Generic Health Sensor Design and Implementation Guide

***Bluetooth*® Informational Publication**

- **Version:** v1
- **Version Date:** 2024-03-19
- **Prepared By:** Medical Devices Working Group

If you are a SIG member and have any specific requested changes to this document, please submit an erratum in Jira:
https://bluetooth.atlassian.net/jira/software/c/projects/ES/issues

Abstract:

The Generic Health Sensor (GHS) Design and Implementation Guide guides implementers of health sensor devices and mobile application developers of applications collecting health data from such sensor devices. This publication covers various non-trivial aspects of GHS such as where to find all relevant information and how to design and implement a sensor or application in a generic, reusable manner. This publication also places the GHS specifications in the broader context of an interoperable healthcare ecosystem.

Version History

Version Number	Date (yyyy-mm-dd)	Comments
v1	2024-03-19	Approved by the Bluetooth SIG Board of Directors.

Acknowledgments

Name	Company
Erik Moll	Koninklijke Philips N.V.

Contents

1 Introduction

This guide aims to help implementers of health sensor devices and mobile application developers of applications collecting health data from such sensor devices with their implementations of the GHS Service [19] and Profile [20]. This publication covers various non-trivial aspects of the GHS specifications such as:

- Where to find all relevant information from the Bluetooth SIG and the referenced standards from other organizations.
- How to design and implement a sensor or application in a generic, reusable manner.

This publication also places the GHS specifications in the broader context of an interoperable end-to-end (E2E) healthcare ecosystem and provides an overview of how to upload GHS data to healthcare cloud services.

This is not a Bluetooth specification, therefore, the established Bluetooth SIG specification language conventions for use of the words ***shall***, ***shall not***, ***must***, ***should***, ***should not***, ***may***, and ***can*** do not apply to this document.

1.1 Scope

The following aspects will be covered in this publication:

- How does GHS fit into an E2E healthcare system?
- Which Bluetooth SIG publications and associated documents are relevant to GHS and where can these be found?
- Where are the IEEE Medical Device Communication (MDC) codes for an existing sensor type and its generated data defined?
- How to request IEEE MDC codes for new sensor types and new types of generated data?
- Which existing open-source code repositories can be used when implementing GHS?
- What design choices should be considered when implementing the GHS protocol to upload data?
- How to make GHS implementations reusable for other sensor device types?
- How to upload GHS data to an HL7® FHIR® standard [1] server in the cloud?
- How to handle the privacy and security aspects of medical data in a GHS implementation?

In covering these aspects, a small number of common sensor types will be used as examples, such as a blood pressure cuff, a heart rate monitor, and a pulse oximeter as example sensor devices.

The intended audience of this publication consists of designers and developers of healthcare sensor devices and of associated applications that typically run on mobile devices, such as mobile phones or tablets, or sometimes run on dedicated hardware.

2 How does GHS fit into an E2E healthcare system?

GHS is based on the IEEE 11073-10206™ [3](*) standard that defines an abstract and generic information content model (ACOM) of the data that a personal health device (PHD) provides to a user of the device. GHS defines a concrete implementation of ACOM on Bluetooth GATT.

GHS can be used to implement common PHD sensor types such as thermometers, weighing scales, blood pressure monitors, pulse oximeters, heart rate and ECG monitors, and less common sensor types, such as spirometry devices, medication monitors, and more, by using the IEEE 11073-10101™ [2] nomenclature standard to identify observations and their attributes.

ACOM is illustrated in Figure 2.1. Appendix A provides more details on the ACOM classes.

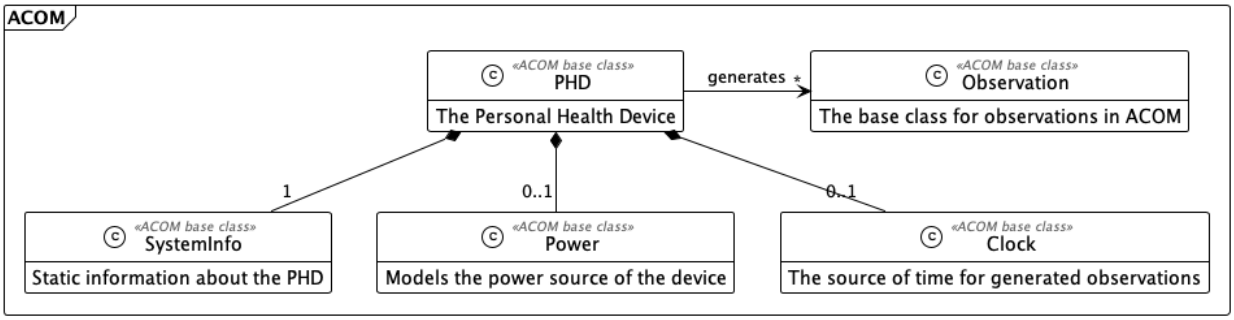

Figure 2.1: ACOM overall model

ACOM objects can easily be transcoded into HL7® Fast Healthcare Interoperability Resources (FHIR®) [1] resources. HL7® FHIR® provides a widely used model and exchange protocol for healthcare data. Supported resource types include Observation, Patient, Device, Procedure, Medication, Diagnostic Report, Care Plan, and much more.

Support of FHIR® is strongly encouraged by the Office of the National Coordinator for Health Information Technology (ONC) [4] for Electronic Health Record (EHR) systems for a growing set of data as defined in the United States Core Data for Interoperability (USCDI) [5].

ACOM Observations can be uploaded as FHIR® Observation resources to a FHIR® server and can be used for patient monitoring, diagnostic purposes, and care plan definitions. The types and fields of GHS observations map one-to-one to FHIR® Observation data elements. FHIR® supports the use of IEEE 11073-10101 [2] nomenclature codes. Details can be found in Section 8.

The typically static device information from the ACOM SystemInfo object can be uploaded as a FHIR® Device resource. In a FHIR® server, devices can be assigned to patients or users and can be referenced from observations.

(*) The IEEE standards or products referred to in this document are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated. IEEE publications are available from the Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).

The Power information, such as the battery level and remaining time, can be uploaded as FHIR® Observation resources. The same holds for the Clock; the current time can be uploaded together with the client's time as a Coincident Time Stamp Observation resource. Power information and Clock data are used to manage the device instead of the patient.

See Figure 2.2:.

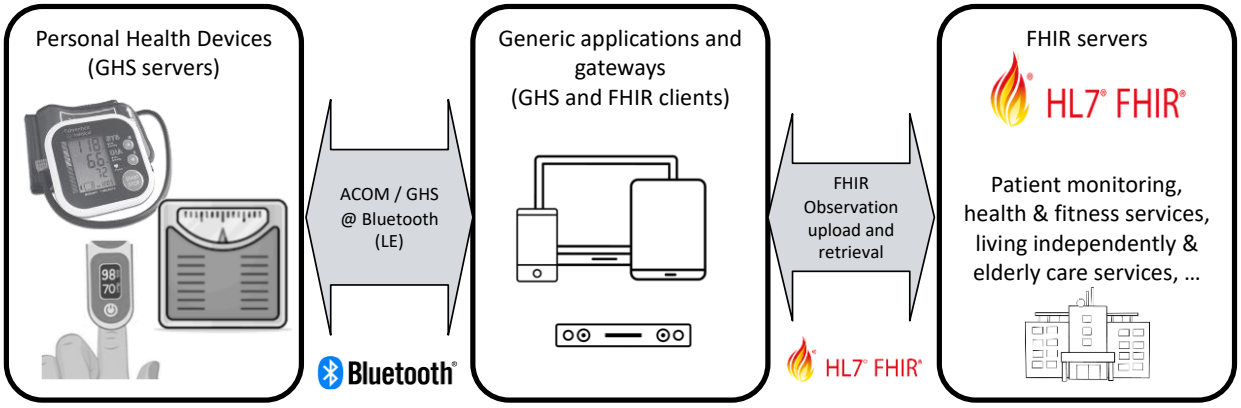

Figure 2.2: E2E healthcare system overview

3 Which Bluetooth SIG publications are relevant to GHS?

The Bluetooth SIG develops and maintains Bluetooth specifications and supporting documents and makes them available through its website [6].

