the ‘MDEP Data Type’ field of the SDP record.

Section 2.6 of this document and Appendix B of the HDP specification include useful SDP Record examples for HDP devices.

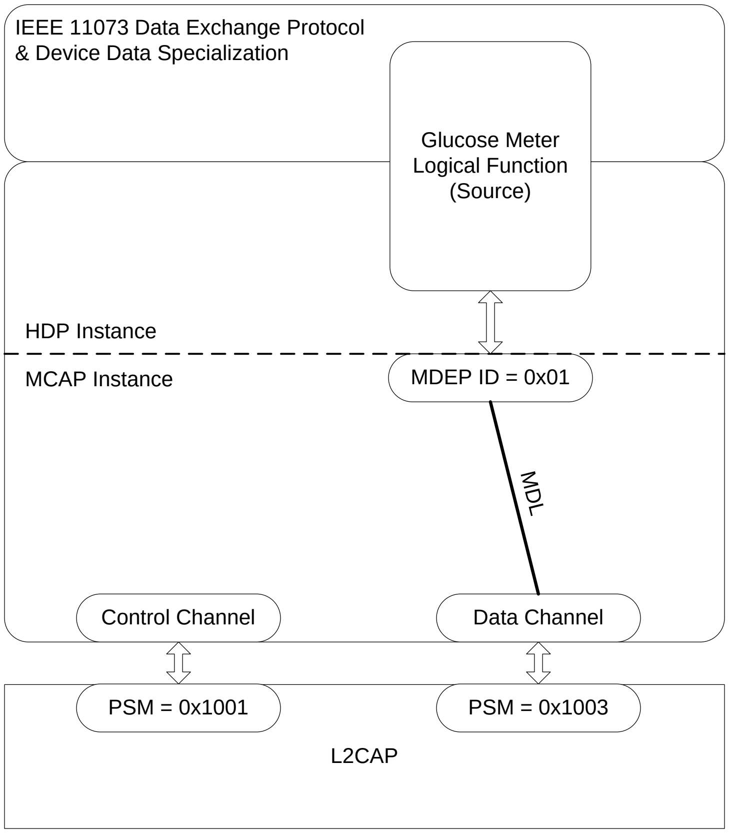
2.6 HDP/MCAP CONFIGURATION EXAMPLES

HDP with MCAP allows for a wide variety of implementation configurations that can support simple single-function single-role devices to complex multi-function multi-role devices. This includes implementations with multiple IEEE 11073 stacks and HDP implementations sharing single Bluetooth radio hardware for wireless communication.

This section is purely informative and does not represent all possible implementations. The examples in this section intend to show the relations among the following: Control Channel, Data Channel, MCAP Data Endpoint logical function (MDEP ID), MCAP Instance, HDP instance and the IEEE 11073 protocol implementations.

2.6.1 HDP/MCAP CONFIGURATION EXAMPLE 1

Figure 2.2 shows a glucose meter (Source) device, with a single logical function advertised by a single logical endpoint (MDEP ID = 0x01). Since HDP requires that at least one Reliable Data Channel exists, the MDL shown represents this channel type.



Single Function Device with a single logical Endpoint (MDEP ID)

Figure 2.2: HDP/MCAP Configuration - Example 1

The following table shows an example of the relevant portion of the HDP SDP record for the previous diagram.

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x09	Data elements sequence 9 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Glucose Meter (0x1011)	0x09 0x10 0x11	UINT 16
MDEP Role 0 (Source)	0x08 0x00	UINT 8

Table 2.1

2.6.2 HDP/MCAP CONFIGURATION EXAMPLE 2

Figure 2.3 shows multi-function Source device with two separate logical functions (i.e. a glucose meter Source and a blood pressure monitor Source). Both of these data types are supported over a single logical endpoint (MDEP ID 0x01).

When sharing MDEP IDs for multiple data types (as shown below), it is required to have a separate Supported Features List entry (also known as MDEP List) for each data type in the SDP record. All entries with the same MDEP ID are required to have the same MDEP Role. Multiple MDLs can share the same MDEP ID.

While it is optional to use separate MDEP IDs and Data Channels for each logical function, it is recommended that a single reliable Data Channel is created and a single association performed unless there is a specific reason why separate Data Channels are necessary. Some examples where these might need to be separate include the requirement to create a streaming Data Channel in addition to the reliable Data Channel or perhaps a second reliable Data Channel is needed due to different L2CAP configuration requirements from the first reliable Data Channel. Since MCAP treats the first reliable Data Channel differently than subsequent reliable Data Channels in some cases, the implementation becomes more complicated if there is a need to handle multiple reliable Data Channels. HDP does not propose a specific method for managing this and this is left to the implementer.

A single HDP/MCAP instance can support multiple device types. Similarly, a single instance of the 20601 data exchange protocol can support multiple device types.

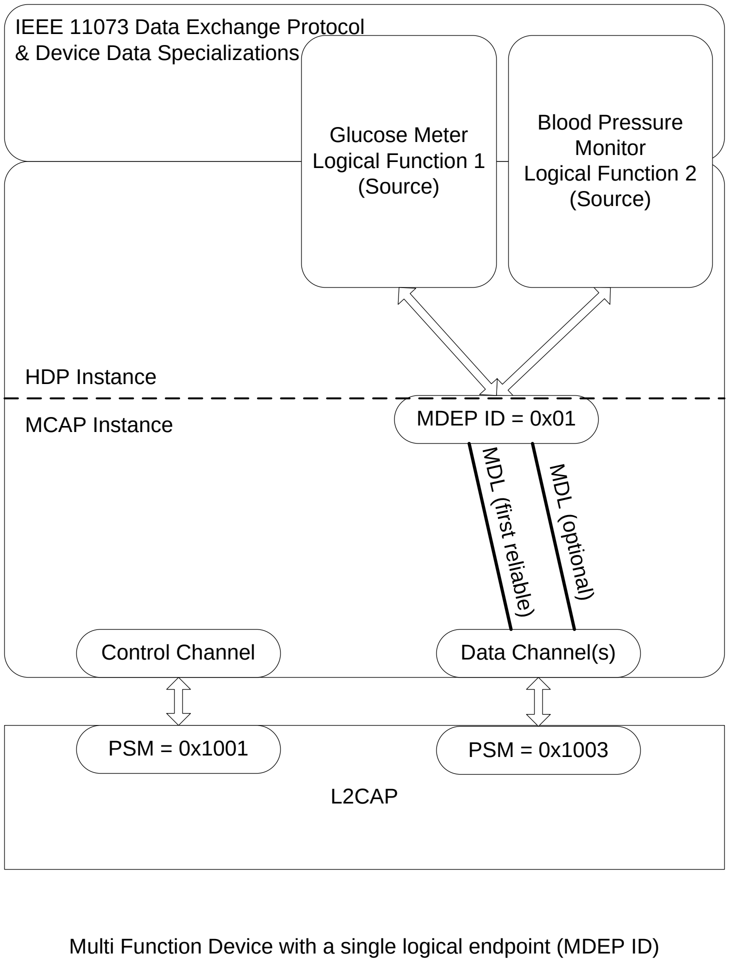


Figure 2.3: HDP/MCAP Configuration - Example 2

The following table shows an example of the relevant portion of the HDP SDP record for the previous diagram. This example shows two endpoints sharing a common MDEP ID. In this example, the optional MDL on the right hand side of the diagram is not used. Refer to Section 2.6.4 for an example of a device with multiple MDLs.

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x12	Data elements sequence 18 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Glucose Meter (0x1011)	0x09 0x10 0x11	UINT 16
MDEP Role 0 (Source)	0x08 0x00	UINT 8
Second endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Blood Pressure (0x1007)	0x09 0x10 0x07	UINT 16
MDEP Role 0 (Source)	0x08 0x00	UINT 8

Table 2.2

2.6.3 HDP/MCAP CONFIGURATION EXAMPLE 3

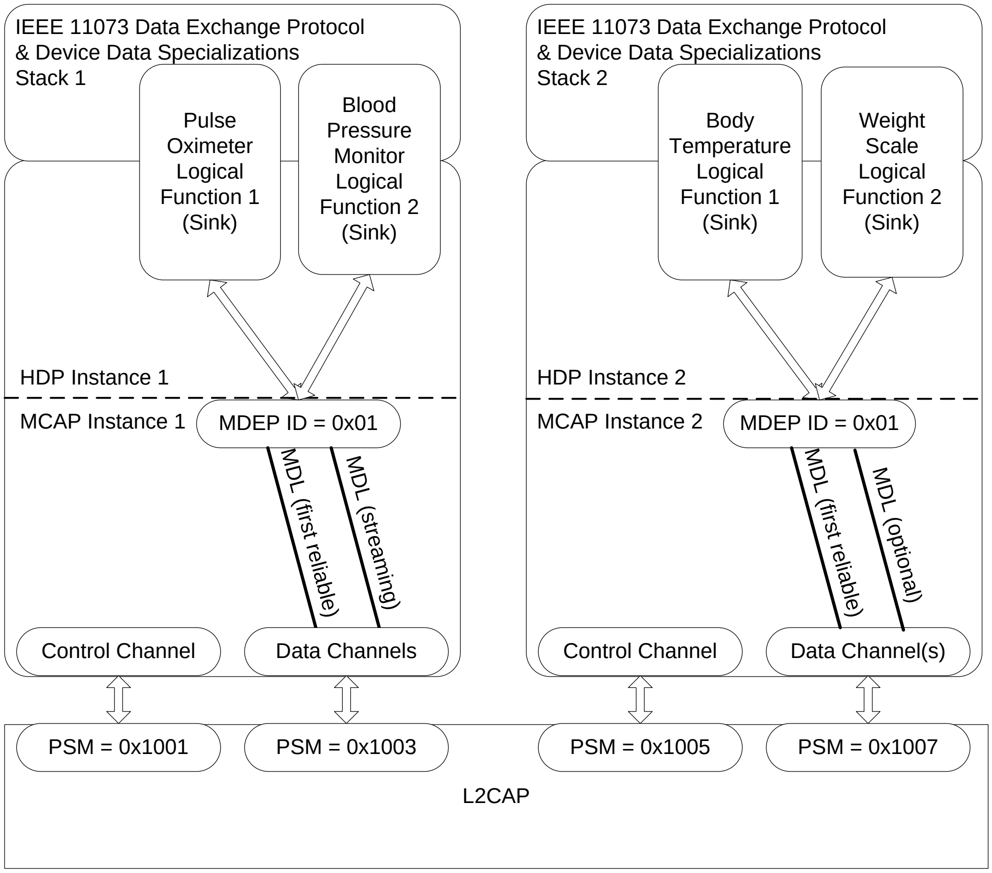
Figure 2.4 shows a complex multi-function Sink device with independent IEEE 11073-20601 Data Exchange Protocol stack instances.

Some implementations may have upper layer software to support different device types from different suppliers thus separate IEEE data layer and HDP/MCAP instances, but have a single Bluetooth radio shared by all logical functions. Each implementation instance of the 11073 stack has its own instance of HDP and each HDP instance has an instance of MCAP.

When sharing MDEP IDs for multiple data types (as shown below), it is required to have a separate Supported Features List entry (also known as MDEP List) for each data type in the SDP record. All entries with same MDEP ID are required to have the same MDEP Role. Multiple MDLs can share the same MDEP ID.

The left side of the diagram shows the use of two separate MDLs: one for the Streaming Data Channel used by the pulse oximeter, and one for the Reliable Data Channel that is shared between the pulse oximeter and the blood pressure meter.

The right side of the diagram shows two devices that typically both use Reliable Data Channels. In this case, the devices may share a common Reliable Data Channel (and MDL) for all data or may optionally have separate MDLs for their data, but share the first Reliable Data Channel for IEEE association and confirmed event traffic. For the latter case, the first Reliable Data Channel is used for all association and confirmed event traffic. If due to some error condition, the MDL with the first Reliable Data Channel is closed or disconnected, no further association and confirmed event traffic would be possible for either logical device and the remaining MDL would have to be disconnected and all MDLs reconnected.


IEEE 11073 Data Exchange Protocol & Device Data Specializations Stack 1
Pulse Oximeter Logical Function 1 (Sink)
Blood Pressure Monitor Logical Function 2 (Sink)
HDP Instance 1
MCAP Instance 1
MDEP ID = 0x01
MDL (first reliable)
MDL (streaming)
Control Channel
Data Channels
IEEE 11073 Data Exchange Protocol & Device Data Specializations Stack 2
Body Temperature Logical Function 1 (Sink)
Weight Scale Logical Function 2 (Sink)
HDP Instance 2
MCAP Instance 2
MDEP ID = 0x01
MDL (first reliable)
MDL (optional)
Control Channel
Data Channel(s)
PSM = 0x1001
PSM = 0x1003
PSM = 0x1005
PSM = 0x1007
L2CAP

Multi Function Device with two implementations sharing common Bluetooth hardware.

Figure 2.4: HDP/MCAP Configuration - Example 3

Table 2.3 shows an example of the relevant portion of the HDP SDP record for the previous diagram. In this example, the optional MDL on the right hand side of the diagram is not used.

HDP Instance 1:

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x12	Data elements sequence 18 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Pulse Oximeter (0x1004)	0x09 0x10 0x04	UINT 16
MDEP Role 1 (Sink)	0x08 0x01	UINT 8
Second endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Blood Pressure (0x1007)	0x09 0x10 0x07	UINT 16
MDEP Role 1 (Sink)	0x08 0x01	UINT 8

Table 2.3

HDP Instance 2:

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x12	Data elements sequence 18 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Body Temperature (0x1008)	0x09 0x10 0x08	UINT 16
MDEP Role 1 (Sink)	0x08 0x01	UINT 8
Second endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Weight Scale (0x100F)	0x09 0x10 0x0F	UINT 16
MDEP Role 1 (Sink)	0x08 0x01	UINT 8

Table 2.4

2.6.4 HDP/MCAP CONFIGURATION EXAMPLE 4

Figure 2.5 shows a single function Source device with multiple MDLs. HDP defines that when a device supports multiple Data Channels, the first Data Channel established shall be a Reliable Data Channel. This example focuses on the connection of additional required Data Channels that may be either Reliable or Streaming. A pulse oximeter is an example of a device that may require a Streaming Data Channel in addition to the required Reliable Data Channel. In this example, there are different cases depending on whether the pulse oximeter is an Initiator or Acceptor.

Pulse oximeter Source as Initiator:

When a pulse oximeter is a Source and acts as an Initiator, it will first request a Reliable Data Channel connection using MD_CREATE_REQ with Configuration “Reliable” using the pulse oximeter MDEP ID advertised in the SDP record of the Sink.

IEEE association and confirmed event traffic is exchanged on this Reliable Data Channel since it is the first (and in this example the only) Reliable Data Channel.

Additional MDLs may be created by either of the participating devices, that is, either by the Sink or Source. This typically occurs after association, but may also occur before association.

Case 1: Source initiates second Data Channel connection

> The pulse oximeter Agent (Source) requests a Streaming Data Channel using MD_CREATE_MDL with configuration “Streaming” on the same pulse oximeter MDEP ID of the Sink as used for the creation of Reliable Data Channel. Creation of the MDL by the Source is the most common case.
>
> The Sink then accepts the second MDL and the Streaming Data Channel is used to exchange IEEE unconfirmed operational data. Since Sinks are required to support Streaming Data Channels, the Sink accepts the Streaming Data Channel connection unless it has a serious resource constraint.
>
> If for any reason, a device cannot support an additional Data Channel connection (e.g. due to a serious resource constraint), it is required to reject the MD_CREATE_MDL_REQ. For this case, the IEEE data layer will decide if it is necessary to disconnect the MCL.

Case 2: Sink initiates second Data Channel connection

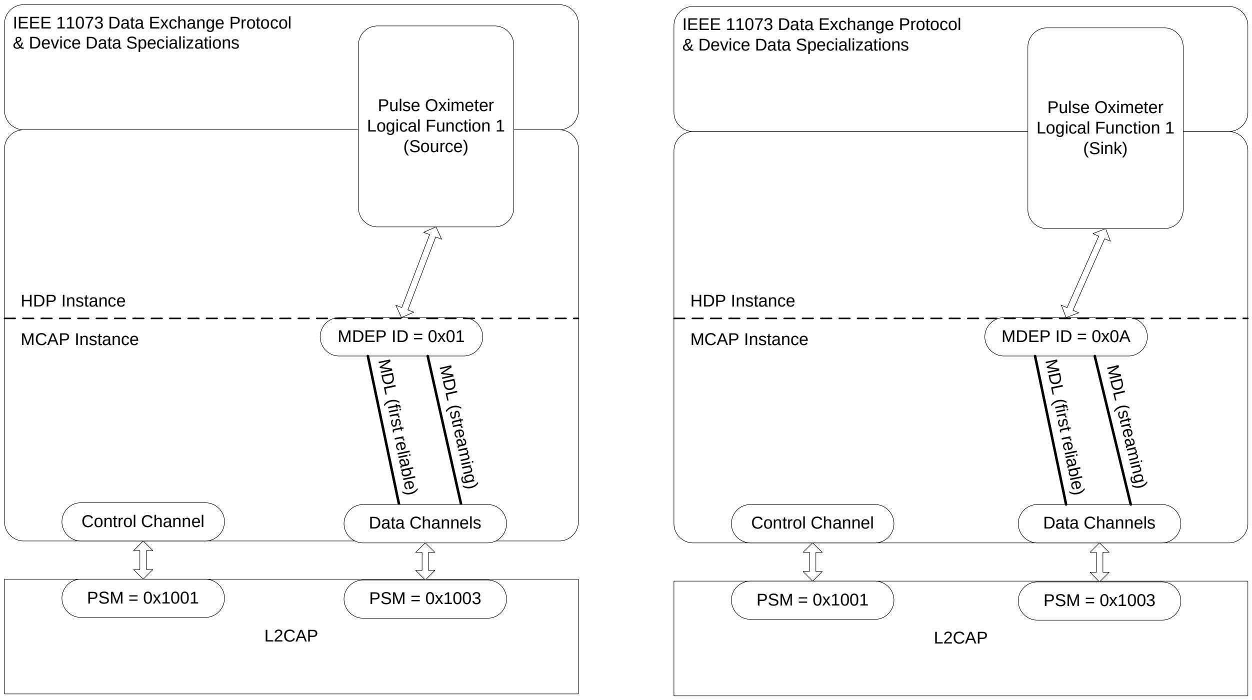
> The pulse oximeter Manager (Sink) wants to request the pulse oximeter Agent (Source) to start sending data of a session and desires the measurement data on a Streaming Channel. This case is rare but possible.
>
> The Sink requests a Data Channel creation using MD_CREATE_REQ with Configuration “No Preference” on the same pulse oximeter MDEP ID of the Source as used for the creation of Reliable Data Channel. The Source responds with the Configuration of “Streaming” and the connection continues as expected unless serious resource constraints exist as described in Case 1. For the case of success, IEEE unconfirmed operational data is exchanged on the Streaming Data Channel.

Pulse oximeter Source as Acceptor:

When a pulse oximeter Sink acts as an initiator, it initiates MD_CREATE_REQ with Configuration “No Preference” using the pulse oximeter MDEP ID advertised in the SDP record of the Source. The Source responds with Configuration “Reliable”. IEEE association and confirmed event traffic is exchanged on this Reliable Data Channel since it is the first (and only in the example) Reliable Data Channel.

Again, the additional MDL may be requested by either of the participating devices if necessary. The handling of the scenario will be same as described above in Case 1 and Case 2.

In all the above cases, a single MDEP definition is sufficient for the creation of an additional MDL and hence is no different from a pulse oximeter device requiring a single data channel.



Single Function Source Device with a Multiple MDLs

Single Function Sink Device with a Multiple MDLs

Figure 2.5: HDP/MCAP Configuration - Example 4

Table 2.5 shows an example of the relevant portion of the HDP SDP record for the case where the pulse oximeter is a Source:

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x09	Data elements sequence 9 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x01	0x08 0x01	UINT 8
MDEP Data Type Pulse oximeter (0x1004)	0x09 0x10 0x04	UINT 16
MDEP Role 0 (Source)	0x08 0x00	UINT 8

Table 2.5

Table 2.6 shows an example of the relevant portion of the HDP SDP record for the case where the pulse oximeter is a Sink:

Description	Value	Type
HDP Supported Features Attrib ID 0x200	0x09 0x02 0x00	UINT 16
Data Element Sequence	0x35 0x09	Data elements sequence 9 bytes
First endpoint	0x35 0x07	Data Element Sequence 7 bytes
MDEP ID 0x0A	0x08 0x0A	UINT 8
MDEP Data Type Pulse oximeter (0x1004)	0x09 0x10 0x04	UINT 16
MDEP Role 1 (Sink)	0x08 0x01	UINT 8

Table 2.6

2.7 L2CAP CONFIGURATION EXAMPLES

The HDP specification provides significant flexibility for the use of L2CAP configuration using new features defined in versions of L2CAP after 2.1 (Volume 3, Part A of Core Specification Addendum 1 and also Bluetooth Core specification 3.0 + HS). This enables the possibility to use an application-specific tuned configuration, but as a side effect, care must be taken to maintain interoperability. A common mistake is for a Source or Sink to try to enforce that the peer use same values as the local device and that can lead to interoperability issues. A good design principle is to be liberal in what will be accepted from the remote side when possible.

The following section includes some clarifications and guidance for the L2CAP configuration procedure. The guidance differs between Control Channels, Reliable Data Channels and Streaming Data Channels.

A better understanding of HDP and the new features of L2CAP will reduce the possibility that implementations may misunderstand these newer features possibly leading to interoperability problems.

2.7.1 TERMINOLOGY AND GENERAL GUIDANCE:

Mode of Operation:

- In HDP, possible Modes of Operation for L2CAP channels are only Enhanced Retransmission Mode (ERTM) and Streaming Mode (SM). HDP defines the appropriate mode depending upon the channel type being established. Devices will negotiate or reject a connection if the Mode of Operation is not supported by an implementation or is not compliant with HDP requirements.

TxWindow:

- TxWindow is used by a device sending a configuration request to indicate the maximum number of I-frames it can receive without an ACK and is an indication of the amount of receive buffers it has.
- TxWindow is also used by a device returning a configuration response to indicate the maximum number of I-frames it can send. With this information, devices can avoid exceeding the limitations of the other device and may use this information to optimize transmissions. This parameter is not negotiated and therefore one device shall not reject the connection of another device based on this parameter.

MaxTransmit:

- MaxTransmit controls the number of retransmissions that L2CAP is allowed to try in Enhanced Retransmission mode before accepting that a packet and the channel is lost. When a packet is lost after being transmitted MaxTransmit times the channel shall be disconnected by sending a Disconnect request. In general, this value should be set to a large enough value to avoid premature disconnects due to transient radio interference, but short enough to determine a link is not making progress in a reasonable amount of time. This parameter is not negotiated and therefore the device receiving the configuration request shall not reject the connection based on this parameter unless the value does not meet the requirements defined in the Core Specification. MaxTransmit values in a configuration response shall be ignored.

MTU:

- The Maximum Transmit Unit (MTU) parameter is used by a device sending a configuration request to indicate the maximum size SDU that it can receive and is an indication of the amount of buffer space it has.
- MTU is also used by a device returning a configuration response to indicate the maximum size SDU that it will send. With this information, devices can avoid exceeding the limitations of the other device and may use this information to optimize transmissions. This parameter is not negotiated and therefore one device shall not reject the connection of another device based on this parameter unless 1) the value does not meet the requirements defined in the Core Specification or 2) the responding device indicates that it will send an MTU larger than the requesting device indicated it can support.

FCS Option:

- The Frame Check Sequence (FCS) Option allows different FCS types to be requested on the L2CAP channel. Today the only allowable FCS types defined are 16-bit FCS or No FCS. The preferred type is communicated through configuration requests from each side. Since 16-bit FCS is the default, and there is only one other option defined, unless both sides indicate a desire to use No FCS, it is mandatory for 16-bit FCS to be used.
- For example, if both devices specify in a configuration request an FCS Option of “No FCS”, the channel shall bypass FCS checking. However, if at least one device specifies “16-bit FCS” in a configuration request or does not send the FCS Option at all, the channel shall use 16-bit FCS. This parameter is not negotiated and therefore one device shall not reject the connection of another device based on this parameter unless the value does not meet the requirements defined in the HDP specification or Core Specification.

2.7.2 RECOMMENDED AND REQUIRED SETTINGS

Other than the general guidance in Section 2.7.1, this section summarizes HDP and L2CAP requirements and provides further guidance that sometimes differs between channel types.

Recommended and Required Control Channel L2CAP settings:

- Enhanced Retransmission Mode is required by HDP for this channel type.

- Typically, a TxWindow value of 1 is recommended for requesting and responding devices since MCAP is a command/response protocol. For implementations that use MCAP standard op codes simultaneous with CSP, a TxWindow of 1 may not be sufficient as described below.

  Based on the CSP synchronization role(s) supported by the implementation, the exact size of TxWindow can be derived based on number of simultaneous packets that can be received without acknowledging. For example, if a device is acting as a Sync-Master and requests timestamp update indications, a TxWindow size of 3 is necessary as it shall be capable of receiving a standard command, a clock synchronization response and an update indication without acknowledgements. A Sync-Slave will not be able to transmit all three of these simultaneously if the Sync-Master is not capable of receiving them. On the other hand, the Sync-Slave implementation needs a TxWindow size of 2 to acknowledge a standard command request/response and a clock synchronization request simultaneously.

  A device supporting both roles simultaneously on the same connection requires a TxWindow size of 4 to be able to receive a clock synchronization response and an update indication as Sync-Master, to acknowledge a clock synchronization request as a Sync-Slave and to process standard commands.
- A MaxTransmit value of 10 is generally a good value in most cases.
- An MTU value of 48 bytes is recommended by HDP for requesting and responding devices since the largest Control Channel message will always be smaller than this value.
- FCS is required by HDP for this channel type.

Recommended and Required Reliable Data Channel L2CAP settings:

- Enhanced Retransmission Mode is required by HDP for this channel type.
- No specific TxWindow value is recommended for requesting or responding devices other than the general guidance provided in Section 2.7.1.
- No specific MaxTransmit value is recommended for requesting devices other than the general guidance provided in Section 2.7.1.
- SAR is required by HDP to be supported by senders of data. SAR is required by the Core Specification to be supported by receivers of data.
- An FCS Option value of “No FCS” is recommended for Sinks to allow Sources the option of either bypassing FCS checking or requiring FCS.

Recommended and Required Streaming Data Channel L2CAP settings:

- Streaming Mode is required by HDP for this channel type.
- TxWindow is not a parameter that is used with Streaming Mode and shall be set to zero in its request and ignored by the receiving device.
- MaxTransmit is not a parameter that is used with Streaming Mode and shall be set to zero in its request and ignored by the receiving device.
- SAR is required by HDP to be supported by senders of data. SAR is required by the Core Specification to be supported by receivers of data.
- An FCS Option value of “No FCS” is recommended for Sinks to allow Sources the option of either bypassing FCS checking or requiring FCS.

3. IEEE 11073-20601 Data Exchange Protocol Overview

The ISO/IEEE 11073-20601 Data Exchange Protocol (known as 20601) [4] provides a framework of object-oriented information modeling, information access and measurement data transfer suitable to a wide variety of personal health devices. Examples of such health devices are as follows: weighing scales, thermometers, pulse oximeters, blood pressure monitors, and glucose meters. In addition to health and fitness sensors, the protocol is designed to support a range of home health sensors. This enables interoperability between a data management device to process, display or transfer the specific measurements. Since the design of the 20601 protocol is for use with transports such as Bluetooth wireless technology and USB, integration with HDP is not a cumbersome process.

The ISO/IEEE 11073 specifications contain the 20601 core protocol specification, which describe the tools to represent and convey data, and 104xx Device Data Specialization specifications [5], which provide details on how the 20601 tools are applied for each health device's implementation.

3.1 TRANSFER OF MEASUREMENT DATA OVER HDP AND 20601

Before describing the steps involved in conveying measurement data, the terminology used requires some definition. HDP defines a *Source* to be a transmitter of application data, and 20601 uses the term *Agent* for the node that transmits personal health data. Similarly, HDP's notion of a *Sink* is essentially that of the 20601 *Manager* (i.e. a node receiving data from one or more Agents). Discussion of these pairs of terms occurs in their respective contexts where Source and Sink describe HDP process steps, and Agent and Manager for discussing the 20601 processes.

Figure 3.1 and Figure 3.2 illustrate the transactions emphasizing the 20601 data transfer services. Figure 3.1 shows the transactions for a device using a single Reliable Data Channel and Figure 3.2 shows the transactions for a device using Reliable and Streaming Data Channels. Either a Source or a Sink may initiate a connection by means of establishing a Control Channel. After the Control Channel is established, MCAP commands are used to establish one or more Data Channels. Reliable Data Channels are appropriate for transmitting measurement or alert information where the confidence in the robustness of the exchange needs to be at its highest (e.g. store and forward measurements). Streaming Data Channels are useful when the timeliness is a higher priority than the reliable delivery of every frame (e.g. waveform data, where the occasional loss of a small amount of data may be tolerable).

In most implementations, once the MCAP Data Link (MDL) has been established and the first Reliable Data Channel created, a connection indication is sent to the 11073-20601 application layer. This initiates a series of transactions defined by connection state machines within the Agent and Manager. The first step involves the Agent sending an Association Request message, which includes a high-level description of itself to the Manager. Only Agents are allowed to request Association.

If the Manager does not wish to communicate with the Agent, it will reply with an Association Response message with a rejection status code. On the other hand, if the Manager deems it appropriate to continue the Association, it will respond in one of two ways. 1) If the Manager has previously communicated with this device, or it has been programmed to understand the collection of objects, attributes and data transmission details (e.g. "*standard configuration*"), then the Manager will respond by accepting the association, at which point the Agent and Manager are considered to be in the Operating state. 2) If the Manager is not familiar with enough details of the Agent's implementation, then the Manager will accept the association, but will ask that the Agent describe its implementation by means of a Configuration process.