The following Bluetooth specifications are relevant for GHS:

- Generic Health Sensor Service (GHSS) specification – defines the format and exchange of ACOM observations using GATT.
- Generic Health Sensor Profile (GHSP) specification – defines the set of services and their usage to implement an ACOM PHD as a GATT server and defines the requirements for a GHS client.
- Elapsed Time Service (ETS) specification – defines the exchange of clock and/or current time information for an ACOM PHD over GATT.
- Device Information Service (DIS) specification – defines the static information that describes a PHD as a set of GATT characteristics. From DIS Version 1.2 and later, DIS includes the Unique Device Identifier (UDI) for medical devices. This characteristic provides a unique, regulated device identifier managed by an authority such as the United States Food and Drug Administration.
- Battery Service (BAS) specification – defines the exchange of battery information over GATT.
- User Data Service (UDS) specification – defines how to support multiple users on a GHS server.
- Reconnection Configuration Service (RCS) specification – defines how to update connection parameters of a GHS server and supports a handshake mechanism to disconnect.
- Bluetooth Core Specification (CS) – defines GATT/ATT/GAP/SM, the foundation on which GHS is built.
- Core Specification Supplement (CSS) – defines common ATT/GATT error codes and the data types used in Advertisement Data and Scan Response Data.

The following supporting documents are relevant for GHS:

- GATT Specification Supplement (GSS) [7] – contains the definitions of all re-usable GATT characteristics.
- Assigned Numbers document [8] – contains the UUID values for the services and the characteristics used by GHS; this document is available in PDF and YAML format.

Figure 3.1 illustrates the use of these specifications for GHS.

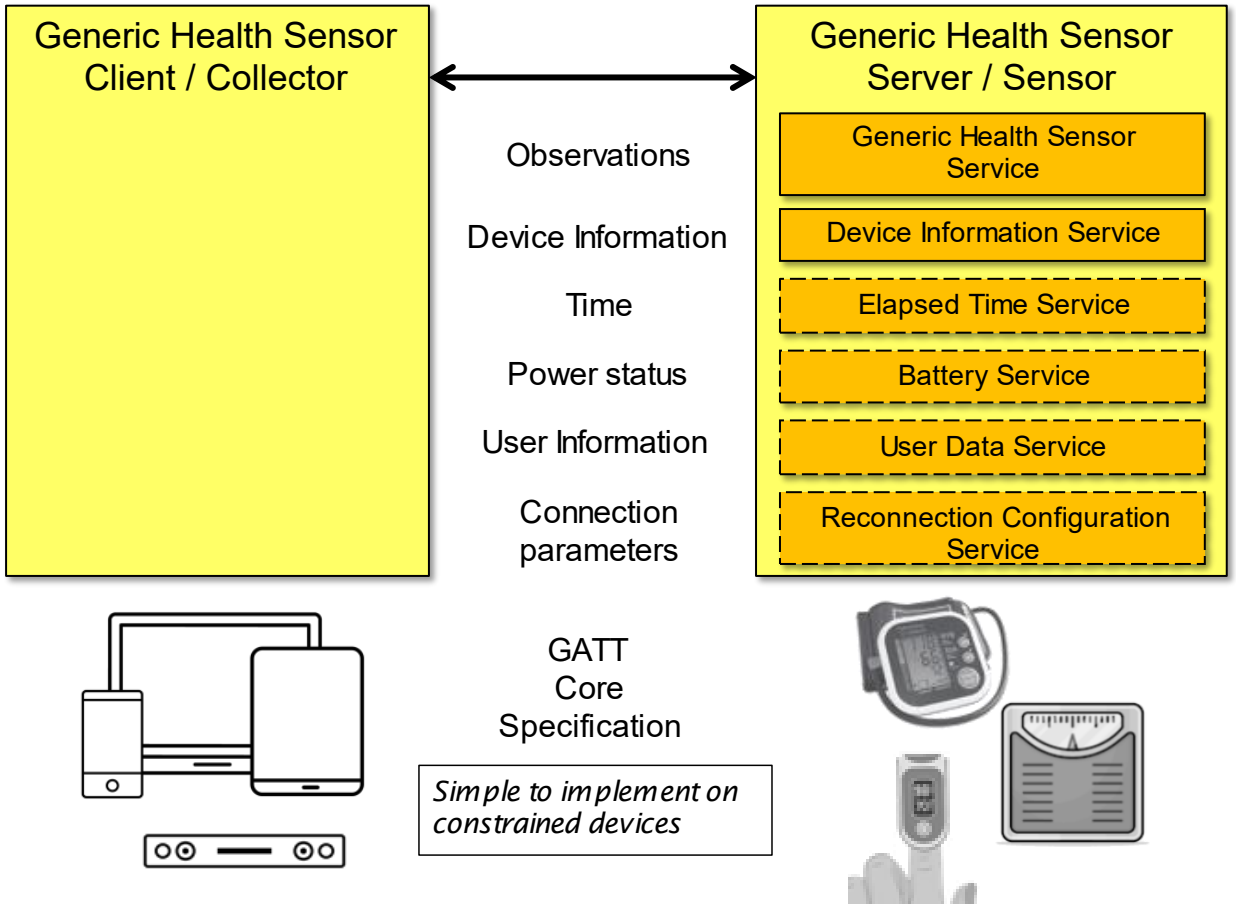

Figure 3.1: GHS usage of Bluetooth specifications

4 Where are the IEEE MDC codes for an existing sensor type and its generated data defined?

The IEEE 11073-10101 Medical Device Communication (MDC) codes are codes that uniquely define a medical concept such as an observation type, a location on the human body, or a sensor type.

MDC codes are 32-bit values that consist of a 16-bit partition code and a 16-bit term code. Partitions contain all codes for a specific purpose. For example, partition 4 contains the codes for dimensions or units of measurement. Each code also has a unique textual name, the Reference Identifier or RefId and a longer description of its purpose. For example, the MDC code for degrees Celsius is 4::6048 (hexadecimal 0x000417A0, decimal 268192). In some cases, when the partition is known or fixed, the term code suffices to define a concept. In GHS, a numeric observation value uses a 2-octet term code to identify the unit of measurement.

The MDC codes are defined by the IEEE 11073 Standards Committee (SC). This SC has two working groups, one for Personal Health Devices (PHD) and one for Point of Care Devices (PoCD). The codes defined by these groups are collected and published in the IEEE 11073-10101 nomenclature standard [2].

The HL7® FHIR® standard [9] includes an overview on the use of MDC codes [9].

Appendix B gives an overview of the device specializations, observation types, and the corresponding MDC codes used for common sensor device types.

There are different ways to find out which MDC codes are to be used for a given sensor device type, for observations the sensor device generates, and for the attributes of these observations:

- The ACOM standard, IEEE 11073-10206 [3], contains a compact definition of the information model of the most commonly used vital signs sensor devices, including the MDC codes (RefIds only) used in these specializations:
  - Pulse oximeter
  - Basic ECG or heart rate
  - Blood pressure monitor
  - Thermometer
  - Weight scale
- Obtain the corresponding IEEE 11073-104xx standard [10] and use the nomenclature codes as defined in the Domain Information Model in Chapter 6 of that standard. The following IEEE 11073-104xx standards are available:
  - IEEE 11073-10404™ Pulse Oximeter
  - IEEE 11073-10406™ ECG
  - IEEE 11073-10407™ Blood Pressure Monitor
  - IEEE 11073-10408™ Thermometer

- IEEE 11073-10413™ Respiration Rate Monitor
- IEEE 11073-10415™ Weighing Scale
- IEEE 11073-10417™ Glucose Meter
- IEEE 11073-10418™ INR (blood coagulation)
- IEEE 11073-10419™ Insulin pump
- IEEE 11073-10420™ Body composition analyzer
- IEEE 11073-10421™ Peak flow monitor
- IEEE 11073-10422™ Urine Analyzer
- IEEE 11073-10424™ Sleep Apnoea Breathing Therapy Equipment (SABTE)
- IEEE 11073-10425™ Continuous Glucose Monitor
- IEEE 11073-10426™ Home Healthcare Environment Ventilator (draft)
- IEEE 11073-10427™ Power status monitor of personal health devices
- IEEE 11073-10429™ Spirometry
- IEEE 11073-10441™ Cardiovascular fitness and activity monitor
  - Includes a Step Counter and an Activity Monitor profile (further specialization).
- IEEE 11073-10442™ Strength fitness equipment
  - Supports reporting exercise sessions on various types of fitness equipment.
- IEEE 11073-10471™ Independent living activity hub
  - Supports reporting of various types of sensors in and around the home environment to assist patients or users with living independently. Examples include a Fall Sensor, a Motion Sensor, a Switch Use Sensor, a Current Temperature Sensor, a Temperature Alert Sensor, a Utility Usage Sensor (Gas, Electricity, Water), etc. These sensors can also report their location in a home as supplemental information.
- IEEE 11073-10472™ Medication monitor

- Obtain the IEEE 11073-10101 standard, check Annex A.7 for the sensor type that is being implemented, and use the metric codes for the GHS Observation type values as needed.
- Use the SIG's Personal Health Devices Transcoding white paper [11], which is available to all Bluetooth SIG members. This document covers the following sensor device types:
  - Thermometer
  - Heart Rate
  - Blood Pressure Monitor
  - Glucose Meter
  - Weight Scale

- Continuous Glucose Monitor
- Pulse Oximeter

- Use the Rosetta Terminology Mapping Management System (RTMMS) [12] to find numeric values of specific MDC codes:
  - RTMMS is an online database of all current and draft MDC codes and is maintained by NIST. After registration, the database can be searched in various ways.
  - As an example, Figure 4.1 captures the results from looking for “PRESS” in the MDC codes to find codes for use with a blood pressure monitor. The result includes both the RefId and partition-code::term-code, as well as the decimal 32-bit value.

Enter search terms for individual columns...or Switch to 'Search All'

	Link	RefID	Term Code	CF Term Code	Description/Definition
		PRESS	2::		
1		MDC_PRESS_BLD	2::18944	150016	Pressure of the blood
2		MDC_PRESS_BLD_SYS	2::18945	150017	Pressure of the blood at the systolic phase
3		MDC_PRESS_BLD_DIA	2::18946	150018	Pressure of the blood at the diastolic phase
4		MDC_PRESS_BLD_MEAN	2::18947	150019	Pressure of the blood as computed by averaging on one cycle
5		MDC_PRESS_BLD_NONINV	2::18948	150020	Pressure of the blood, obtained noninvasively (i.e., fingertip) – this refers normally to the measurement of diastolic and systolic together
6		MDC_PRESS_BLD_NONINV_SYS	2::18949	150021	Pressure of the blood, obtained noninvasively (i.e., fingertip), at the systolic phase
7		MDC_PRESS_BLD_NONINV_DIA	2::18950	150022	Pressure of the blood, obtained noninvasively (i.e., fingertip), at the diastolic phase
8		MDC_PRESS_BLD_NONINV_MEAN	2::18951	150023	Pressure of the blood, obtained noninvasively (i.e., fingertip), as computed by averaging on one cycle
9		MDC_PRESS_BLD_NONINV_CTS	2::18952	150024	Pressure of the blood recorded continuously and noninvasively (i.e., fingertip)
10		MDC_PRESS_BLD_NONINV_SYS_CTS	2::18953	150025	Pressure of the blood, obtained continuously and noninvasively (i.e., fingertip), at the systolic phase

Figure 4.1: Example output from RTMMS

5 How to request IEEE MDC codes for new sensor types and new types of generated data?

The IEEE 11073™ Standards Committee manages the assigned MDC codes and publishes them in the IEEE 11073-10101 standard. To get a new publicly assigned code, send a request to this committee, using the information on the committee's home page [13].

The IEEE PHD (Personal Health Devices) working group and the IEEE PoCD (Point of Care) working group fall under this Standards Committee and work together to maintain the assigned MDC codes.

The IEEE 11073-10101 standard has defined areas for private codes that can also be used temporarily until a public code is assigned:

- Partition 1024 is the Private Partition and provides space for 64K of private codes.
- Other assigned partitions reserve the range 0xF000–0xFFFF for private codes.
- A manufacturer should maintain the privately assigned codes and make sure that they are used consistently in this manufacturer's products.

6 Which open-source code repositories can be used when implementing GHS?

There are a few open-source software (OSS) implementations of GHS. Two of them are discussed briefly as examples in the following subsections.

6.1 LNI's GHS implementations

Brian Reinhold from Lamprey Networks, Inc. (LNI) participated in the development of the GHS specifications and implemented a GHS server on a Nordic nRF52840 development board and nRF51 development kits. The repository is located on GitHub, https://github.com/brianreinhold/NordicPHDs [14], and also contains an Android application (as a set of Android Package Kits (APKs)) for an Android GHS Client. This software comes with a Massachusetts Institute of Technology (MIT) license.

This repository uses the #define mechanism for MDC codes, with the partition number in hex and the code within the partition in decimal notation. This allows easy copying of codes from the IEEE standards (see Table 6.1).

```
#define MDC_PRESS_BLD_NONINV (0x20000 + 18948)

#define MDC_PRESS_BLD_NONINV_SYS (0x20000 + 18949)

#define MDC_PRESS_BLD_NONINV_DIA (0x20000 + 18950)

#define MDC_PRESS_BLD_NONINV_MEAN (0x20000 + 18951)

#define MDC_SAT_O2_QUAL (0x20000 + 19248)

#define MDC_TEMP_BODY (0x20000 + 19292)

#define MDC_TEMP_TYMP (0x20000 + 19320)

#define MDC_PULS_OXIM_PERF_REL (0x20000 + 19376)

#define MDC_PULS_OXIM_PLETH (0x20000 + 19380)

#define MDC_PULS_OXIM_SAT_O2 (0x20000 + 19384)
```

Table 6.1: #define MDC codes

This client builds on LNI's Health@Home application and also supports other Bluetooth SIG services and profiles and FHIR® or PCD-01 data upload. See https://www.lnihealth.com/products/healthhome-android-hub [15].

6.2 Philips GHS implementations

Philips implemented a GHS client and server and made it available on GitHub to the public:

- https://github.com/philips-labs/BLE-GHS-Client-Example [17]
- https://github.com/philips-labs/BLE-GHS-Server-Simulator [18]

The Philips repositories come with an MIT license.

The Philips code is based on BLESSED [16]. BLESSED is a compact Bluetooth® Low Energy library for Android 5 and higher that is designed to make working with Bluetooth® Low Energy on Android easy. BLESSED takes care of many aspects of Bluetooth so that the application code doesn't need to take care of them.

These applications support all features from the GHS Service and Profile specifications. A screenshot of the client and server application is shown in **Error! Reference source not found.**.