If multiple reliable Data Channels exist, association traffic and confirmed event traffic is to be sent on the first Reliable Data Channel (per MCL) and each MCL must have its own "first Reliable Data Channel" for such traffic. When multiple device specializations are supported by a device, it is recommended to associate to them at the same time as opposed to sequentially as if they are physically separate devices.

The Operating state is where measurement data transfer may occur. This transfer occurs through a number of methods, each of which may be suitable for a different style of communication.

Either the Manager or Agent may initiate data transfer. If the Manager initiates data transfer, then it may do so by (1) asking the Agent for a single measurement, if available, or (2) telling the Agent that it may transfer data for a fixed period of time, or (3) controlling the Agent's transmission of data using explicit Start and Stop commands.

When either the Agent or the Manager terminates an association, it may do so by issuing an Association Release Request. The device on the other end of the link responds with an Association Release Response. At this point, the 20601 layer informs HDP to disconnect the communications link shown by the disconnect indication. Another means of terminating an association, not shown here, is by means of an Association Abort. There is no required response from such a transmission.

A subsequent sequence, shown in the lower portion of the following two diagrams, describes the skipping of the Configuration steps as the Manager has stored the previous configuration information.

The 20601 protocol intends for asymmetric device architectures. Agents, by design, intend to measure health data. Their function is not to be a data processor, network device or timekeeper. Computation complexity transfers to the Manager whenever possible. This allows Agents to be small, inexpensive and simple.

Agents do not communicate with each other, but rather each Agent communicates with a single Manager at any point in time. The Manager communicates with multiple Agents and coordinates activities as in cases where the Manager coordinates precise measurement of physiological phenomena detected by distinct Agents.

The 20601 protocol provides a number of facilities to transfer a combination of discrete numeric data, sampled waveform data as well as the ability for the Agent to store data that had been collected over a long period and forward that information when requested by the Manager.

A Scanner data construct is able to collect discrete Numeric and waveform data objects and combine them into a single packet for transmission. This allows efficient transmission of Streaming data, since the overhead of support information accompanying multiple data objects reduces by only sending the support information of a single packet.

In several use cases, such as sleep studies, it may be more practical to defer transmitting measurements for several hours, or until several sessions of data have been collected. In order to accommodate the needs of such use cases, 20601 defines models to store, describe the structure of, and transfer data.

HDP and 20601 can support Agents providing multiple types of measurements (i.e. pulse oximeters and blood pressure monitors), because constructs within the data exchange layer allow the description of multiple Device Specializations. Since HDP provides for the establishment of multiple Data Channels, it is possible to transfer each function’s measurements across a dedicated Data Channel. It would also be possible to partition the channels so that the Agent could transfer the discrete numeric data from all functional units across a Reliable Data Channel, and transfer all waveform data across the Streaming Data Channels. Figure 3.1 illustrates transactions for a device that transfers discrete numeric data and Figure 3.2 illustrates transactions for a device that transfers streaming data. The blue text in Figure 3.2 shows sequences that differ from those in the reliable data example of Figure 3.1.

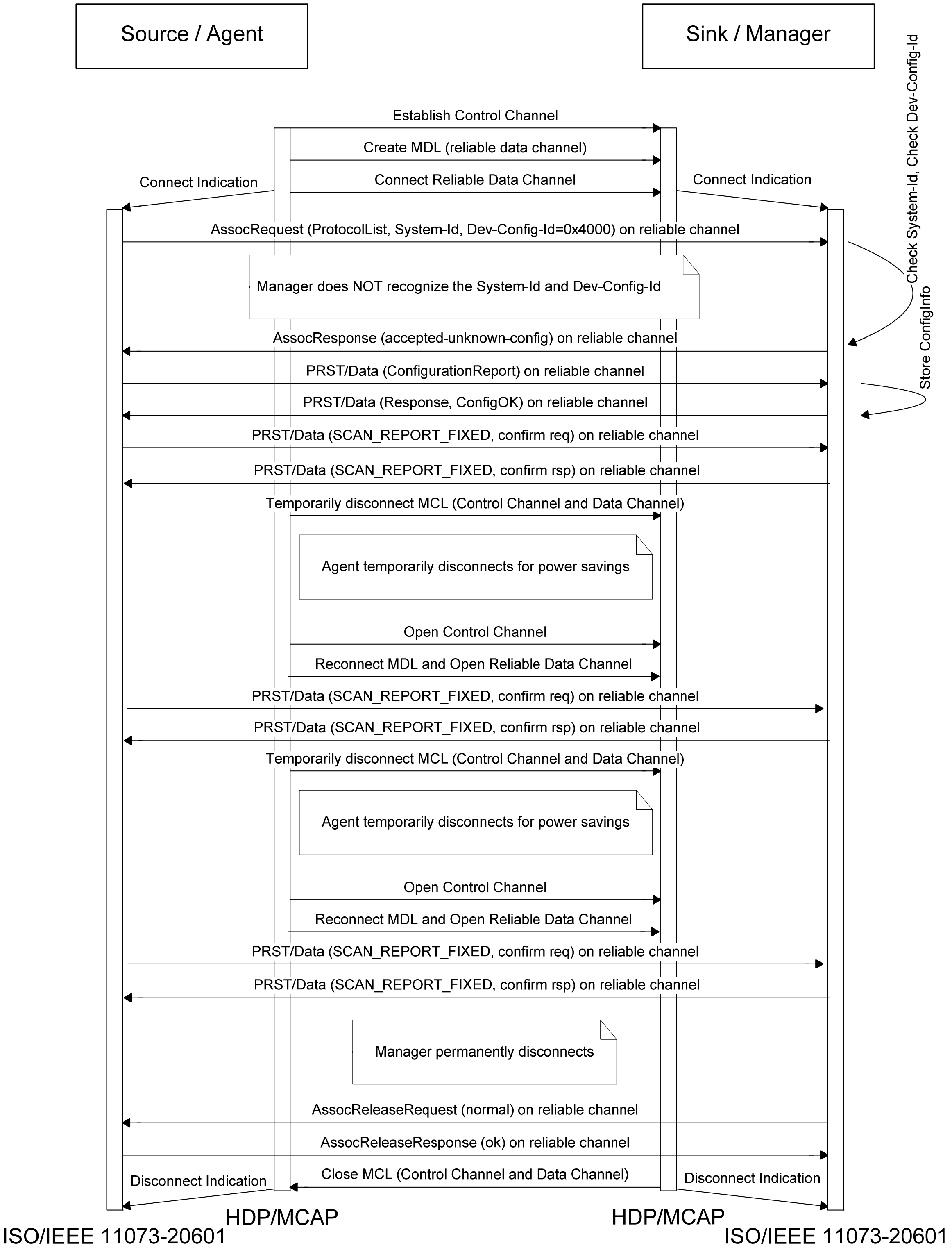

Source / Agent
Sink / Manager
Establish Control Channel
Create MDL (reliable data channel)
Connect Indication
Connect Reliable Data Channel
Connect Indication
AssocRequest (ProtocolList, System-Id, Dev-Config-Id=0x4000) on reliable channel
Check System-Id, Check Dev-Config-Id
Manager does NOT recognize the System-Id and Dev-Config-Id
AssocResponse (accepted-unknown-config) on reliable channel
Store ConfigInfo
PRST/Data (ConfigurationReport) on reliable channel
PRST/Data (Response, ConfigOK) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm req) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm rsp) on reliable channel
Temporarily disconnect MCL (Control Channel and Data Channel)
Agent temporarily disconnects for power savings
Open Control Channel
Reconnect MDL and Open Reliable Data Channel
PRST/Data (SCAN_REPORT_FIXED, confirm req) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm rsp) on reliable channel
Temporarily disconnect MCL (Control Channel and Data Channel)
Agent temporarily disconnects for power savings
Open Control Channel
Reconnect MDL and Open Reliable Data Channel
PRST/Data (SCAN_REPORT_FIXED, confirm req) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm rsp) on reliable channel
Manager permanently disconnects
AssocReleaseRequest (normal) on reliable channel
AssocReleaseResponse (ok) on reliable channel
Disconnect Indication
Close MCL (Control Channel and Data Channel)
Disconnect Indication
HDP/MCAP
HDP/MCAP
ISO/IEEE 11073-20601
ISO/IEEE 11073-20601

Figure 3.1: Source-initiated transactions for reliable HDP device

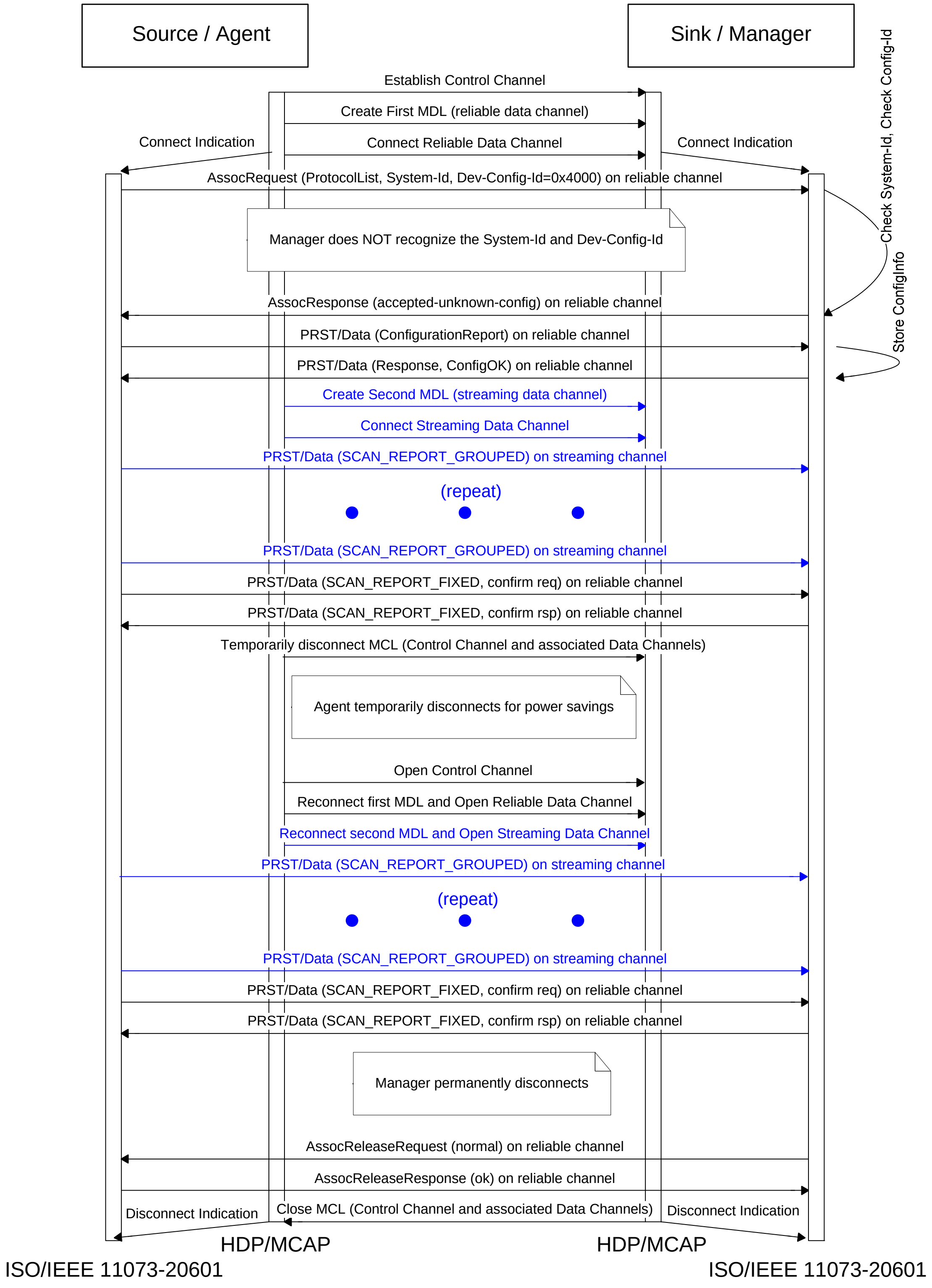

Source / Agent
Sink / Manager
Establish Control Channel
Create First MDL (reliable data channel)
Connect Indication
Connect Reliable Data Channel
Connect Indication
AssocRequest (ProtocolList, System-Id, Dev-Config-Id=0x4000) on reliable channel
Check System-Id, Check Config-Id
Manager does NOT recognize the System-Id and Dev-Config-Id
AssocResponse (accepted-unknown-config) on reliable channel
PRST/Data (ConfigurationReport) on reliable channel
Store ConfigInfo
PRST/Data (Response, ConfigOK) on reliable channel
Create Second MDL (streaming data channel)
Connect Streaming Data Channel
PRST/Data (SCAN_REPORT_GROUPED) on streaming channel
(repeat)
PRST/Data (SCAN_REPORT_GROUPED) on streaming channel
PRST/Data (SCAN_REPORT_FIXED, confirm req) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm rsp) on reliable channel
Temporarily disconnect MCL (Control Channel and associated Data Channels)
Agent temporarily disconnects for power savings
Open Control Channel
Reconnect first MDL and Open Reliable Data Channel
Reconnect second MDL and Open Streaming Data Channel
PRST/Data (SCAN_REPORT_GROUPED) on streaming channel
(repeat)
PRST/Data (SCAN_REPORT_GROUPED) on streaming channel
PRST/Data (SCAN_REPORT_FIXED, confirm req) on reliable channel
PRST/Data (SCAN_REPORT_FIXED, confirm rsp) on reliable channel
Manager permanently disconnects
AssocReleaseRequest (normal) on reliable channel
AssocReleaseResponse (ok) on reliable channel
Disconnect Indication
Close MCL (Control Channel and associated Data Channels)
Disconnect Indication
HDP/MCAP
HDP/MCAP
ISO/IEEE 11073-20601
ISO/IEEE 11073-20601

Figure 3.2: Source-initiated transactions for streaming HDP device

The remote device will usually detect whether a device has become intentionally temporarily disconnected or intentionally permanently disconnected, and can usually detect that the device has become unintentionally disconnected. An intentional temporary disconnect will be known by the remote device when a disconnect request has been received and processed. When this occurs, it is recommended that the remote device wait for future data from the local device by remaining in Operating state. An intentional permanent disconnect will be known by the remote device when a Delete MDL request has been received and processed. When this occurs, both sides disassociate and a disconnect indication is sent to the data layer. An unintentional disconnect is known to the remote device when an ACL link loss is detected without having received a disconnect request. When this occurs, it is recommended that the remote side wait for future data from the local device by remaining in Operating state as long as is reasonable. Additional notifications such as error codes will be up to the implementation. In case of unintentional disconnection, if detection of disconnection is later than the IEEE 11073-20601 defined timeout (typically 3 seconds), then implementations cannot benefit from retaining the states, as the data layer states will be reset in any case. Therefore, configuration of Link Supervision Timeout for the underlying ACL link should be between 1-2 seconds, so that the detection of unintentional disconnection happens before the IEEE 20601 defined timeout occurs. The exact communication API between HDP and the Data Exchange Protocol components is undefined. Bluetooth hardware can detect when it has been dropped from a piconet for some reason (additionally, a master can detect that a slave has dropped out). Temporary disconnection is further described in Section 3.1.1.

3.1.1 TEMPORARY DISCONNECT/RECONNECT AND RELATION TO IEEE DATA LAYER

The purpose of this subsection is to clarify the HDP Reconnect feature (used to restore temporary disconnection) and its relation to the IEEE 11073-20601 data layer. The main purpose of the HDP Reconnect feature is to allow HDP devices to disconnect quickly and efficiently to conserve radio power and later reconnect quickly when new data is available to transmit. This feature’s design is to be efficient for cases of temporary disconnection by allowing unneeded IEEE association and configuration steps to be skipped as well as unneeded HDP MDL operations.

The 20601 protocol is a connection-oriented data exchange protocol and in the normal case, a permanent disconnection of the transport (Bluetooth wireless technology in our case) would require transport disconnect notification to this data layer. Disconnecting Bluetooth channels temporarily is a way to "pause" the IEEE data layer connection so that the Bluetooth radio can be turned off to save power. Reconnecting Bluetooth channels is a way to "resume" the IEEE data layer connection once the Bluetooth radio is turned back on. It is not intended that a transport disconnect indication is sent to the data layer during these periods of temporary disconnection. This is illustrated in Figure 3.1 and Figure 3.2.

Before “pausing” or temporarily disconnecting at the HDP layer, the IEEE data layer would typically have gone through an association phase and, if necessary, the configuration phase thus ending up in the 20601 Operating state. Once in this state, the HDP implementation may choose to temporarily disconnect or permanently disconnect. For cases of temporary disconnection, a transport disconnect indication is not indicated to the IEEE layer. A transport disconnect indication is only used for cases where 1) an MDL disconnection is successful (permanent disconnect case using MDL Delete) 2) a Reconnect fails or 3) an MDL is created while that same MDL already exists, i.e. when overwriting an existing MDL. As there is no transport disconnect/reconnect indication sent for the temporary disconnect case, this HDP operation is transparent to the data layer and allows the skipping of the IEEE association and configuration phase. This is because the data layer was not aware that the Bluetooth link dropped.

Reconnecting without re-association requires both sides to keep their IEEE data layer in exactly the same state as they did just before the disconnection. If anything changed at the layers above HDP on either side, then the reconnection will fail and a full re-association and possibly re-configuration (depending upon what has changed) will occur. If the data layer is unassociated before disconnection, then the Reconnect feature does not save steps from an IEEE perspective since the full association and, if necessary, configuration steps would have to be repeated upon reconnection.

The 20601 specification describes the normal conditions for associating and un-associating without regard for a feature like HDP "pause” and “resume" (disconnection and reconnection). This is because the IEEE standards attempt to be independent of the type of transport used and HDP’s design of disconnection and reconnection is transparent to the IEEE layer.

It is therefore recommended that implementations retain their state when Bluetooth intentionally disconnects to save redundant configuration steps upon a future HDP reconnect. If this is not done, two cases are possible. 1) a Reconnect initiated by the other device fails because the remote device is no longer in the Operating state or 2) the Reconnect is successful, but the Agent/Manager notices that their states are out of sync and fall back to association and configuration. Either case will be inefficient. Therefore, to maximize efficiency, it is important that both sides follow these guidelines.

PSMs cannot be guaranteed to be persistent between connections, so it is strongly recommended that the PSM be retrieved from the SDP record before each (re)connection. Before retrieving the PSM from the SDP record, the optional ServiceRecordState attribute from the SDP record should be retrieved first. If this attribute value is unchanged since the last query, this indicates that the PSMs have not changed since last time the ServiceRecordState was sampled. However, if this attribute value has changed or the attribute does not exist in the SDP record, then it is strongly recommended that the PSM be retrieved from the SPD record.

3.2 FRAME DELIVERY AND DISCONNECTIONS

In addition to the CRC and retransmission protocol in the Bluetooth baseband, L2CAP allows for a high degree of confidence that a frame is delivered correctly or not at all. By utilizing Enhanced Retransmission Mode, frames (SDUs), or portions of the SDU called PDUs, are transmitted using sequence numbers and acknowledgements. Packets not received between the two devices' L2CAP layer retransmit to ensure that no packets are lost.

It is safe to assume that connections will always be lost symmetrically. The enforcement of this expectation is by the master/slave format and means that a slave could certainly not transmit without a connection to the master (receiving a slot). The slave is required to respond only if the master has sent something to it (user data or a Bluetooth Link Manager Protocol PDU) or if the master has sent a POLL packet to the slave. The Bluetooth baseband implementations take care of forcing a response if the previous packet from the slave was so long ago that it approaches the Link Supervision Timeout [3]. This allows the master to notice the absence of the slave. The implication is that no situation would arise when the ACK is lost because the "Sink/Manager" may not be able to perform reconnect, or the "Source/Agent" may not receive report that connection is lost, so the “Source/Agent” would have to perform a retransmit of any unacknowledged frames in its buffer.

The need for an API/Command Set with both generic and link-specific functionality between Data Exchange Protocol and HDP is required, but it is not be part of the Bluetooth specification. Internal API definitions are up to implementers, since only the external communication requires standards to ensure interoperability.

3.3 DEVICE SPECIALIZATIONS

Device Specialization specifications [5] are defined to work specifically with the IEEE 11073-20601 Data Exchange Protocol and provide details on how the 20601 tools are applied to for a given health device's implementation. Valid Device Data Specializations are defined by the MDEP Data Type attribute values and are maintained in the Bluetooth Assigned Numbers. Sources and Sinks initiating connection to a remote device should first check for a compatible MDEP Data Type before attempting to establish an HDP connection. This would avoid the unnecessary pairing between HDP devices that do not have any MDEP Data Type values in common and thus cannot interoperate.

Several Device Specialization standards are in various stages of development and the assignment of new values will occur as these become available and as requested by HDP developers. As of the writing of this document, Device Specializations that have been approved for use with HDP include weight scales, blood pressure monitors, glucose meters, thermometers and pulse oximeters. Members of the Bluetooth SIG who desire support for devices not listed in the Bluetooth Assigned Numbers are to submit their request to med-feedback@bluetooth.org. All requests for new numbers follow a formal process within the MED WG and BARB and a link to that process can be found from that section of the Assigned Numbers page.

3.3.1 MAX MTU FOR DEVICE SPECIALIZATIONS

The 20601 standard states generically that the maximum size of the Application PDU (APDU) to be sent by a Manager to an Agent is 8KB and the maximum size of the APDU to be sent from an Agent to a Manager is 64KB. Since devices can vary in complexity significantly, 104xx Device Data Specializations for some devices specify a lesser max size for the APDU that can be sent from the Manager to Agent and from the Agent to Manager. The table below summarizes these requirements for approved device specializations at the time of writing this document. The maximum APDU will affect the maximum MTU that is needed to support a given Device Data Specialization by HDP. If multiple Device Specializations are used over the same MDL, support for the largest APDU is required.

Within each IEEE Device Specialization there is a section titled “Communications characteristics” which defines the limits on the size of an APDU transmitted or received by an Agent.

The two terms Ntx and Nrx specify the APDU size. An Agent device shall not transmit any APDU larger than Ntx and shall be capable of receiving any APDU up to a size of Nrx. The APDU of an e.g. pulse oximeter Source may be as large as 9216 octets, which implies that the pulse oximeter Sink shall have an L2CAP MTU of 9216 octets; L2CAP takes care of segmentation and reassembly of these large SDUs and exchanges Protocol Data Unit (PDU) packets.

The maximum APDU should be an optimum value based on the platform specific resources of the local device, such as controller buffer size, supported packet types, etc.

Note: The IEEE Device Specializations use the term ‘octet’, which is defined as a “one byte bit string” in the 20601 specification. For the purposes of this document, the terms octet and byte are interchangeable.

Device Specialization APDU maximums as specified from relevant IEEE device specializations are shown in the following table.

IEEE Device Specialization	**Ntx (octets)**	**Nrx (octets)**
10404 – Pulse Oximeter	9216	256
10407 – Blood Pressure Monitor	896	224
10408 – Thermometer	896	224
10415 – Weighing Scale	896	224
10417 – Glucose Meter	896	224

Table 3.1

4. Resources Required for HDP Implementation

The aim of this section is to provide an overview of the resource requirements in order to develop a system based on HDP.

4.1 BLUETOOTH CORE SPECIFICATION REQUIREMENTS

HDP requires at least a Bluetooth version 1.2 Controller together with at least a version 2.0+EDR Host along with support for Volume 3, Part A of Core Specification Addendum 1 (which specifies L2CAP enhancements, such as Enhanced Retransmission Mode, Streaming Mode, and Frame Check Sequence (FCS) options). As of Core Specification version 3.0 + HS, the enhanced L2CAP features defined in Volume 3, Part A of Core Specification Addendum 1 are included as a normative part of Bluetooth Core specifications.

It is strongly recommended that new implementations of HDP should use at least Bluetooth Specification 2.1 to take advantage of Secure Simple Pairing. This improves the security of the connection and significantly reduces the risk of a man-in-the-middle attack during initial pairing. See also Section 7.1.

HDP specifies the use of some enhanced features of L2CAP [3] (e.g. Enhanced Retransmission Mode, Streaming Mode, Segmentation and Reassembly and FCS options). Refer to HDP [1] for details of these requirements. These enhanced L2CAP features are used to define a reliable Control Channel and allow for flexible Data Channel configuration for Reliable Data Channels or Streaming Data Channels. The FCS options defined in CSA 1 and Bluetooth Specification 3.0 + HS include the ability to disable FCS. Resource-constrained Sources may take advantage of the FCS option to disable FCS for its Data Channels to reduce processing power when appropriate.

4.2 HARDWARE REQUIREMENTS

The HDP and MCAP specifications provide low requirements for RAM and ROM/flash; however, based on the variety of Bluetooth stacks currently on the market and the wide variety of implementations, the resources and requirements can vary significantly.

Minimal implementations of MCAP and HDP require just a few KB for the code and RAM. At the other extreme, there will be systems capable of handling more than one instance of HDP at a time, and more than one HDP connection at a time for each instance. System requirements for RAM and ROM/flash will depend on the needs of the application. A simple sensor may not need a non-volatile memory to store data, whereas systems such as a data logger may have the need to store a large quantity of data.

MCAP defines an optional Clock Synchronization Protocol (CSP). This feature may require access to the Bluetooth Clock in a host system, which means that the Bluetooth controller must support the HCI_READ_CLOCK HCI command or have an alternative way to read the Bluetooth controller clock. The CSP may also need access to a local hardware timer in order to improve the accuracy, but this is implementation specific.

The IEEE11073-20601 Data Exchange Protocol and 104xx Device Specialization layers will also require some extra RAM and ROM/flash. It is important to note that, in general, health sensors (e.g. glucose meter, pulse oximeter, weight scale, blood pressure meter) tend to have lower processing power, memory, and storage space than a Sink device used to collect data. For this reason, the 11073 protocols intentionally place more burden on the Manager devices (Sinks) than on the Agents (Sources).

Depending on the use case, a HDP application may require non-volatile memory in order to store received data and MDL status information to allow efficient MDL reconnection.

4.3 SOFTWARE REQUIREMENTS

For an application to comply with HDP, the implementation of all mandatory features of MCAP and HDP is required. As noted in Section 4.1, HDP requires a Bluetooth stack that implements enhanced features of L2CAP [3].

Developing HDP on open platforms, such as a mobile phone or a PC, requires interfacing directly with the L2CAP and SDP layers.

4.4 MULTI-PROFILE CONSIDERATIONS

It is likely that devices implementing HDP will also implement other profiles such as A2DP, for example in a mobile phone. In this case, both services may be active at the same time, sharing resources but running independently. In this example, it is important that A2DP streaming is not stopped nor its streaming rate affected while at the same time HDP data transfer can proceed. The Bluetooth SIG is currently working on multi-profile recommendations such that one profile can work well with another. Refer to www.bluetooth.org for developing recommendations in this area.

5. Migration from SPP to HDP

Although it is highly desirable that all the Bluetooth enabled health devices on the market make a quick transition to HDP, it will take time for products based upon the Serial Port Profile (SPP) profile to migrate fully to HDP.

SPP is convenient because it is widely supported by existing Bluetooth protocol stacks, both on PC and mobile devices. It is also easy to implement on standalone systems thanks to the availability of pre-certified modules. Such modules act as a “Serial Port Replicator”, transferring data using SPP from and to a real hardware UART interface. Bluetooth systems based on such hardware partitioning schemes usually require a dedicated microcontroller for the application. The interface between the application processor and the module is typically vendor specific.

On the other hand, implementing HDP provides clear benefits of interoperability, ease of use, and some additional functionality. For this reason, it is important for some manufacturers to create a migration path that can simplify the ultimate adoption of HDP.

Although HDP modules and stacks are now available, the migration from SPP to HDP is not just a switch of modules within a health device. Some modules, for instance, may implement only HDP and leave the data layer up to the external microcontroller.

Switching from an SPP module to a HDP module may require other changes because the command interface used with SPP modules lack definition by any profile; rather, it is vendor specific. The primary advantage of using modules is that HDP functionality is contained within the module itself.

Early products implementing HDP might also include SPP, to support interfacing with current non-HDP systems (such as PCs or mobile devices). Once HDP becomes ubiquitous, the expectation is that the numbers of health devices using SPP will decrease markedly. In the mean time, some manufacturers employ various transition strategies.

5.1 CONSIDERATIONS FOR SUPPORTING LEGACY SPP

It is likely that some of the first medical, healthcare and fitness devices implementing HDP that are transitioning away from Serial Port Profile (SPP), both Source and Sink, would also include support for SPP in order to guarantee backward compatibility. In this case, it could be acceptable that once a connection to a service establishes, the other service is unavailable.

SPP implementations are unlikely to be compatible with devices from other manufacturers using SPP and is only discussed here as a migration strategy. This was a fundamental reason for the creation of HDP.

The command format over SPP is proprietary, as the Bluetooth SIG or any other group did not standardize it. Therefore, the application implementation that needs to support both HDP and legacy SPP must be able to handle the proprietary legacy commands as well as HDP in order to be interoperable with both new and legacy devices.

Figure 5.1 shows block diagrams depicting transitional implementations of HDP devices that allow some level of backward compatibility as a manufacturer transitions their product line from full SPP to full HDP over a period. When it is required to support communications using SPP, an SPP wrapper module should be provided to convert HDP data to SPP for the transmitted data and SPP to HDP format for received data.