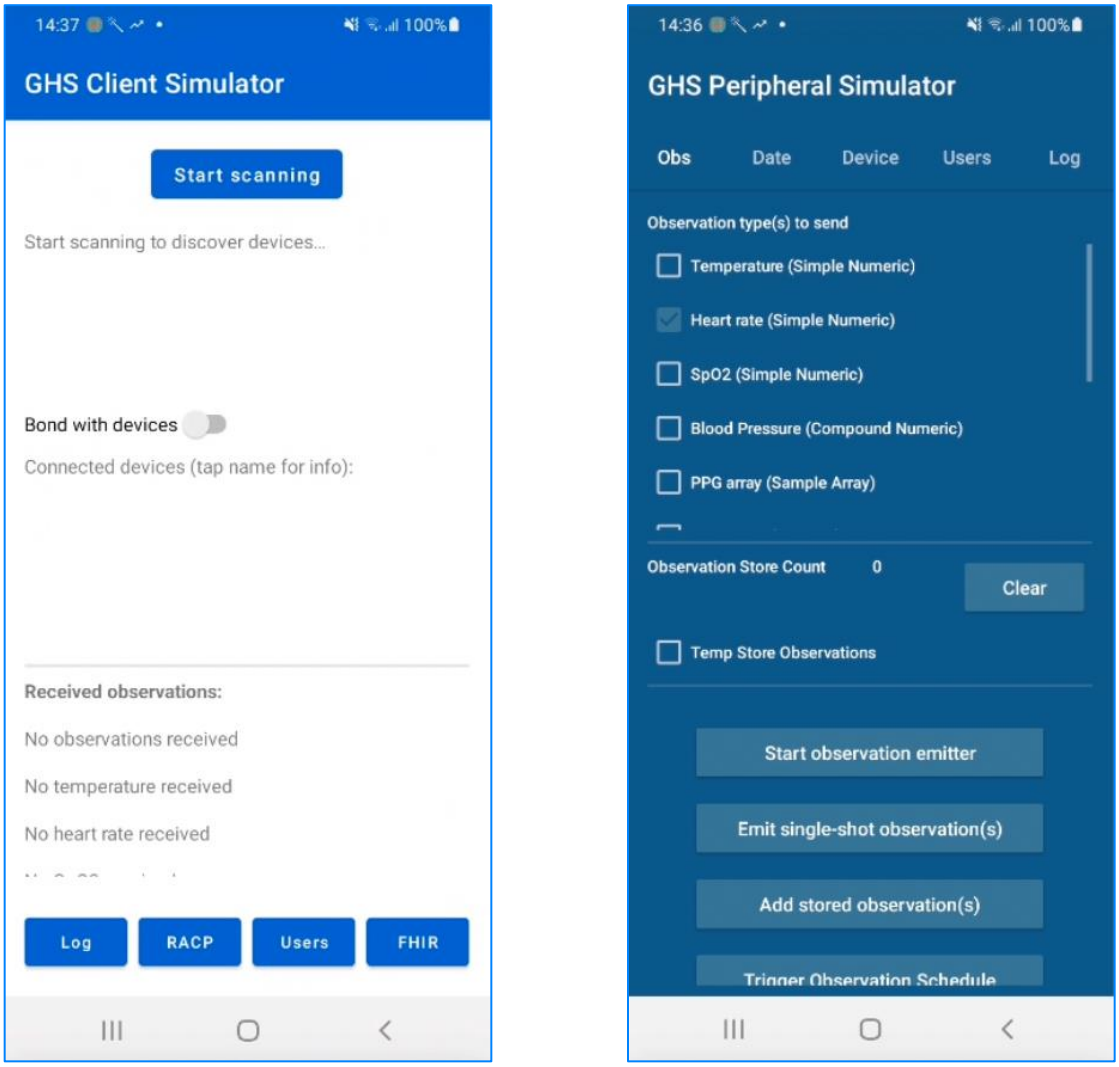

Figure 6.1: Example GHS applications

7 What design choices to consider when implementing the GHS protocol to upload data?

7.1 Decide on supporting live, temporarily stored, stored observations, or a mix

In GHS, a PHD may use different reporting schemes for observations. An implementer can choose any of these schemes:

- Live observations are generated when a client (mobile application) is connected to a sensor device and these live observations are pushed by the sensor device to the connected client. After transmission, the sensor device can delete all data associated with the observation. Live observations do not need a time stamp. The client can add a time stamp as needed, based on its own clock.
- Temporarily stored observations are generated when no client is connected to a sensor device. After an observation is generated, the sensor device starts advertising to indicate it has new observations. Once a client connects, the sensor device transmits the new observations to the connected client. After transmission, the sensor device can delete all data associated with these observations. Temporarily stored observations do need a time stamp. If the client considers the time on the device inaccurate, then the client can adjust the time stamps of the observations. The client should also update the clock of the sensor device in such situations.
- Stored observations are generated and stored on the sensor device for later retrieval by one or more clients using Record Access Control Point (RACP) procedures. When new observations are generated, the sensor device may start advertising when new observations are stored or may allow periodic retrieval by one or more clients.

If the sensor device does not have a clock, then using live observations is the only option.

If the sensor device typically communicates with only one client, then use of temporarily stored observations is a logical choice.

If the sensor device maintains a history of all observations that can be retrieved by multiple clients, then stored observations should be used.

When using stored or temporarily stored observations, the implementer should have a mechanism to handle running out of memory. One option is to support a delete procedure on the RACP and have the client delete older observations before running out of memory. Another option is to implement an automatic deletion of the oldest observations on the server.

An implementer may choose to report certain observations only as live observations during a connection and not store them. An example would be the intermediate cuff pressure reports from a blood pressure monitor that have little value once the measurement is complete.

More information can be found in GHSS [19] Section 2.7.

7.2 Decide on using GATT indications, notifications, or a mix

Implementers must also decide on the use of GATT indications or notifications for the transfer of live, temporarily stored, and stored observations.

Notifications have higher throughput because their reception is not acknowledged, and therefore they could get lost in the software stack before reaching the application.

The sensor device implementer must choose which mechanism or mechanisms to support for live and stored observations. See GHSS [19] Section 2.7.

7.3 Decide on multiuser support using UDS

GHSP includes UDS for multi-user devices. UDS exposes user data and supports a user consent mechanism to access user data. If a device is intended to be used by more than one person, UDS should be implemented. This will add another layer of complexity to the device because it will have to maintain a database of observations per user.

Also, when uploading data to a FHIR® server, some mapping is needed from the user ID in UDS to a patient identifier in the FHIR® observations (see Section 8.6).

The GHS client needs to maintain a mapping from the User ID in UDS to a FHIR® Patient Identifier. to the method for getting these patient identifiers is out of scope for the Bluetooth SIG GHS specifications.

7.4 Timestamping and clock features

GHSP includes ETS to support a clock and time stamping of observations. A GHS server supports a fixed type of time in a fixed resolution. Both are static during the lifetime of the device. The supported types of time are:

- Coordinated Universal Time (UTC) time, with or without TZ/DST offset
- Local time, with or without TZ/DST offset
- Tick counter

The supported resolutions are:

- 1 second
- 100 milliseconds
- 1 millisecond
- 100 microseconds

When observations are stored or temporarily stored, a GHS server must support ETS. The implementer should choose the type and resolution of the clock that is best suited for the sensor device type.

The implementer must choose whether the sensor clock can be set or not via ETS.

When the clock is set as defined in Section 3.3 of GHSP [20], the implementer must choose which policy on updating the time stamps of stored observations to follow:

- Update the time stamp to match the update of the clock.
- Set the flag to indicate that the time stamp is not from the current timeline.

For an E2E healthcare ecosystem, the use of UTC time with an offset for TZ/DST is the preferred choice. For other types of time, the GHS client has to map the time to UTC before uploading it to a FHIR® server. See Section 8.3.

7.5 How to make GHS implementations reusable for other sensor device types?

A GHS server is typically implemented as embedded software on dedicated hardware with limited memory and processing capabilities. It is possible to use these resources efficiently by implementing templates.

An example of a template of a numeric observation would be a structure with the fields and values for an oxygen saturation observation from a pulse oximeter. Most fields in this structure are fixed and can be defined as shown in Table 7.1: and from the template. In this template, the values of the current time and the result of the measurement can be filled in, as shown in italics in Table 7.1:.

Field	Data Type	Size (in octets)	Example	Value (hex)
Observation Class Type	uint8	1	Numeric observation	01
Length	uint16	2	29 octets	00 1D
Flags	Boolean[16] (a bitfield)	2	• Observation Type present • Time Stamp present • Supplemental Information present 0b0000 0000 0010 0011	00 23
Observation Type	uint32	4	MDC_PULS_OXIM_SAT_O2 – 0x00024BB8	00 02 4B B8
Time Stamp	***struct***	***9***	***The current time – variable***	***T0 T1 T2 T3 T4 T5 T6 T7 T8 T9***
Supplemental Information	struct	var	Count = 1 Codes = MDC_MODALITY_SPOT	01 00 02 4C 3C
Observation Value	-			

Field	Data Type	Size (in octets)	Example	Value (hex)
Unit code	uint16	2	MDC_DIM_PER_CENT	02 20
Value	***FLOAT***	***4***	***The observed value – variable***	***V1 V2 V3 V4***

Table 7.1: Example template structure

The resulting octets can then be passed to a transmitter function that transmits them as a GATT indication or notification of a GHS Health Observations characteristic.