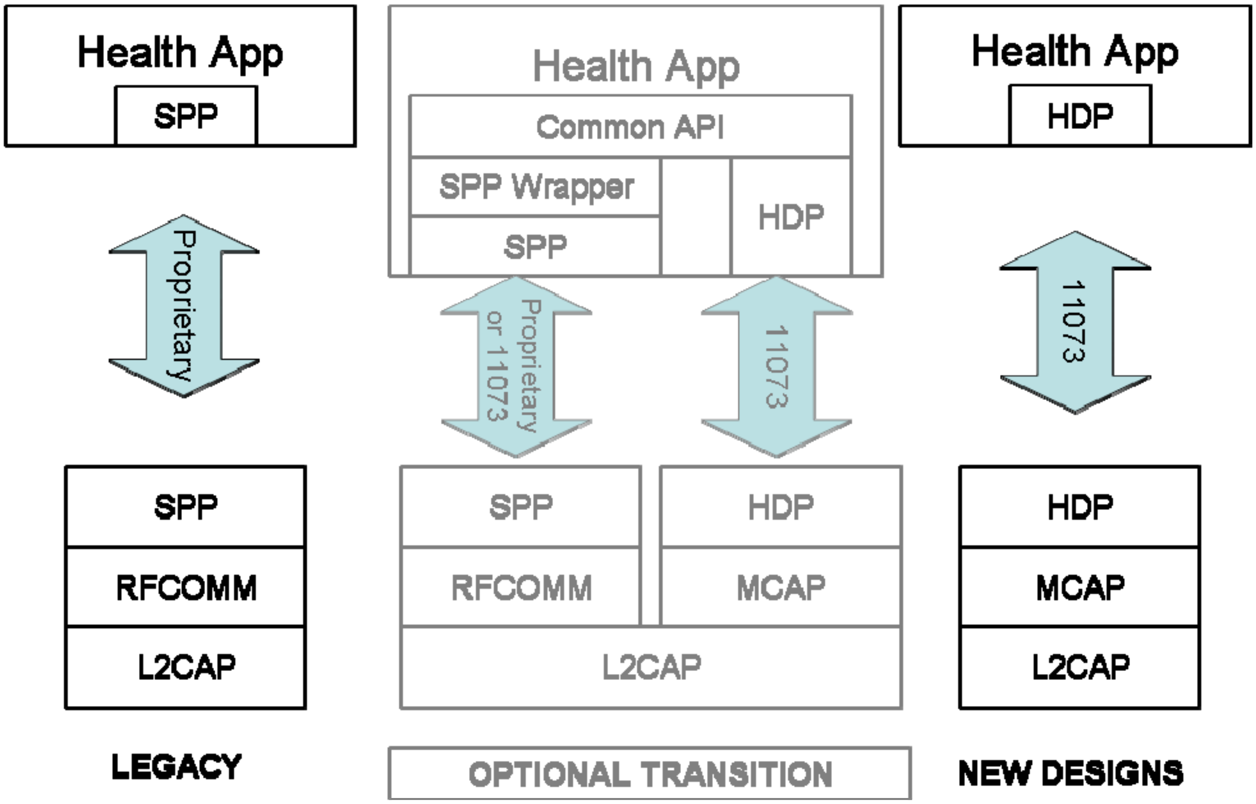


Figure 5.1: Transition from Legacy SPP to HDP

The following steps should be followed for a health and fitness application that needs to support both HDP and legacy SPP during a market transitional stage:

1. Make an SDP query for a peer device.
2. Search for HDP attributes.
3. If HDP attributes are not found, check for SPP attributes.
4. Initiate an HDP or SPP connection based on the results of 2 and 3.

6. Power Management

Because many Bluetooth health devices in today's market operate on battery power, with many smaller devices powered from small coin-cell batteries, it is important to take advantage of the low power features offered by the Bluetooth Core Specification and HDP. It is estimated that 80% of healthcare devices are battery powered, and this proportion will be even higher for smaller fitness devices. User expectations are to maintain a minimum battery life of greater than 6 months for a simple device (e.g. episodic physiological sensor with AA batteries used one time per day); however, the expectations placed upon smaller fitness devices will generally be for a minimum battery life of greater than one year. In order to prolong battery life, development efforts should allow implementations to take advantage of power savings features such as Sniff mode, which is already available in Bluetooth, and the MCAP reconnect feature.

6.1 SNIFF MODE OVERVIEW

Bluetooth offers a power saving mode of operation, called *Sniff*, which allows two devices to save power by negotiating periods of time during which the basebands will not exchange data packets over the air. This allows the receiving device, and to a lesser extent the sending device, to use less power. This mode is ideal for health devices that only exchange data periodically and stay connected between transmissions (i.e. heart rate monitors, pedometers, etc.) as it allows them to save power in between data transfers. Streaming devices can also operate in Sniff mode, depending on their required data rates. Placing the connection in Sniff mode when not transferring data, also frees air bandwidth for use by other device connections that the controller may be servicing.

Bluetooth version 2.1+EDR includes an additional low power feature called Sniff Subrating and allows even greater power savings. Additional information about Sniff modes is available in the Bluetooth whitepaper, "*Bluetooth Master/Slave Communications and Sniff/Sniff Subrating Modes*" [7].

Sniff mode is recommended for most devices; however, the parameters to use are outside the scope of this document. These parameters depend upon the device of which Sniff mode is to be implemented into, and upon the types of devices with which that device will be expected to interoperate. Sources are typically slaves, especially in the case of when the Sink is collecting data from multiple devices.

Refer to Section 8.5 for considerations when using Sniff Mode with the Clock Synchronization Protocol.

6.1.1 SNIFF MODE OPERATION

Sniff mode allows two devices to save power by negotiating periods of time during which the two devices' Link Managers agree to sustain transmission. This differs from the normal mode, in which the receiver in the slave device has to be active all the time (thus the normal mode would not optimize power consumption while being active at times when there is no data for transfer). The Host device can inform the Controller of its profile needs by sending its Sniff parameters. This includes information such as the minimum and maximum interval, the Sniff attempt, and the Sniff timeout[2]. The two Link Managers then negotiate each device's parameters and pick an appropriate Sniff interval.

6.1.2 SNIFF SUBRATING

Sniff Subrating, available in Core 2.1 + EDR and later versions of the Bluetooth Specification, provides a means for further reduction of the power consumed by link management. SSR allows either device to increase the time between listening for data packets. While this change may reduce the responsiveness of the link, it also may reduce the number of packets exchanged to maintain the link, and thus reduces power consumption. The master is forced to keep the slots represented by the underlying sniff mode setup to be reserved for this particular slave device. It holds the configuration even if other connections are requested and established in the piconet. Slots can be reserved and are always available to the application. The application can choose to use them as needed.

This mode is best suited for data that is intermittent by nature or for applications where the data interval changes during the life of the connection. ECGs, pulse oximeters and heart rate monitors are good examples of devices that might use Sniff Subrating to transmit data to a collector, where the rate of transmission could change intermittently depending on whether the user is stationary, walking or running.

Sniff Subrating mode is used in conjunction with Sniff mode, and is only used when the controllers on both devices can negotiate it based on their Hosts' requirements.

6.2 ENHANCED DATA RATE (EDR)

Devices based on Versions 2.0 + EDR and later of the Bluetooth Core Specification, may offer the use of Enhanced Data Rate (EDR) packets that allow data packets to be transmitted quicker. The amount of energy consumed by a Bluetooth device depends on the length of time it is active. Since EDR packets can allow data

[2] Refer to section 7.2.2 Sniff Mode Command, Part E – Host Controller Interface Functional Specification, Volume 2 – Core System Package, Core Specification 2.1 + EDR.

to be transmitted up to three times faster than non-EDR packets, the radio only needs to be active a third of the time when compared to Basic Rate. Therefore, the energy required to transmit the data is reduced when large amounts of data are transferred. It should be noted that there is little or even negative advantage in using EDR when the amount of data in any given transmission is small.

6.3 RECONNECT FEATURE

It is recommended that an HDP device having enough state memory, take advantage of the reconnect feature defined in the MCAP specification. This feature allows devices to reconnect using an already established MDL ID, thus saving redundant reconfiguration steps and reducing the number of bytes required for transfer over the air. This feature is very useful for devices such as weight scales, blood pressure meters and glucose meters that typically send a measurement, disconnect and turn off. This can also be very useful in scenarios where the vital signs of ambulatory patients are monitored continuously (e.g. ECG, SpO2), and those patients routinely walk in and out of range of the monitors.

6.4 POWER SAVINGS ROADMAP

The Bluetooth SIG is constantly looking for ways to improve the usage of power. As of this writing, Bluetooth low energy wireless technology is nearing final adoption within the Bluetooth SIG. This technology will allow some user scenarios to transfer small packets of data on a connectionless basis with a different link layer. It is best suited for devices that periodically need to transmit small amounts of data. The Medical Devices Working Group is currently working on ways to make use of this new technology, where appropriate. See the Bluetooth low energy wireless technology Marketing Requirements Document [9] for details and user scenarios. Initial consumer products based upon this technology are expected to become available in 2010.

7. Data Reliability, Authenticity and Privacy

This section addresses reliability, authenticity, and privacy of data transfers. Privacy and authenticity both imply some knowledge of the partner, while reliability is specific to the channel and endpoint design, so the discussion is divided into two sections.

7.1 PRIVACY AND AUTHENTICITY

A private message is one that has not been read by any (unauthorized) third party. If some sensor “A” is required to send a measurement to a computer “B” without interception, then “B” must first share some secret with “A” that allows each to verify the identity of the other. If this were not the case, then some attacker “C” could first falsely claim to be “B” to receive the message from “A”, then falsely claim to be “A” when relaying the message to “B”. In this case, the unauthorized “C” has read the message, and privacy has been compromised. This scenario describes a Man-In-The-Middle (MITM) attack.

As will be briefly discussed in Section 7.2, loss of privacy through a successful MITM attack can lead to compromised data integrity. This adds to the importance of foiling MITM attacks.

In order to prevent MITM attacks, shared secrets exchange when devices pair. Prior to Bluetooth version 2.1+EDR, this occurred by transmitting the value over a channel encrypted with a key based on a user-entered PIN. These PINs were very often easy to guess, making the secret accessible to attackers with special hardware if the attacker could observe this pairing. Although this is normally a very small weakness, attackers could cause the pairing to be unnecessarily repeated. These multi-stage attacks cast doubt on Bluetooth security, although the actual channel encryption and authentication were as secure as the shared secret.

Bluetooth version 2.1+EDR uses a more sophisticated exchange of secrets [8] such that even if all the messages are intercepted, the attacker cannot determine the shared secret. The only risk is that the MITM attacker will establish itself as a permanent middle-man in the communication (it cannot determine the value of a shared secret between “A” and “B”, but it can still share a single secret with “A”, and another with “B”, so each thinks it is communicating with the other). To defeat this, Secure Simple Pairing (SSP) uses displayed or typed values based on the shared secret to ensure that the two devices the user has in hand are actually sharing the same secret, which then guarantees that no MITM attacker has been inserted into the communication.

Having established a shared secret, “A” and “B” can use that value to authenticate subsequent communications as being genuinely from the device with which they originally exchanged that secret, and can also use it to generate encryption keys so their subsequent exchanges cannot be read by third parties. HDP requires that authentication and encryption are to be used for all links.

Bluetooth encryption is generally processed in hardware, so it is not usually useful for secure local storage on persistent media.

7.2 RELIABILITY

All Bluetooth packets are protected by a CRC[3] (Cyclic Redundancy Check), and are acknowledged at the Baseband level when received. In addition to that, HDP mandates the use of L2CAP Enhanced Retransmission Mode [3] (for Control Channel and reliable Data Channels) or L2CAP Streaming Mode [3] (for streaming data Channels). Both of these two modes provide for FCS[4] (Frame Check Sequence) options that affect reliability. While the CRC verifies the reliability of the Baseband packets through which an L2CAP packet is transferred, the FCS verifies the integrity of the contents of the L2CAP packet, allowing the ERTM protocol to retransmit when needed.

As noted in Section 7.1, a successful Man-In-The-Middle attack can lead to loss of reliability, as well as loss of privacy. If, in the example of that section, “C” can convince “A” that it is “B”, and can convince “B” that it is “A”; then “C” can do more than just read the messages and relay them as it can also alter them. For this reason, reliable authentication and encryption are more than just a privacy issue. Designing a good pairing solution is important to designing a good system, and balancing ease-of-use with security is addressed in the Secure Simple Pairing Whitepaper [8].

8. Clock Synchronization Protocol

This section provides an overview of the Clock Synchronization Protocol (CSP), an optional feature of the MCAP (and HDP) specification. It allows devices to time synchronize their data streams to a high degree of precision.

It is noted that MCAP allows independent use of the CSP without the need to support standard MCAP op codes. This is to allow other profiles to be able to take advantage of CSP in the future.

CSP relies on the Bluetooth Clock used on the BR/EDR link. Even if an AMP is used, a BR/EDR link is still required.

8.1 GENERAL DESCRIPTION

Precise timing synchronization between the input and output of two or more devices in the millisecond (or even microsecond) range is necessary for various applications, including synchronized data collection from several high speed sensors (such as those measuring vibration, acceleration, propagation delay, or coordinated event initiation). All these applications need not only the information *that* a specific event has occurred (or will occur), but also the precise declaration of *when* it occurred (or will occur) in order to combine (or coordinate) the information of multiple devices.

To enable the aforementioned types of applications, a highly precise timing synchronization of the data from wireless devices is needed. Since Bluetooth devices rely on error correction based on lower layer

[3] Baseband ACL data packets use a 16-bit CRC with generator polynomial $g(D) = D^{16}+D^{12}+D^5+1$

[4] The FCS is another CRC. The default setting for FCS is a 16-bit CRC, based on the generator polynomial $g(D) = D^{16}+D^{15}+D^2+1$

retransmissions, there is no fixed end-to-end propagation delay provided to synchronize data points. The CSP, described in MCAP, takes advantage of the time-slotted nature of Bluetooth radio links. It also takes advantage that the master numbers every slot as a “Bluetooth slot” and the start times of the master transmissions lock to the master’s internal clock. The slaves synchronize a piconet clock to the receptions from the master and maintain accuracy typically within 10 microseconds. This provides a mechanism for high-resolution synchronization of the data streams of Source devices in a piconet using a coordinated Time-Stamp mechanism.

The CSP can be used to precisely synchronize the 64-bit Time-Stamps of one or more Sync-Slaves that are connected to a Sync-Master via a Bluetooth radio link or allow the recalculation and timing error correction of the timing of events. These Time-Stamps are normalized to microseconds (μs).

8.2 CLOCK SYNCHRONIZATION ROLES

The CSP defines the roles of Sync-Master and Sync-Slave.

8.2.1 SYNC-MASTER

The device that requests Time-Stamp information, or configures synchronization requirements on a remote device is called Sync-Master for the time the corresponding request is processed by the Sync-Slave.

8.2.2 SYNC-SLAVE

The device that receives a request for Time-Stamp information, or receives configuration synchronization requirements from a remote device master is called Sync-Slave for the period during which the corresponding request is processed.

8.3 OPERATION MODES

There are two types of operating modes depending on the required degree of resolution.

8.3.1 LOW RESOLUTION USAGE

- **Simple Time-Stamp Clock Set**

(MD_SYNC_SET_REQ -> BluetoothClock_SyncTime = 0xFFFFFFFF)

In this simple usage of CSP, the Sync-Master just requests the Sync-Slave to set its Time-Stamp Clock to a specific value without any synchronization to the Bluetooth Clock, so no Bluetooth Clock access is needed on Sync-Master or Sync-Slave side to perform this operation.

- **Simple Time-Stamp Clock Read**

(MD_SYNC_SET_REQ -> BluetoothClock_SyncTime = 0xFFFFFFFF, TimeStamp_SyncTime= 0xFFFFFFFFFFFFFFFF)

Using the Read commands, the Sync-Master requests the Sync-Slave to return the Time-Stamp Clock actual value without any synchronization to the Bluetooth Clock, so no Bluetooth Clock access is needed on Sync-Master or Sync-Slave side to perform this operation.

The achieved precision of the described “Set” and “Read” operations depend on the propagation delay of the Bluetooth connection. Since the timing error cannot be determined due to the lack of a common time base (like the Bluetooth Clock) on the involved devices, this “Simple Time-Stamp Clock Set/Read” is only for use cases where the precision requirements are not high. These use cases also can accept a possible timing error in the range of some 10ms.

8.3.2 HIGH RESOLUTION USAGE

- **Synchronized Time-Stamp Clock Set**

(MD_SYNC_SET_REQ -> BluetoothClock_SyncTime != 0xFFFFFFFF)

In this complex usage of the CSP, the Sync-Master requests the Sync-Slave to set the Time-Stamp Clock to a specific value in synchronization to the Bluetooth Clock, so Bluetooth Clock access is needed on Sync-Master and Sync-Slave side to perform this operation.

- **Synchronized Time-Stamp Clock Read**

(MD_SYNC_SET_REQ -> BluetoothClock_SyncTime != 0xFFFFFFFF, TimeStamp_SyncTime= 0xFFFFFFFFFFFFFFFF)

In this complex usage of the CSP, the Sync-Master requests the Sync-Slave to return the Time-Stamp Clock value in synchronization to the Bluetooth Clock, so Bluetooth Clock access is needed on Sync-Master and Sync-Slave side to perform this operation.

The achieved precision of the “Set” and “Read” operations do not depend on the propagation delay of the Bluetooth connection, but rather only on the capabilities of the involved devices. This “Synchronized Time-Stamp Clock Set/Read” is for use cases where the precision requirements are high and a possible timing error in the range of the capabilities of the involved devices is required.

The “Synchronized Time-Stamp Clock Set/Read” can be used not only to synchronize the Time-Stamps of Sync-Master and Sync-Slave, but also to synchronize the Time-Stamps of multiple Devices in one Bluetooth piconet. The latter case can be achieved by a Sync-Master not having access to the Bluetooth Clock, by using the Bluetooth Clock readings in the Sync-Slaves responses (with some calculation to be performed by the Sync-Master).

8.4 CSP CLOCK DESCRIPTIONS

8.4.1 BLUETOOTH CLOCK

Every Bluetooth device has an internal system clock (i.e. the Bluetooth Clock), which determines the timing and hopping of the transceiver. If two devices share a Bluetooth connection, then the master’s Bluetooth Clock is used as the master clock for the connection as defined in the Bluetooth Core Specification [3].

It should be noted that the Bluetooth Clock is not related to the time of day. The Bluetooth Clock provides the “heart beat” of the Bluetooth transceiver. Its resolution is exactly half the TX or RX slot length, or 312.5 µs. The period of the Bluetooth Clock is approximately 23.3 hours.

Through the requests and responses described in Section 8.3, the common Bluetooth Clock (which is common to all devices in a piconet, but not necessarily directly accessible by the upper layers of each device) can be used to synchronize an internal Time-Stamp Clock, if at least the Sync-Slave is capable of directly accessing the Bluetooth Clock. The particular Bluetooth Clock, which is used as the time base for piconet synchronization, can be accessed via the HCI_READ_CLOCK command that is defined in the Bluetooth Core Specification [3].

8.4.2 TIME-STAMP CLOCK

The Time-Stamp Clock is a clock with a resolution of one microsecond (µs). It is defined as a 64 bit counter.

8.5 CLOCK DRIFT CONSIDERATIONS FOR TIMING PRECISION

The Bluetooth Clock in the Bluetooth slave devices will drift with respect to the master’s clock during times when the slave is not receiving packets from the master. The periods of non-reception will be caused by lack of master transmissions, radio link degradations that prevent the slave from receiving packet for variable periods. In addition, times when the master and/or slave are invoking Bluetooth low power modes that allow for lack of radio channel activity. How slaves handle these periods of loss of packet reception is not specified and thus the accuracy of Time-Stamps in slaves may drift at rates of 40 to 500 µsec/sec (i.e. 20 to 250 µsec/sec relative to the master’s clock). The drift occurs at the exact point where the Time-Stamp Clock is reset is all that matters. Any “indications” sent after the Set event will have the same sort of variance.

The "TimeStamp_NativeAccuracy" is the worst-case drift of the Time-Stamp. Therefore, if a Sync-Slave allows low power mode while a pending Time-Stamp synchronization, the Sync-Slave must include that into its worst-case calculations. Since receiving an MD_SYNC_INFO_IND message from the Sync-Master re-synchronizes the Bluetooth Clocks, the Sniff interval of a MCL link is not allowed to exceed the send period of the MD_SYNC_INFO_IND messages.

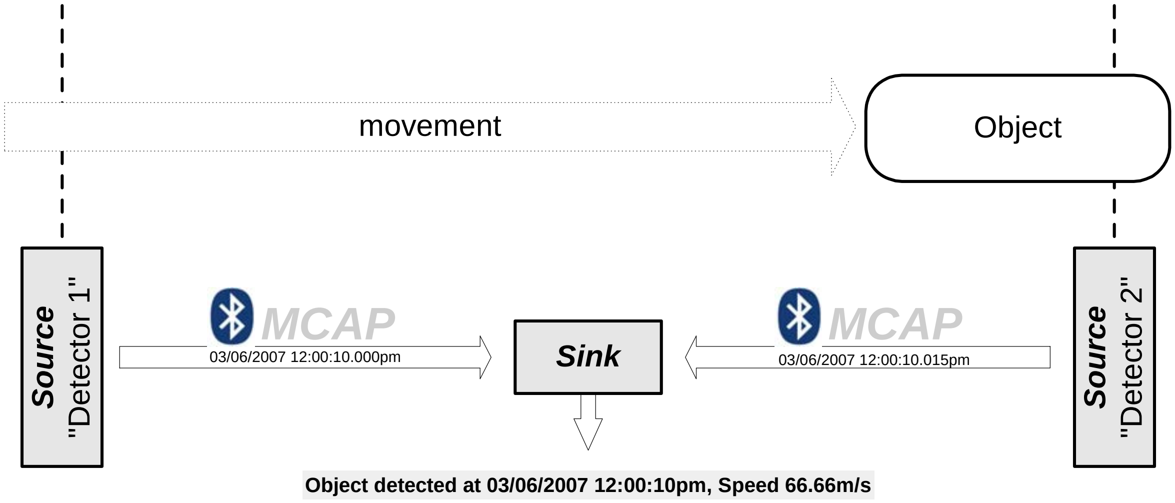
It is also important to note that the master is allowed to go “to sleep”, that is, operate off a low power oscillator that can have an accuracy as poor as 250 ppm. All devices in the piconet will then be synchronized to the master’s clock, but this clock is based on an oscillator with only 250 ppm accuracy.

8.6 CSP USAGE EXAMPLES

This section shows some examples for the CSP.

In these examples, MCAP transports event information for events that occur spontaneously. These hypothetical examples require not only the fact *that* an event occurred, but also the information *when* it occurred in relation to other events.

A "real world" example might be that two devices, each with an optical detector, report whether or not their sightlines are occluded by sending messages through an MCAP connection to a Computation Engine. The Computation Engine then uses this information to calculate the direction and speed of the object responsible for occluding the sightlines.


movement
Object
Source
"Detector 1"
MCAP
03/06/2007 12:00:10.000pm
Sink
MCAP
03/06/2007 12:00:10.015pm
Source
"Detector 2"
Object detected at 03/06/2007 12:00:10pm, Speed 66.66m/s

To fulfill this task, the Computation Engine needs to know 1) which order the two events (the occlusions of Detector 1 and Detector 2) occur to determine the direction of the moving object, and 2) the delay between the two events to determine the speed of the object. The CSP can be used in several ways to enable the Computation Engine to fulfill these tasks.

In these examples, the hypothetical "higher layer" protocol (in place of IEEE 11073-20601 and compatible Device Data Specializations) that would communicate events is always the same and quite simple; it consists of exactly one message:

"ReportEvent" ➔ indicates that an event (a sightline occlusion) occurred at Time-Stamp Clock time [SysTime]

For simplification, a possible wrap-around of the Bluetooth Clock and Time-Stamp Clock is not considered in calculations, and TimeStamp_SampleAccuracy is not considered.

All of the examples described below assume the following:

- Distance [D] between light barriers is 1 meter
- Required accuracy [A] of calculated speed is 0.001m/s
- The accuracy of the oscillator [O] of all three devices is 10ppm

8.6.1 EXAMPLE 1: RECALCULATING EVENT TIMING - BASIC CASE

In this example, the Sync-Slave uses a 625μs counter as a base for its Time-Stamp Clock. (Note that this must be a local clock and not the common Bluetooth Clock)

- **Step1: Sync-Master actions:**

  Perform calculation to convert use case accuracy requirements [A] and [D] to required accuracy [R] in ppm:

  [R] = 1s*([D]/[A]) = 0.001s = 1000ppm

  Since we have two slaves with oscillators that might drift in opposite directions:

  [R] = [R]/2 = 500ppm

  Send to Sync-Slaves:

```
        MD_SYNC_CAP_REQ
{       TimeStamp_RequiredAccuracy = 500 ([R], just calculated)
}
```

- **Step2: Sync-Slave reactions:**

  Respond with own capabilities:

```
        Send MD_SYNC_CAP_RSP
{       BluetoothClock_NativeResolution = 2 (access to full slots of local Bluetooth Clock)
        SyncLeadTime  = 10 (time needed to read Bluetooth Clock)
        TimeStamp_NativeResolution = 625 (one local clock counter slot)
        TimeStamp_NativeAccuracy = 10 (as defined in [O])
}
```

- **Step3: Sync-Master actions:**

  Send to Sync-Slaves:

```
        MD_SYNC_SET_REQ
{       TimeStamp_UpdateInformation = 1 (request MD_SYNC_INFO_IND messages)
        BluetoothClock_SyncTime = 0xFFFFFFFF (request instant synchronization)
        TimeStamp_SyncTime = 0 (or some other meaningful value)
}
```

- **Step4: Sync-Slave reactions:**

  Send MD_SYNC_SET_RSP with "success"

  Read Bluetooth Clock to [B0] (where B0 is a register or someplace to store information)

  Send to Sync-Master:

```
        MD_SYNC_INFO_IND
{       BluetoothClock_SyncTime = [B0]
        TimeStamp_SyncTime = TimeStamp_SyncTime (from MD_SYNC_SET_REQ)
        TimeStamp_SampleAccuracy = 0 (or some other meaningful value)
}
```

At this point, the Sync-Slave can calculate the Time-Stamp [MT] at a given event time [T] using the formula:

Where [MT] = ([local clock counter at time [T]] - [B0])* TimeStamp_NativeResolution + TimeStamp_SyncTime

Now, the Sync-Slaves send MD_SYNC_INFO_IND messages with a Time-Stamp Clock [MT], a Bluetooth Clock [BT] and the SampleAccuracy of this single instant in at least an interval [I] that allows the Sync-Master to keep track of the drift of the Sync-Slave's oscillators:

In this Example, [I] = 1s*[R]/[O] = 50s (the maximum length of time that the devices can remain incommunicado while maintaining the required accuracy).

- **Event 1: is detected by Sync-Slave-1 at time [T1]:**

Send to Sync-Master:

```
        "ReportEvent"
{       SysTime = Time-Stamp at [T1]
}
```

- **Event 2: is detected by Sync-Slave-2 at time [T2]:**

Send to Sync-Master:

```
        "ReportEvent"
{       SysTime = Time-Stamp at [T2]
}
```

- **Sync-Master reaction:**

Calculate the relative drift of the Slaves:

[E] = (last[MT]-[M0])-((last[BT]-[B0])*TimeStamp_NativeResolution)

([E] has a signed result)

Calculate corrected Time-Stamps of Slaves:

[TCx] = SysTime[Tx]+[Ex]

Calculate time difference from [TC1] to [TC2]:

MoveByTime = abs([TC1]-[TC2]) (the relative drift of the time references of the two slaves)

8.6.2 EXAMPLE 2: RECALCULATING EVENT TIMING - ADVANCED CASE

In this variation of Example 1, the Sync-Slave offers a highly precise Time-Stamp Clock with a precision of 1μs. This might be realized with a counter (CPU register) on the Sync-Slave that is configured in a way that it is automatically incremented every μs by the CPU hardware. The MCAP implementation of this hypothetical advanced Sync-Slave is able to access the Bluetooth Clock directly and instantly in its full half-slot resolution.

- **In this example variation, "Step2" of Example 1 in Section 8.6.1 would look like this:**

Respond with own capabilities:

```
    Send MD_SYNC_CAP_RSP
{   BluetoothClock_NativeResolution = 1 (access to half slots of local Bluetooth Clock)
    SyncLeadTime  = 0 (time needed to read Bluetooth Clock)
```

TimeStamp_NativeResolution = 1 (full resolution)

TimeStamp_NativeAccuracy = 10 (as defined in [O])

}

All other "steps" and calculations are unchanged.

8.6.3 EXAMPLE 3: CALIBRATING SYNC-SLAVE TIME-STAMP TO AVOID RECALCULATION

In this variation of Example 1, the ability of the CSP to configure the Time-Stamp in relation to a specific Bluetooth Clock instant is used to ease the task of the Sync-Master and avoid the processing power consuming and complex recalculation of event timing.

For simplification, we assume in this variation that the Sync-Master and the Sync-Slaves are part of the same Bluetooth piconet with the Sync-Master as Piconet-Master.

➢ **Description of Approach taken:**

First, the Sync-Master synchronizes the Time-Stamp of the Sync-Slave to the "OS-Clock" of the Sync-Master's internal Operating System that runs in "UNIX time" in this example as defined in "POSIX.1/IEEE 1003.1-2001" [10] (the "UNIX time" basically counts the seconds since 01 January 1970).

To do that, the Sync-Master reads/generates an instant of the Bluetooth Clock and his OS-Clock (basically indicating: at Bluetooth Clock [X], the Sync-Master's OS-Clock was [Y])

Then, the Sync-Master takes the SyncLeadTime that was indicated by the Sync-Slaves in the MD_SYNC_CAP_RSP message to calculate a time instant in the future (basically indicating: at Bluetooth Clock[X+SyncLeadTime], the OS-Clock of the Sync-Master will be [Y+SyncLeadTime])

Having done that, the Sync-Master converts the future OS-Clock time [Y+SyncLeadTime] to μs, to make it compatible to the Time-Stamp format (in this example, it has to be multiplied by 1000000, since the OS-Clock resolution is 1 second, but the Time-Stamp resolution is 1μs). The result of this conversion is referred to as [Z] here.