This template can be adapted for the implementation of another numeric observation.

A GHS client is typically implemented on a mobile platform with enough resources to support all GHS features and can be built as a generic implementation without being affected by memory limitations. Here, an object-oriented programming approach can be followed with abstract methods for the generic features of GHS.

The Philips GHS client is an example of an implementation of GHS on the Android operating system that benefits from the memory available on even low-cost Android devices.

8 How to upload GHS data to an HL7® FHIR® server?

A GHS Observation characteristic value can be mapped to a FHIR® Observation resource, encoded in JSON, and uploaded to a FHIR® server. A GHS client receiving observations from a GHS server can use this mapping to upload FHIR® Observation resources to a FHIR® server. In the context of uploading data to a FHIR® server, a GHS client device acts as a Personal Health Gateway (PHG). This section covers some of the details of the mapping.

Table 8.1 gives an example of encoding a pulse oximeter reading into a JSON-encoded FHIR® observation.

ACOM OBSERVATION FIELD	VALUE	FHIR® OBSERVATION DATA ELEMENT & VALUE
OBJECT	Numeric Observation	{ “resourceType”: “Observation”,
TYPE	MDC_PULS_OXIM_SAT_O2 (2::19384)	“code”: {“coding”: [{ “system”: “urn:iso:std:iso:11073:10101”, “code”: “150456” }]},
UNIT CODE	MDC_DIM_PERCENT (4::544)	“valueQuantity”:{ “system”: “urn:iso:std:iso:11073:10101”, “code”: “262688”, “value”: 89.0 },
OBSERVED VALUE	89.0	
TIME STAMP	2017-09-04 16:30:00 +1:00 (1504542600)	“effectiveDateTime”: “2017-09-04T16:30:00+01:00” }

Table 8.1: Example mapping to FHIR®

The FHIR® PHD Implementation Guide (PHD IG) [21] for IEEE 11073 PHD devices that upload data to a FHIR® server is a useful source for information on this topic. This FHIR® IG currently uses FHIR® R4 and provides detailed information about how to map data following the IEEE 11073-10206 PHD standard to FHIR® resources. Because GHS is based on ACOM, and ACOM defines a simplified and updated version of the model used in IEEE 11073-20601, much of this guide is also applicable to GHS.

In addition to uploading the FHIR® Observation resources, the PHG should also upload or retrieve two FHIR® Device resources and upload or retrieve a FHIR® Patient resource at least once. These are used as references in the Observation resources, as illustrated in Figure 8.1:

for a Pulse Oximeter. The gatewayDevice is an extension to FHIR® defined in the PHD IG [21]. More details can be found in Section 8.10.

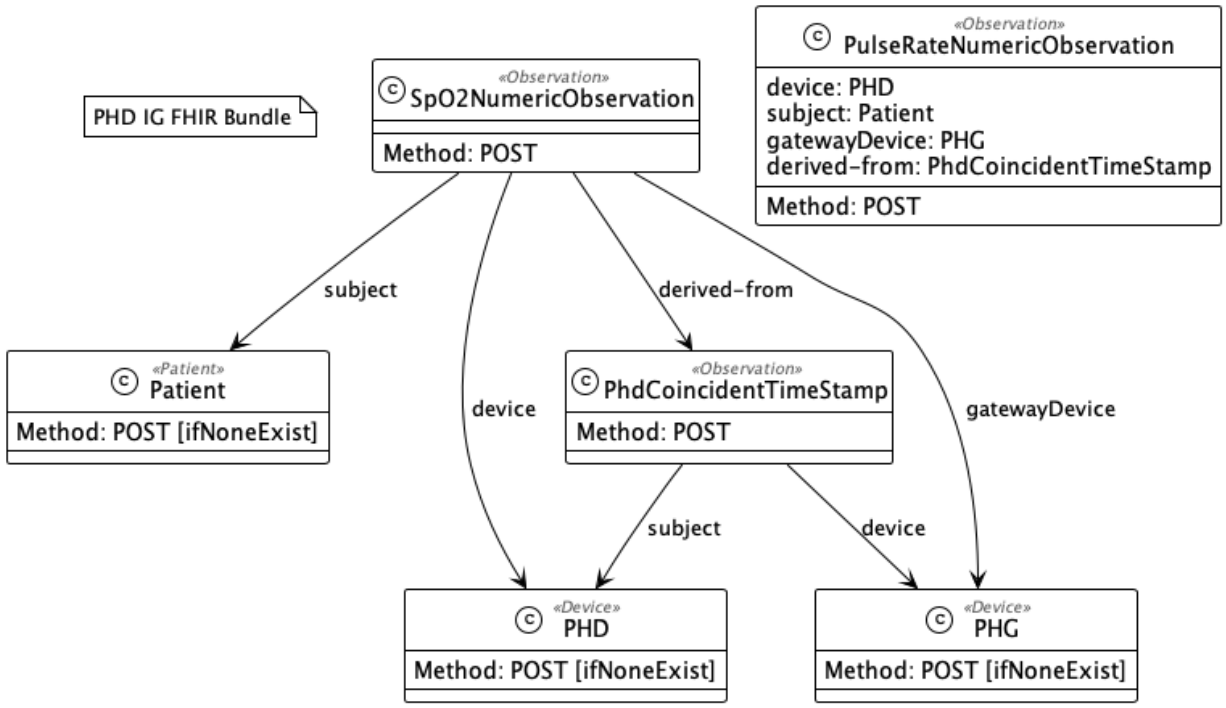

Figure 8.1: PHD IG FHIR® resources and their relationships

The references from the PhdCoincidentTimeStamp to subject and device are updated from what is used in the PHG IG examples. It is an observation made by the PHG with the PHD as subject and not the patient.

A GHS Client can use the mapping described in this section to create and upload FHIR® Observation resources, two Device resources – one for the GHS Server and one for the GHS Client, and to create or refer to one or more Patient resources.

8.1 Observation Class Type and Observation Value mapping

The GHS Observation Class Type field is not directly mapped to a FHIR® data element. Its value determines the type of the FHIR® value[x] field. See Table 8.2.

Observation Class Type	Value	FHIR® Value[x] Type
Numeric observation	1	Quantity
Simple discrete observation	2	CodeableConcept
String observation	3	String
Sample array observation	4	SampledData
Compound discrete event observation	5	Components of type CodeableConcept

Observation Class Type	Value	FHIR® Value[x] Type
Compound state/event observation	6	Components of type Boolean
Compound observation	7	Components
Type-Length-Value (TLV)-encoded observation	8	Depends on the Type field of the TLV
Observation bundle	255	Determined per observation in the bundle.

Table 8.2: Determining the FHIR® value[x] Type

8.2 Observation type

In GHS, each observation has an Observation Type field either directly or in the surrounding GHS bundle. This GHS field contains an MDC code as a uint32 value. This value can be mapped to a FHIR® Observation code data element using the decimal string representation of the MDC value and indicating that the code comes from the ISO/IEEE 11073 nomenclature system. The FHIR® Observation code for such an MDC code is shown in Code Snippet 8.1.

```
"code": {
  "coding": [
    {
      "system": "urn:iso:std:iso:11073:10101",
      "code": "<<decimal string value of the MDC code>>"
    }
  ],
  "text": "optional field with the RefId and definition of the MDC
code"
}
```

Code Snippet 8.1: FHIR® Observation code value for an MDC code

For several vital signs observations, FHIR® requires the use of LOINC®(*) codes [24] when the uploaded observations are searchable for vital signs monitoring of patients. The GHS client or the FHIR® server should add these LOINC codes to these vital signs FHIR® Observation resources. As an example, the resulting code for a FHIR® blood pressure observation resource is shown in Code Snippet 8.2.

(*) This material contains content from LOINC (http://loinc.org). LOINC is copyright © 1995-2023, Regenstrief Institute, Inc. and the Logical Observation Identifiers Names and Codes (LOINC) Committee and is available at no cost under the license at http://loinc.org/license. LOINC® is a registered United States trademark of Regenstrief Institute, Inc.

```
"code": {
  "coding": [
    {
      "system": "urn:iso:std:iso:11073:10101",
      "code": "150020"
    },
    {
      "system": "http://loinc.org",
      "code": "55284-4"
    }
  ],
  "text": "MDC_PRESS_BLD_NONINV: Blood pressure"
}
```

Code Snippet 8.2: FHIR® Observation code for a blood pressure observation

8.3 Time stamp

In FHIR®, an observation should have a time stamp reported through the effective[x] data element.

A GHS time stamp, when present, can be mapped to a FHIR® effective[x] data element as shown in Code Snippet 8.3.

```
If observation.hasTimeStamp Then
    If observation.timeStamp.fromCurrentTimeline Then
        readCurrentTimeFromServer
        If serverTimeisTickCounter or clientTimeIsMoreAccurate Then
            observation.adjustTimeStamp
            uploadCoincidentTimeStampToFHIRServer
            uploadObservationToFHIRServer
        Else
            uploadObservationToFHIRServer
        Endif
    Else
        If observation.timeStamp.isSynchronized Then
            uploadObservationToFHIRServer
        Else
            reportUnsynchronizedObservations
        Endif
    Endif
Else
    observation.addClientTimeStamp
    uploadObservationToFHIRServer
Endif
```

Code Snippet 8.3: Deriving the FHIR® time stamp for an observation

This pseudo code is further explained as follows:

UploadCoincidentTimeStampToFHIRServer: When the GHS client adjusts the time stamp of an observation, it will also upload a CoincidentTimeStamp Observation to the FHIR® server that reports the time of the GHS server on the timeline of the GHS client.

AddClientTimeStamp: When the GHS server does not have a clock, the GHS client's clock is used to add a time stamp to an observation before uploading it to a FHIR® server.

Figure 8.2 illustrates two scenarios with GHS observations with a tick counter time stamp, one with an observation not from the current timeline, and one with an observation from the current timeline that can be adjusted.

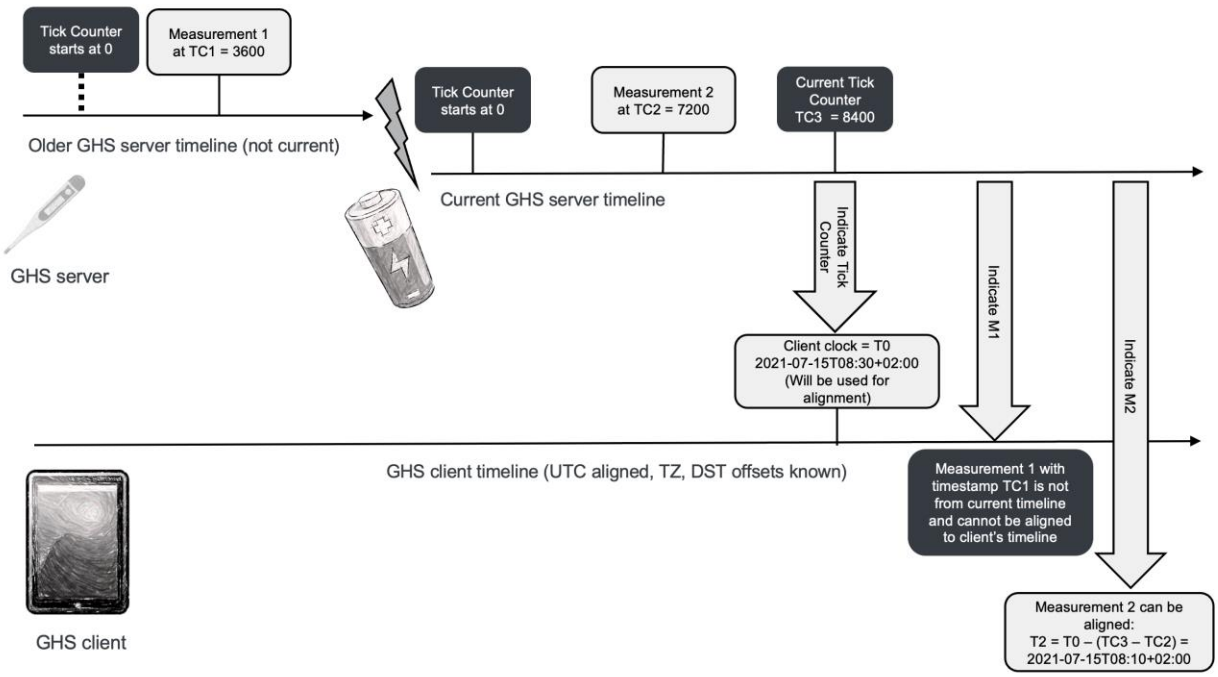

Figure 8.2: Observations from different timelines

The observation with a tick counter that is not from the current timeline cannot be aligned with a synchronized timeline and should be reported as an issue (reportUnsynchronizedObservations).

The GHS Time Stamp field can be mapped to a FHIR® effectiveDateTime or effectiveInstant data element. When the GHS Measurement Duration field is present, the time stamp and measurement duration value are mapped to a FHIR® effectivePeriod data element with a start time and an end time both of type DateTime.

The format of a FHIR® DateTime value with a time zone offset is: "YYYY-MM-DDThh:mm:ss[+|-]zz:zz".

The format of a FHIR® Instant value with a time zone offset is: "YYYY-MM-DDThh:mm:ss.sss[+|-]zz:zz" and supports millisecond resolution. This format should be used if the resolution of the GHS server clock is in milliseconds or better.

In FHIR®, the time is reported as UTC time, ideally with a time-zone offset, or as UTC time using the format "YYYY-MM-DDThh:mm:ss.sssZ", also known as Zulu time.

Because FHIR® requires the time to be UTC time, ideally with a time zone part zz:zz present, the GHS client should convert a GHS server time to UTC and fill in the time zone when not provided by the GHS server.

To convert a tick counter to a UTC time, the client should read the tick counter and record its own time at the moment of reading. This recorded pair of values enables the GHS client to map a tick counter value from the current timeline of the GHS server to a UTC time. This pair of readings should also be uploaded to the FHIR® server as a Coincident Time Stamp observation.

The GHS client also may correct the time provided by the GHS server if it determines its time as more accurate. In this case, a Coincident Time Stamp observation should also be uploaded. The same is true for the client converting the server time to UTC.

An example of a Coincident Time Stamp is shown in Code Snippet 8.4. There is a 1-minute difference between the time of the GHS server and that of the client. The references to PHD and PHG are further explained in Section 8.9.

```
{
  "resourceType": "Observation",
  "id": "obs-12345678",
  "meta": {
    "profile":
"http://hl7.org/fhir/uv/phd/StructureDefinition/PhdCoincidentTimeStam
pObservation"
    ]
  },
  "extension": [
    {
      "url": "http://hl7.org/fhir/StructureDefinition/observation-
gatewayDevice",
      "valueReference": {
        "reference": "Device/phg-xyz"
      }
    }
  ],
  "status": "final",
  "code": {
    "coding": [
      {
        "system": "urn:iso:std:iso:11073:10101",
        "code": "67975"
      }
    ],
    "text": "MDC_ATTR_TIME_ABS: Absolute time"
  },
  "effectiveDateTime": "2023-12-04T15:21:00+01:00",
  "valueDateTime": "2023-12-04T15:22:00+01:00",
  "device": {
    "reference": "Device/phd-abc"
```

```
    }
}
```

Code Snippet 8.4: Coincident time stamp resource

This Coincident Time Stamp observation should then be referenced as a “derived from” observation in any observation to which it applies.

See the section on coincident time stamps in the PHD IG [21].

8.4 Numeric values

In GHS, IEEE 11073 FLOATs are used to represent numeric measurement values. These are 32-bit structures consisting of a signed 8-bit exponent and a signed 24-bit mantissa. The number represented is (mantissa) × (10**exponent).

FLOATs support mapping to decimal strings without errors while maintaining precision.

In FHIR®, numeric values are represented as strings of a maximum of 18 decimal digits in the Observation.valueQuantity.value data element. When mapping FLOATs to FHIR® values, precision should be maintained.