Finally, the Sync-Master sends the Sync-Slaves a MD_SYNC_SET_REQ message, requesting to set the Time-Stamp of the Sync-Slave to [Z], when the common Bluetooth Clock reaches [X+SyncLeadTime].

If that task is fulfilled by the Sync-Slave, the Sync-Slave responds with a MD_SYNC_SET_RSP message.

At this point, all events reported by the Sync-Slaves can be easily re-converted to a highly precise Sync-Master OS-Clock time (in this hypothetical case, simply by dividing the Time-Stamp of the reported event by 1000000)

➢ **Example variation use case:**

Synchronization of Sync-Slave's Time-Stamp at Bluetooth Clock [B0] to

"03/06/2007 12:00pm GMT UNIX time":

"UNIX time" for "03/06/2007 12:00pm GMT" is 1173182400

That number converted to Time-Stamp μs resolution is:

*1173182400*1000000 = 1173182400000000 = 0x42B00D0357000*

➢ **In this example variation, "Step3" of Example 1 in Section 8.6.1 would look like this:**

Send to Sync-Slaves:

MD_SYNC_SET_REQ

{ TimeStamp_UpdateInformation = 0 (no MD_SYNC_INFO_IND messages)

BluetoothClock_SyncTime = [B0] (request synchronization at [B0])

TimeStamp_SyncTime = 0x42B00D0357000

}

- **In this example variation, Event 1 is detected by Sync-Slave-1 at time [0x42B00F3F8B600]:**

TimeStamp_SyncTime = 0x42B00F3F8B600

"UNIX Time" = TimeStamp_SyncTime/1000000

"UNIX Time" = 0x45ED5A18 = 1173183000 = "03/06/2007 12:10.000000pm GMT"

All that is left for the Sync-Master to do is to re-synchronize the Sync-Slave's Time-Stamp in an interval that is a result of the TimeStamp_NativeAccuracy indicated in the MD_SYNC_CAP_RSP responses of the Sync-Slaves, and the given accuracy requirements.

For calculation examples, see calculation of Interval [I] in "Step4" of Example 1 in Section 8.6.1.

9. Qualification

This section provides an overview of the Bluetooth SIG qualification process as it applies to HDP/MCAP and describes the different processes a device manufacturer may want to consider when attempting to qualify, list and mark a product that supports HDP. Refer to the qualification resources on bluetooth.org for definitive information on this topic.

Two related issues are important in this context:

***Bluetooth Qualification and Listing**:*

This is the Bluetooth Special Interest Group (SIG) program to ensure compliance to the Bluetooth specifications.

***Bluetooth Trademark**:*

The Qualification Program asserts that a manufacturer is compliant with that which is declared. On the other hand, interoperability is ensured by the Member. The Bluetooth SIG provides several tools to help the Member, but many are not mandatory (i.e., UnPlug Fests, extended test suites within the Profile Tuning Suite, Experience Icon program, etc.). A manufacturer is allowed to use the logo if the manufacturer is a member, the product has passed qualification, and it conforms to Brand Book requirements. Members of the Bluetooth SIG are required to sign an agreement that grants to the company permission to use the Bluetooth Trademark for Bluetooth qualified products. If not signed, the Bluetooth Trademark must not be used. The trademark rules are available on the Marketing Resources page of http://www.bluetooth.org.

9.1 BLUETOOTH QUALIFICATION AND LISTING

Since its inception, the Bluetooth SIG has put in place a program to test and improve interoperability amongst Bluetooth products by verifying that the specifications have been properly implemented. The requirements of this program are defined in the PRD [12]. This program makes it possible to test interoperability and qualify Bluetooth designs, Components and Subsystems as well as End Products (see Section 9.1.1 for a definition of the product types). The current version of the PRD contains two different listings; Qualified Design Listing (QDL) and End Product Listing (EPL). These are defined later in this section.

The PRD allows manufacturers to perform the qualification and listing by themselves. The Bluetooth SIG provides a web-based Test Plan Generator (TPG) tool that may be used to create test plans and product listings. An executable Profile Tuning Suite (PTS) is available for download to run profile test cases and generates test reports; although, some of the required lower layer tests (e.g. radio tests) may require a fully equipped and approved external radio lab. The PTS has released support for all HDP and MCAP test cases. In the near future, it is expected that the use of PTS will become mandatory for qualification of HDP and MCAP. Unlike HDP test cases, which are a normative part of the PTS and free for Associate and Promoter members, MCAP test cases are part of a protocol test suite and require purchase of an annual license.

It is also possible to use a recognized external test house and a recognized Bluetooth Qualification Expert (BQE), to perform the testing and listing.

9.1.1 PRODUCT TYPES

A product complying with HDP may be built-up by parts from several vendors. The Bluetooth qualification program defines the following product types:

➢ **End Product Type**

This represents a fully qualified product complying with the advertised profiles. An End Product is generally a finished product that includes a Bluetooth radio such as a blood pressure monitor or pulse oximeter.

➢ **Subsystem Product Type (Controller, Host or Profile Subsystem)**

Subsystems permit partial Bluetooth implementations to be manufactured and sold to end-users (e.g., OS stacks, or USB dongles). Bluetooth Subsystem Products may be combined together without further qualification.

➢ **Component Product Type**

Component Products must be integrated and qualified as part of another product type qualification (typically as an End Product). For a Component Product type, the conformance tests (indicated within the Test Case Reference List (TCRL) as '-C' tests) can be reused but the interoperability tests (the '-I' tests) need to be performed within End Product qualifications.

An implementation may also "subset" (remove) an optional feature to create a compliant subset of an already qualified design.

9.1.2 BLUETOOTH QUALIFICATION TEST TYPES

The main purpose of a Bluetooth qualification is to ensure that the specifications have been properly implemented. Qualification divided into two main parts:

- Tests performed on the radio assuring the Bluetooth radio design complies with Bluetooth specification. These measurements are typically performed in a qualified RF lab. The tests are required for all new physical Bluetooth designs.
- Compliance and interoperability as tested at the protocol and profile level. Protocol testing may be performed using commercially available testers or the PTS. Profile testing is generally performed using the PTS, product versus product testing or other records (e.g. Bluetooth sniffer logs).

Requirements for testing HDP and MCAP are shown within their Test Specifications and the TCRL.

9.1.3 LISTING TYPES

The QDL may be done for Components, Subsystems as well as for End Products. A QDL listed product is assigned a unique number called Qualified Design ID (QD ID) such as B000999. Fees for product listing will depend on the SIG membership level.

The EPL is a more "market oriented" listing of products that use the Bluetooth Technology. Products listed on the EPL refer to the QD ID of the Qualified Design they are using to implement the Bluetooth solution of their product. On the listing, there is a possibility to add sales related information about the product such as pictures, data sheets and links to product web pages. EPL listings are free and, as of PRD 2.1, mandatory for members of the Bluetooth SIG.

Several commercial products may refer to the same Qualified Design (Subsystems or End Product).

A QDL has certain fees associated with it depending on level of Bluetooth membership; however, an EPL is free of charge for Bluetooth SIG members. As of Program Reference Document (PRD) 2.1 [12], the Bluetooth SIG requires the final product to be listed on the Bluetooth SIG EPL website. A member must create a QDL before creating an EPL.

9.2 QUALIFICATION OF AN HDP PRODUCT

There are likely to be several different scenarios for qualifying an HDP product. Some examples are listed in the following sections.

9.2.1 SCENARIO 1- COMPLETELY NEW BLUETOOTH END PRODUCT DESIGN

The new HDP device is a completely new PCB on which an HCI-level Bluetooth chip is mounted. The Bluetooth chip is connected to a host processor running a Bluetooth software stack. The radio level test is always required for such a new radio design.

Typically, the Bluetooth chip that is being integrated is qualified as a Component product type. This means that the qualification process needs to be performed, but for some of the required test cases, a reference to the Component listing may be done (the conformance tests) when integrated into the new design (typically an End Product Type). Integration of a Component can be performed when creating a new Qualified Design of type End Product or Subsystem. The TPG provides a combiner tool that will automatically import the PICS values from the Component into the new TPG project. Where conformance tests (-C tests) were pre-qualified within the integrated Component, those tests do not have to be performed. However, the interoperability tests (-I tests) cannot be inherited from a Component and need to be performed. These will typically govern the amount of testing that will be required.

The same applies to the host stack. If a host stack qualified as a Component or Host Subsystem product type is used (including pre-qualified MCAP/HDP implementation), then only a limited number of test cases are required (by referring to Component qualification).

Sometimes the Device Data Specializations are not included as part of the pre-qualified host stack when sold to customers for integration. For these cases, some interoperability tests specified in the HDP test specification will be required for the particular Device Data Specializations used. Examples of these tests include the SDP Record test and applicable 11073-20601 Data Exchange Protocol tests.

Finally, the product is required to be listed on the QDL and the EPL.

9.2.2 SCENARIO 2 – COMBINATIONS OF SUBSYSTEMS (HCI MODULE)

Under this scenario, the HDP device uses a complete module (up to HCI) that is qualified (on the QDL) as a Controller Subsystem. The module is used on a small carrier board with a host processor running a host stack including HDP and is qualified as a Host Subsystem.

No radio-level Bluetooth qualification tests are required as these are performed for the module.

If the host stack is a Host Subsystem and no changes to the combined Subsystems were performed, no additional qualification is needed. This will be typical for implementations based on a known platform, like Windows or Windows CE, where the stack is a Host Subsystem and the Bluetooth module (e.g. USB dongle or mini-PCI card) is a Controller Subsystem. It is required for the Subsystem combination to be listed on the EPL.

This is typically the case for an HDP device based on a microcontroller and RTOS (Real-Time Operating System), but using a complete HCI module for the Bluetooth interface. It is required for the product to be listed on the QDL and on the EPL.

9.2.3 SCENARIO 3 – COMBINING AN END PRODUCT MODULE WITH HDP/MCAP EMBEDDED WITH A PROFILE SUBSYSTEM.

A module manufacturer supplies a module with an embedded stack that is listed as an End Product Type and the listing includes MCAP. Another manufacturer supplies a Profile Subsystem including HDP. The combination of the End Product Module with the Profile Subsystem as a new product implementation does not require further qualification (note that no modification to the End Product or Subsystem qualifications are allowed without re-assessment for additional testing or listing, refer to the Product Change Checklist [11] for more information).

9.3 MARKING REQUIREMENTS

Members are required to list product implementations sold or distributed to market on the EPL. Refer to the PRD [12] and to the Marking Guide recommendations [13] found on the Bluetooth SIG member web site.

10. Errata

Refer to the member website routinely for HDP/MCAP errata and related test specification errata. It is recommended that all HDP/MCAP errata classified as “ESR Candidate” or “Adopted” be incorporated into designs as appropriate.

11. Regulatory

This is technical advice. It is not legal advice. Consult an attorney to determine if your specific chip, module, implementation, software, or device is subject to government regulation in any market.

HDP is intended for a wide range of devices in different markets, such as healthcare, fitness, and medical diagnostic equipment. If the device is intended, through its design or marketing for the treatment, or diagnosis of a disease, it then may be considered a regulated medical device. For example, if a Heart Rate monitor is marketed for sports or athletic training purposes it is typically not considered a regulated medical device. On the other hand, if the same Heart Rate monitor is marketed to diagnose or treat a disease such as hypertension, it is then considered a regulated medical device.

To protect patient safety and ensure device efficacy, the medical device industry is classified as a regulated industry; that is, the design, manufacturing, distribution, and marketing of devices are regulated by the government of the country in which the device is marketed and sold. The primary goal is to protect public safety for both the device user and the caregiver. At the same time, governments require the manufacturer to demonstrate the effectiveness of the devices and to improve the device performance continuously through post-market surveillance activities. In most cases, there are two requirements: (a) the company or the facility needs to be registered; (b) the product needs to be registered. Different governments exert different levels of control in the above areas. This section will focus on three major markets (USA, EU, and Japan) and discuss Global Harmonization Task Force (GHTF) briefly. Manufacturers should note that during the device registration process, system risk analysis is a key element.

In most regulatory schemes, the final manufacturer of the medical device is responsible for passing regulatory hurdles. The manufacturer of a generic component such as a computer chip or module that has no specific medical functions and is not marketed as a medical device would generally not be regulated. Therefore, a generic HDP/MCAP-enabled radio chip or module is not necessarily a medical device. Once the chip or module is installed into a medical sensor, such as a blood pressure monitor, SpO2, body temperature, weight scale, etc., the chip or module then typically falls under the same medical device requirements as the finished product.

11.1 USA REGULATORY REQUIREMENTS

The Food and Drug Administration (FDA) regulates medical devices and radiation-emitting medical products in the USA. ‘Radiation-emitting’ refers to radiation used for diagnosis, imaging or therapy, not to radio frequency communication, including Bluetooth technology; regulation of radio frequency spectrum usage is addressed in Section 11.6.

The FDA draws regulatory authority from the Food Drug and Cosmetic Act (FD&C Act). To fulfill the provisions of the FD&C Act that apply to medical devices and radiation-emitting products, FDA develops, publishes and implements regulations. These regulations are published annually into the Code of Federal Regulations (CFRs). The CFRs are divided into 50 titles representing broad areas. Medical devices are specifically addressed under Title 21 CFR.

Brief overviews of some of the requirements found in the CFRs are listed below. A full assessment of a specific device and/or use case relative to regulatory requirements is recommended prior to placing the product on the market in the USA. Additional guidance can be found at http://www.fda.gov/cdrh/devadvice.

Establishment Registration (http://www.fda.gov/cdrh/devadvice/341.html)

An owner/operator of an establishment engaged in the manufacture, preparation, propagation, compounding, assembly, or processing of a medical device intended for commercial distribution (marketing) is required to register with the FDA. An initial distributor (or importer) takes first title to the devices imported into the U.S. and further distributes the devices. Initial distributors/importers are required to register. However, they are Not likely required to list the devices that they import.

Registration is renewed annually; there is a fee for registration.

Device Listing (http://www.fda.gov/cdrh/devadvice/342.html)

Most medical device establishments required to register with FDA must also identify to FDA the devices they have in commercial distribution including devices produced exclusively for export. This process is known as medical device listing and is a means of keeping FDA advised of the generic category(s) of devices an establishment is manufacturing or marketing. The regulations for medical device listing are provided in 21 CFR 807.

Listing of a medical device is not approval of the establishment or a device by FDA. Unless exempt, premarketing clearance or premarketing approval is required before a device can be marketed (placed into commercial distribution) in the U.S.

While there is no separate fee for listing your device, device listing is completed as part of the annual registration process.