FLOAT (hex)	Exponent	Mantissa	FHIR® Encoding
0x00000002	0	2	2
0xFF000014	-1	20	2.0
0xFE0000C8	-2	200	2.00

Table 8.3 Example GHS FLOAT to FHIR® mapping

There are also some special values: NaN, NRes, +Infinity, -Infinity, and RFU. The special values should be mapped to a corresponding Observation.dataAbsentReason:

- 0x007FFFFE = +Infinity → positive-infinity
- 0x007FFFFF = NaN → not-a-number
- 0x00800000= NRes → error
- 0x00800001= RFU → error – reserved for future use
- 0x00800002 = -Infinity → negative-infinity

8.5 Unit code

GHS uses IEEE MDC codes for the unit of any numeric observation or observation component.

FHIR® prefers the use of Unified Code for Units of Measure (UCUM) [22] codes; and when uploading numeric data, ideally the MDC code is replaced by the corresponding UCUM code.

RTMMS can be used to find the UCUM Unit for many existing MDC unit codes. For example, the MDC code MDC_DIM_BEAT_PER_MIN maps to the UCUM value “{beat}/min”, “{beats}/min”, “1/min”, or “/min”.

See https://rtmms.nist.gov/RTM/Unit/93?view-name=Details&view-type=Organizer.

8.6 Patient ID

The Continua Design Guidelines for FHIR® Observation Upload [23] cover several methods for a GHS gateway to obtain a logical identifier for the FHIR® Patient resource.

Here is a summary of these methods:

1. Obtain the patient’s logical identifier (FHIR® Patient.id) out of band: administrative staff configures the gateway with the logical identifier of the patient or user of the gateway.
2. The gateway uses another Patient Designator, such as the patient’s account number from an insurance company. This Patient Designator is used to generate a logical identifier for the patient on the FHIR® server or to retrieve a logical identifier for a patient from the FHIR® server.
3. The gateway uses OAuth to authenticate itself and to gain access to a limited scope of data on the FHIR® server that includes exactly one Patient resource. The gateway retrieves this resource and uses its logical identifier for subsequent observation uploads.

Other methods may be possible as well.

The patient’s logical identifier or a Patient Designator by themselves do not contain personally identifiable information (PII).

In this way, the GHS client can create or retrieve one or more Patient IDs that will be used when uploading Observations as reference in the “subject” data element of these Observations. When UDS is supported, the GHS client should create a Patient ID for each registered UDS user.

8.7 Measurement status

The Measurement status field is a bit field, and a bit set can sometimes be mapped to a code of an Observation data element, whereas other bits trigger other mappings.

- A subset of the GHS/ACOM measurement status bits map directly to HL7® FHIR® measurement status codes that are defined as a FHIR® CodeSystem.
- Other codes map to FHIR® Observation status codes.
- Some codes have no good mapping but trigger other actions.

See also the section on Measurement Status in the PHD IG [21]. An overview is shown in Table 8.4:.

GHS Measurement Status Bit	FHIR® Observation Resource Data Element	Further Handling Options
Invalid	dataAbsentReason = error – from CS1 status = entered-in-error – from CS2	A device or gateway could decide not to upload invalid observations and could report an error by other means.
Questionable	interpretation = questionable – from CS3	-
Not available	dataAbsentReason = not-performed – from CS1	A device or gateway could decide not to upload observations with no value and could report an error by other means.
Calibrating	interpretation = calibration-ongoing – from CS3	-
Test data	meta.security = HTEST – from CS4	Test data should in most cases not be uploaded to a FHIR® server, except for testing purposes.
Early estimate	interpretation = early-indication – from CS3 status = preliminary – from CS2	-
Manually entered	-	Add a note to the Observation resource that it was manually entered.
Setting	-	For device settings, the Observation resource should reference the Device resource as a subject and not a Patient resource.
Threshold error	interpretation = in-alarm – from CS3	Ignore or add a note to the Observation resource that it is outside its boundaries.
Thresholding disabled	interpretation = alarm-inhibited – from CS3	Ignore or add a note to the Observation resource that its boundaries are not checked.

Table 8.4: GHS-to-FHIR® observation status mapping

CS1: CodeSystem: Data Absent Reason – http://terminology.hl7.org/CodeSystem/data-absent-reason

CS2: CodeSystem: Observation Status – http://hl7.org/fhir/observation-status

CS3: CodeSystem: Measurement Status Codes – https://build.fhir.org/ig/HL7/uv-pocd/CodeSystem-measurement-status.html

CS4: CodeSystem: Act Reason – http://terminology.hl7.org/CodeSystem/v3-ActReason

8.8 Supplemental information

GHS supplemental information can be mapped to FHIR® observation components in the following generic way (with the variable part shown in ***bold italics***) shown in Table 8.5:.

GHS Supplemental Information	FHIR® Observation Component Data Elements

MDC_XXX
123456
	"component": [{
	"code": {
	"coding": [{
	"system": "urn:iso:std:iso:11073:10101",
	"code": "68193"
	}],
	"text": "MDC_ATTR_SUPPLEMENTAL_TYPES"
	},
	"valueCodeableConcept": {
	"coding": [{
	"system": "urn:iso:std:iso:11073:10101",
	"code": "*123456*"
	}],
	"text": "***MDC_XXX: common term or description - optional***"
	}
	}]

MDC_MODALITY_SPOT
150588
	"component": [{
	"code": {
	"coding": [{
	"system": "urn:iso:std:iso:11073:10101",
	"code": "68193"
	}],
	"text": "MDC_ATTR_SUPPLEMENTAL_TYPES"
	},
	"valueCodeableConcept": {
	"coding": [{
	"system": "urn:iso:std:iso:11073:10101",
	"code": "150588"
	}],
	"text": "***MDC_MODALITY_SPOT: SpO2 spot check***"
	}
	}]

Table 8.5: GHS supplemental information mapping to FHIR®

For more details, see this section of the PHD IG [21].

8.9 References (Observation ID, Derived From, and Is Member Of)

When uploading Observations, the PHG can follow one of the following strategies to fill in references to the patient and the device resources:

- Use a conditional create operation to repeatedly upload the patient information and device information in a FHIR® transaction bundle with a new set of observations.

- Remember the IDs of the patient and device resources that are returned by the FHIR® server on initial upload and use these IDs as a reference in subsequent Observation uploads.

A similar strategy can be followed for Observations that refer to each other. The IDs of referenced Observations can be remembered for later reference, or the referenced Observation can be included in the FHIR® transaction bundle with a conditional create operation.

The Derived From reference can be directly mapped to a FHIR® Observation derivedFrom data element.

The Is Member Of reference relation between GHS observations must be reversed and mapped to a FHIR® Observation hasMember data element in the referenced observation.

8.10 Device resources

The information of the GHS Server or PHD that should be uploaded in a FHIR® Device resource by the PHG comes from different services:

- The Device Information Service
  - Any information from the DIS that uniquely identifies a device can be used for the FHIR® Device.identifier. This includes the System ID, the Serial Number String, and the UDI for Medical Devices. If these are missing, then the Bluetooth Device Address can be used, provided that this address is static (see CS, Vol 3, Part C, Section 15.1).
- The Elapsed Time Service
- The GHS Service

Table 8.6 provides an overview of this information. More details can be found in this section of the PHD IG [21]. The mapping for the supported device specializations is different for FHIR® R4 and FHIR® R5.