Product Classification (http://www.fda.gov/cdrh/devadvice/313.html)

Medical devices are classified into one of four classifications: Unclassified, Class I, II, or III. Unclassified devices are automatically considered to be Class III unless proven otherwise and agreed upon by FDA.

The three classes and the requirements that apply to them are:

- Class I General Controls
  - With Exemptions
  - Without Exemptions
- Class II General Controls and Special Controls
  - With Exemptions
  - Without Exemptions
- Class III General Controls and Premarket Approval

Device classification depends on the intended use of the device and upon indications for use. Indications for use can be found in the device's labeling, but may also be conveyed orally during sale of the product, through marketing and sales materials, and / or through marketing agreements (e.g. co-marketing agreement).

In addition, classification is risk based, that is, the risk the device poses to the patient and/or the user is a major factor in the class it is assigned. Class I includes devices with the lowest risk and Class III includes those with the greatest risk.

Premarket Notification / Premarket Approval

FDA authorizes placement of regulated medical devices into commercial distribution through one of three modes: exemption, premarket notification, or premarket approval. These authorizations are roughly related to the classification assigned to the medical device in question. Most Class II and some Class I devices require premarket notification [510(k)] clearance from FDA before the product can be placed into commercial distribution. Class III devices require premarket approval (PMA) from FDA prior to placing the product into commercial distribution.

For more detailed information, refer to the following websites:

Premarket Notification [510(k)]:

http://www.fda.gov/cdrh/510khome.html

http://www.fda.gov/cdrh/devadvice/314.html

Premarket Approval (PMA):
http://www.fda.gov/cdrh/devadvice/pma/

Devices requiring 510(k) clearance or Premarket Approval and placed in commercial distribution without clearance or approval are considered to be adulterated under section 501 of the FD&C Act and cannot be marketed.

Quality System

Furthermore, the FDA requires device manufacturers to follow the "Quality System Regulation" specified in 21 CFR part 820. This regulation covers design, documentation, corrective action, labeling, production, records, etc. Further information can be found at http://www.fda.gov/cdrh/devadvice/32.html.

11.2 EU REGULATORY REQUIREMENTS

Since the time of introducing the single market concept in Europe (January 1993), efforts to support free trade across borders between European Union (EU) member states and countries aligned via the European Economic Area (EEA). These efforts have been balanced with the desire to protect public safety through implementation of regulations recognized by all countries taking part in the single market concept. Key to the single market is the concept of free movement of goods. To remove barriers to free movement, two regulatory instruments were developed focusing upon medical devices: the New Approach to device regulation and the Global Approach to conformity assessment. These are described in associated Directives.

The New Approach directives define content and structure framework to guide the legislative approach based, legally, under Article 95 of the European Council Treaty. The New Approach is not limited to healthcare devices. Devices that are granted market access (placement on the market) using the New Approach are distinguished by the European Conformity (CE) Mark.

About 20 Directives came into force during the mid-1980s. Of these, only a few are of interest to healthcare related devices. The following Directives govern most healthcare devices; specific devices and/or use cases should be reviewed in detail to ensure all Directives are applied appropriately prior to placement of the device(s) on the market in the EU. The following Directives establish the base medical device requirements (http://ec.europa.eu/enterprise/medical_devices/legislation_en.htm):

- Active Implantable Medical Device Directive (AIMDD)
- Medical Device Directive (MDD) [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31993L0042:EN:NOT]
- In Vitro Diagnostic Devices Directive (IVDD) [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31998L0079:EN:NOT]

Directives that are also of interest to any intentional radio frequency (RF) emitter:

- Radio and telecommunications terminal equipment (RTTE) [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31999L0005:EN:NOT]
- Electromagnetic compatibility [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32004L0108:EN:NOT]

Other standards-receptive directives of interest:

- Waste electrical and electronic equipment (WEEE) [http://eur-lex.europa.eu/LexUriServ/site/en/consleg/2002/L/02002L0096-20031231-en.pdf]
- Packaging and packaging waste [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31994L0062:EN:NOT]

The target audiences for these Directives are the EU Member State Competent Authorities (CAs), not individuals or manufacturers. Each CA must establish how directive requirements are implemented (transposed) in its Member State's national laws. Additional requirements can, and often are, introduced into the laws if they do not conflict with any EU legislation requirement. These regulatory requirements are supplemented by a series of MEDDEV guidelines that reflect consensus views of authorities and industry. These are published in the EC's Industry Enterprise medical device website.

Oftentimes, additional registration-like information is needed in order for a device to be placed onto a reimbursement formulary. For example, the UK requires additional assessment of product prior to distributing reimbursement funds (http://www.mhra.gov.uk/index.htm). The Nordic countries also require additional information / testing prior to distributing reimbursement funds (http://www.uib.no/isf/noklus/english.htm). Activities associated with reimbursement occur after registration approval of the CE mark.

A key to the application of Directives is an understanding of essential requirements. Essential requirements are set forth in annexes to the Directives and include all criteria necessary to achieve a Directive's objective. Medical devices must fulfill the corresponding directive's essential requirements. The evidence provided to show fulfillment of the requirements are typically in the form of compliance to harmonized standards. Technical product specifications are not included in the directives, but are established separately in harmonized standards. Products manufactured in accordance with harmonized standards are presumed to conform to the Essential Requirements listed in the associated Directive.

There are two primary avenues to legally place product on the market in Europe: conformity assessment through a third-party or conformity assessment by the manufacturer (self-declaration). Conformity assessment is the process by which a determination of compliance to the Essential Requirements is accomplished. Successful conformity assessment indicates that all essential requirements have been met and that the product can be placed on the market in Europe bearing the CE mark.

Some products require a conformity assessment to be performed by a third party; a Notified Body. Each Notified Body is assigned a unique 4-digit number. Product that has been assessed by a given Notified Body and has passed Conformity Assessment bears the CE + 4-digit number of the Notified Body. For example, CE0088 indicates LRQA; CE0123 indicates TUV.

Products not requiring third-party conformity assessment are said to be self-declared. These products must also meet all Essential Requirements, but the manufacturer performs the assessment. These products are marked with only "CE;" no 4-digit number.

The following websites can provide information that is more detailed:

- Guide to the New Approach (Directives and compliance to Essential Requirements):
  - http://ec.europa.eu/enterprise/newapproach/legislation/guide/index.htm
- General information regarding medical devices in the EU:
  - http://ec.europa.eu/enterprise/medical_devices/index_en.htm
- Information specific to medical devices:
  - http://ec.europa.eu/enterprise/medical_devices/meddev/meddev_index_en.htm

11.3 JAPAN REGULATORY REQUIREMENTS

In Japan, the devices are regulated by the Health & Labor Department under the Pharmaceutical Affairs Law (PAL).

Medical device manufacturers or distributors need to have a certified Quality Management System (QMS) license with a five-year expiration/renewal period. The QMS is very similar to ISO 13485:2003. The website, http://asia.bsi-global.com/Japan+MedicalDevice/ordinance.xalter, is quite helpful. Based on the risk level of the medical device, there are different requirements for device approval or/and certification.

11.4 GLOBAL HARMONIZATION TASK FORCE

The Global Harmonization Task Force (GHTF) is a voluntary informal organization formed in 1992 by EU, USA, Canada, Japan and Australia. GHTF does not have regulatory power over medical device manufacturers, but rather serve as a forum for regulators and industry to collaborate on regulatory issues. The GHTF generates guidance documents for governments to adopt. Canada, Australia and many Asian countries have committed to GHTF guidance documents including:

Study Group 1 - Pre-market Evaluation

Study Group 2 - Post-Market Surveillance/Vigilance

Study Group 3 - Quality Systems

Study Group 4 - Auditing

Study Group 5 - Clinical Safety/Performance

Refer to http://www.ghtf.org for more information.

11.5 COMPLIANCE WITH AIRLINE REQUIREMENTS

Although Bluetooth wireless technology has been allowed for use on a number of airlines, some airlines do not permit this. It is recommended that an HDP device be equipped a radio power switch or flight safe mode in case power down is required and this should be documented in the user manual of the device.

11.6 RADIO FREQUENCY REGULATORY COMPLIANCE

Bluetooth wireless technology operates in an unlicensed frequency band of 2.4 GHz. National (e.g. US FCC) and transnational organizations (e.g. European CEPT) establish regulations for the use of this RF spectrum. The Bluetooth Regulatory committee has collected current worldwide regulatory information and posted it for members at https://www.bluetooth.org/Technical/Regulatory/overview.htm.

12. RF Coexistence

Bluetooth wireless technology operates in an unlicensed frequency band of 2.4GHz, which is reserved for Industrial, Scientific and **Medical** applications (ISM band). This band is largely common throughout the world, but this is not universal and there some local rules that need to be followed (refer to https://www.bluetooth.org/Technical/Regulatory/overview.htm for more information). Devices operating in this band must be able to cope with interferences from other devices emitting on the same frequencies. Many other devices common in homes and offices operate in this band, including cordless phones and microwave ovens.

Potential Bluetooth-on-Bluetooth interference is largely handled by a frequency-hopping algorithm that causes a relatively random hopping pattern determined by the Bluetooth Address and clock timing of the master of the piconet. The most problematic interference for Bluetooth devices tends to come from 802.11b/g/n radios (802.11a radios operate entirely in a different frequency band) either in the same device (collocated) or in nearby devices. Managing and mitigating this interference is termed ‘coexistence’ in the Bluetooth specifications.

In Bluetooth wireless technology, coexistence between neighboring devices is facilitated through a Core feature called “Adaptive Frequency Hopping” (AFH). AFH was introduced in version 1.2 of the Bluetooth Core Specification. It consists of two Bluetooth devices in a connection agreeing to use a frequency hopping pattern that avoids areas of frequency that one or both radios has determined is or may be in-use by removing them from the hopping pattern. AFH does not itself affect data throughput, as every available slot is still available to be used, however this causes the piconet to operate on less than the full ISM band and is more vulnerable to Bluetooth-on-Bluetooth interference (i.e. a smaller hop set will result in more collisions). In addition, any fixed frequency interference (such as a further Wi-Fi device) will likely impact data throughput before the AFH channel assessment algorithm can determine that a new interference source is present.

How each radio determines which frequencies should not be used is up to system implementers and usually involves one or more of the following methods:

1. Communication between the local 802.11 driver and the Bluetooth stack to determine what 802.11 channels are being used and mapping those into Bluetooth channels to avoid transmitting on the same frequencies at the same time
2. Noise detection (RSSI) of the Bluetooth channels

3. Error detection during existing Bluetooth communications

In addition to AFH, many Bluetooth controllers will also implement some form of proprietary hardware communication between the Bluetooth MAC/PHY and the 802.11 MAC/PHY, and/or communication between the Bluetooth stack and the 802.11 driver. This type of coexistence is only helpful for devices with collocated radios.

In summary, coexistence is managed by the Bluetooth stack and MAC/PHY and the 802.11 driver; the profile does not need to be aware of MAC/PHY. Support is provided by the hardware and/or stack vendors. The following features assist in mitigating coexistence issues:

- **Adaptive Frequency Hopping (AFH)**
  - Master role for Adaptive Frequency Hopping (AFH) should be supported.
  - Master role for classification of channels should be supported.
  - Slave assessment of interference should be supported if the application is such that the slave may be located relatively far from the master and therefore might be experiencing different interference.
- **Enhanced Data Rate (EDR)**
  - EDR allows for higher over-the-air data transmission. For good coexistence practices with other devices operating in the same frequency band, EDR should be used to lower the duty cycle thus reducing the probability of collisions and retransmissions. However, in cases where the devices transmit very little data, the benefit will be minimal. In fact, if the amount of data transmitted is 2 octets or less per packet, EDR actually increases the time.
- **Alternate MAC/PHY (AMP)**
  - The use of an AMP with HDP means that the HDP data would be exchanged directly on another radio system. At the time of this writing, the only supported AMP is 802.11. If both ends of the HDP link include support for 802.11a, which operates in the 5.1 GHz frequency band, its use would greatly reduce the likelihood of interference on the Bluetooth link. This can be achieved by the use of the Preferred Channels parameter in the AMP_ASSOC when configuring the AMP. The use of an 802.11b/g AMP, which is in the same frequency band as Bluetooth, would overlap and not offer this benefit.

13. References

[1] Specification of the Bluetooth Health Device Profile (HDP), version 1.0

[2] Specification of the Bluetooth Multi-Channel Adaptation Protocol (MCAP), version 1.0

[3] Bluetooth Core Specification 2.0 + EDR or 2.1 + EDR with Volume 3, Part A of Core Specification Addendum 1 or Bluetooth Core Specification 3.0 + HS or later versions of the Bluetooth Core Specification

[4] IEEE Std 11073-20601 ™- 2008 Health Informatics - Personal Health Device Communication - Application Profile - Optimized Exchange Protocol - version 1.0 or later

[5] IEEE Std 11073-104xx ™- 2008 Health informatics - Personal health device communication - Device specializations - version 1.0 or later

[6] Bluetooth Device Identification Specification, version 1.3 or later

[7] Bluetooth Master/Slave Communications and Sniff/Sniff Subrating Modes; Bluetooth SIG

[8] Secure Simple Pairing Whitepaper; Bluetooth SIG

[9] Bluetooth low energy wireless technology Marketing Requirements Document; Bluetooth SIG

[10] 1003.1-2001/COR 1-2002 IEEE Standard for Information Technology - Portable Operating System Interface (POSIX) (Informative)

[11] Product Change Checklist (PCC), Bluetooth SIG (https://www.bluetooth.org/Technical/Qualification/home.htm)

[12] Qualification Program Reference Document; Bluetooth SIG

[13] Product Marking Guide; Bluetooth SIG

[14] Discovery White Paper: Service Discovery Applications; Bluetooth SIG

14. Acronyms and Abbreviations

Abbreviation or Acronym	Meaning
AFH	Adaptive Frequency Hopping
APDU	Application PDU<Description>
BQE	Bluetooth Qualification Expert
CE	European Conformity
CEPT	European Conference of Postal and Telecommunications Administrations
CSP	Clock Synchronization Protocol
ECG	Electrocardiogram
EPL	End Product Listing
FCC	(US) Federal Communications Commission
FDA	Food and Drug Administration
GAP	Generic Access Profile
GHTF	Global Harmonization Task Force
HCI	Host-Controller Interface
HDP	Health Device Profile
ISM	Industrial, Scientific and Medical
KB	Kilobyte
L2CAP	Logical Link Control and Adaptation Protocol
MCAP	Multi-Channel Adaptation Protocol
MCL	MCAP Communications Link
MDD	Medical Device Directive
MDEP	MCAP Data End Point
MDL	MCAP Data Link
MRD	Marketing Requirements Document
PC	Personal Computer
PCB	Printed Circuit Board
PDA	Personal Digital Assistant
PDU	Protocol Data Unit
POTS	Plain Old Telephone Service

PRD	(Bluetooth Qualification) Program Reference Document
PTS	Profile Tuning Suite
QDL	Qualified Design Listing
QoS	Quality of Service
RF	Radio Frequency
SDP	Service Discovery Protocol
SPP	Serial Port Profile
WG	Working Group