GHS Device Information	FHIR® Device Resource Data Element
Model Number String characteristic (from DIS)	modelNumber
Manufacturer Name String characteristic (from DIS)	manufacturer
Serial Number String (from DIS)	serialNumber
Hardware Revision String (from DIS)	version.value, with version.type = Hardware revision
Firmware Revision String (from DIS)	version.value, with version.type = Firmware revision
Software Revision String (from DIS)	version.value, with version.type = Software revision

GHS Device Information	FHIR® Device Resource Data Element
System ID (from DIS)	identifier.value = 8 2-digit hex numbers separated by dashes "-" with identifier.system=”urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680” identifier.type.coding.system=”http://hl7.org/fhir/uv/phd/CodeSystem/ContinuaDeviceIdentifiers” identifier.type.coding.code=”SYSID”
Bluetooth identity address	identifier.value = 6 2-digit hex numbers separated by dashes "-" identifier.system=”http://hl7.org/fhir/sid/eui-48/bluetooth” identifier.type.coding.system=”http://hl7.org/fhir/uv/phd/CodeSystem/ContinuaDeviceIdentifiers” identifier.type.coding.code=”BTMAC”
UDI for Medical Devices (from DIS), with the following fields: - UDI Label - UDI Device Identifier - UDI Issuer - UDI Authority	udiCarrier data element with the following sub elements - udiCarrier.carrierHRF = UDI Label - udiCarrier.deviceIdentifier = UDI Device Identifier - udiCarrier.issuer = UDI Issuer - udiCarrier.jurisdiction = UDI Authority - udiCarrier.entryType = electronic-transmission (fixed value)
Supported Device Specializations (from GHSS, Features characteristic) - a set of tuples of MDC-codes and version numbers	FHIR® R4: - specialization.systemType = {code = MDC-code, system = urn:iso:std:iso:11073:10101} - specialization.version = version number FHIR® R5: - conformsTo.category = communication (fixed value) - conformsTo.specification = {code = MDC-code, system = urn:iso:std:iso:11073:10101} - conformsTo.version = version number If in GHSS just one Device Specialization is supported, this can be mapped to Device.type directly; otherwise, the MDC-code MDC_MOC_VMS_MDS_SIMP can be used to give Device.type a value.
Current Elapsed Time characteristic (from ETS) - supports write - Time is a tick counter - Time is UTC - Time resolution - TZ/DST offset is used - Clock capabilities	The clock capabilities map to a set of FHIR® Device.property data elements, with code system http://hl7.org/fhir/uv/phd/CodeSystem/ASN1ToHL7 and a value of "Y" or "N". The following codes can be used: - 68219.0 - the clock supports (absolute) date time (without offset) - 68219.1 - the clock can be set - 68219.2 - the clock is a tick counter (also known as

GHS Device Information	FHIR® Device Resource Data Element
- Clock applies DST rules - Clock manages TZ changes	"relative time") - 68219.7 - the clock supports date time with TZ/DST offset
Current Elapsed Time characteristic (from ETS) - Time resolution	The clock resolution maps to a Device.property data element, with code system "urn:iso:std:iso:11073:10101" and a valueQuantity matching the Time resolution. The MDC code used depends on the type of clock: MDC_TIME_RES_REL - 68223 - for a tick counter MDC_TIME_RES_ABS - 68222 - for an absolute time, time without offset MDC_TIME_RES_BO - 68239 - for a time with offset

Table 8.6: Device information mapping

The GHS Client acting as a PHG should also upload information about itself as a FHIR® Device resource to the FHIR® server, because it plays a role in the time synchronization of the GHS server and the uploaded observations as explained in Section 7.4. Information that should be included in this PHG device resource includes:

- The time synchronization method (e.g., NTP) and accuracy (e.g., 1 minute)
- The manufacturer and model number

For more details, see this section of the PHD IG [21].

9 How to handle privacy and security aspects of medical data in a GHS implementation?

A GHS server that handles medical data should take proper care of privacy and security aspects. Some recommendations follow:

- Use an encrypted connection when transferring GHS observations or information that uniquely identifies the device or its user(s).
- The use of Secure Connections Only mode, also known as “FIPS” mode, is recommended. In this mode, security mode 1 level 4 must be used except for services that only require security mode 1 level 1 (as defined in Volume 3, Part C, Section 10.2.4 of the Bluetooth Core Specification [6]).
- The support and the use of LE Legacy Pairing should be avoided.
- Use bonding to prevent connections with arbitrary clients, and make sure that bonds and user data can be deleted from the device by the user(s).
- Support of a secure way to delete all user data from a device is strongly recommended.
- Do not advertise in discoverable mode unless initiated by the user.
- Support of the Encrypted Advertising feature is recommended and should be used after bonding.
- Use link layer privacy in personal health devices that are typically carried by the user.
- Do not include any device-traceable information in the advertisement data or in the scan response data.
- The use of the Privacy feature is recommended. This reduces the ability to track a LE Device over a period of time by changing the Bluetooth Device Address on a frequent basis.

More elaborate recommendations on privacy and security can be found in the Bluetooth® Security and Privacy Best Practices Guide [25].

As a consequence of the GHSP specification [20], a GHS client must support GHS servers that follow these recommendations.

10 Conclusion

In this document, we covered various aspects of the design and implementation of GHS clients and servers. To provide feedback on this publication, Bluetooth SIG members can send their suggestions to the Medical Devices Working Group: med-main@bluetooth.org.

11 References

[1] HL7® International, “FHIR® Specification” – current published version, https://hl7.org/fhir/

[2] Institute of Electrical and Electronics Engineers (IEEE), “11073-10101-2019 - IEEE Standard for Health informatics--Point-of-care medical device communication - Part 10101: Nomenclature”, October 2019, https://standards.ieee.org/standard/11073-10101-2019.html

[3] Institute of Electrical and Electronics Engineers (IEEE), “11073-10206-2022 - Health informatics -- Device interoperability -- Part 10206: Personal health device communication -- Abstract Content Information Model”, January 2023, https://standards.ieee.org/ieee/11073-10206/10311/

[4] The Office of the National Coordinator for Health Information Technology, https://www.healthit.gov/topic/about-onc, providing one-stop access to information on health IT from the U.S. Government

[5] United States Core Data for Interoperability (USCDI), https://www.healthit.gov/isa/united-states-core-data-interoperability-uscdi

[6] Bluetooth Specifications and Documents page, https://www.bluetooth.com/specifications/specs/

[7] GATT Specification Supplement, https://www.bluetooth.com/specifications/gss/

[8] Assigned Numbers document, https://www.bluetooth.com/specifications/specs/assigned-numbers/

[9] HL7® International, “Using MDC codes with FHIR”, https://terminology.hl7.org/MDC.html

[10] IEEE 11073-104xx standards, IEEE 11073 Standards Committee, https://standards.ieee.org/search/?q=11073

[11] Personal Health Device Transcoding white paper, https://www.bluetooth.com/bluetooth-resources/personal-health-devices-transcoding/

[12] Rosetta Terminology Mapping Management System (RTMMS)/NIST, https://rtmms.nist.gov/

[13] IEEE 11073 Standards Committee homepage, https://sagroups.ieee.org/11073/

[14] LNI’s Nordic MPM and GHS Server implementations, https://github.com/brianreinhold/NordicPHDs#nordic-mpm-and-ghs-server-implementations

[15] LNI’s Health@Home Android Hub, https://www.lnihealth.com/products/healthhome-android-hub

[16] BLESSED for Android - BLE made easy, https://github.com/weliem/blessed-android

[17] Philips BLE Generic Health Sensor Client Example, https://github.com/philips-labs/BLE-GHS-Client-Example

[18] Philips BLE Generic Health Sensor Server Simulator, https://github.com/philips-labs/BLE-GHS-Server-Simulator

[19] Generic Health Sensor Service specification, https://www.bluetooth.com/specifications/specs/generic-health-sensor-service/

[20] Generic Health Sensor Profile specification, https://www.bluetooth.com/specifications/specs/generic-health-sensor-profile/

[21] HL7® International, “Personal Health Device Implementation Guide” for HL7® FHIR®, https://build.fhir.org/ig/HL7/phd/index.html

[22] Unified Code for Units of Measure (UCUM), https://ucum.org/

[23] Continua Design Guidelines for FHIR® Observation Upload, published by the ITU as “HSTP-H812-FHIR - Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload for trial implementation", October 2017, https://www.itu.int/dms_pub/itu-t/opb/tut/T-TUT-EHT-2017-H812FHIR-PDF-E.pdf

[24] Regenstrief Institute, Inc., Logical Observation Identifiers Names and Codes (LOINC), https://loinc.org/

[25] Bluetooth® Security and Privacy Best Practices Guide, https://www.bluetooth.com/bluetooth-resources/bluetooth-security-and-privacy-best-practices-guide/

Appendix A ACOM GHS UML models

This appendix provides a UML of ACOM basic classes and of ACOM Observation classes.

A.1 ACOM basics

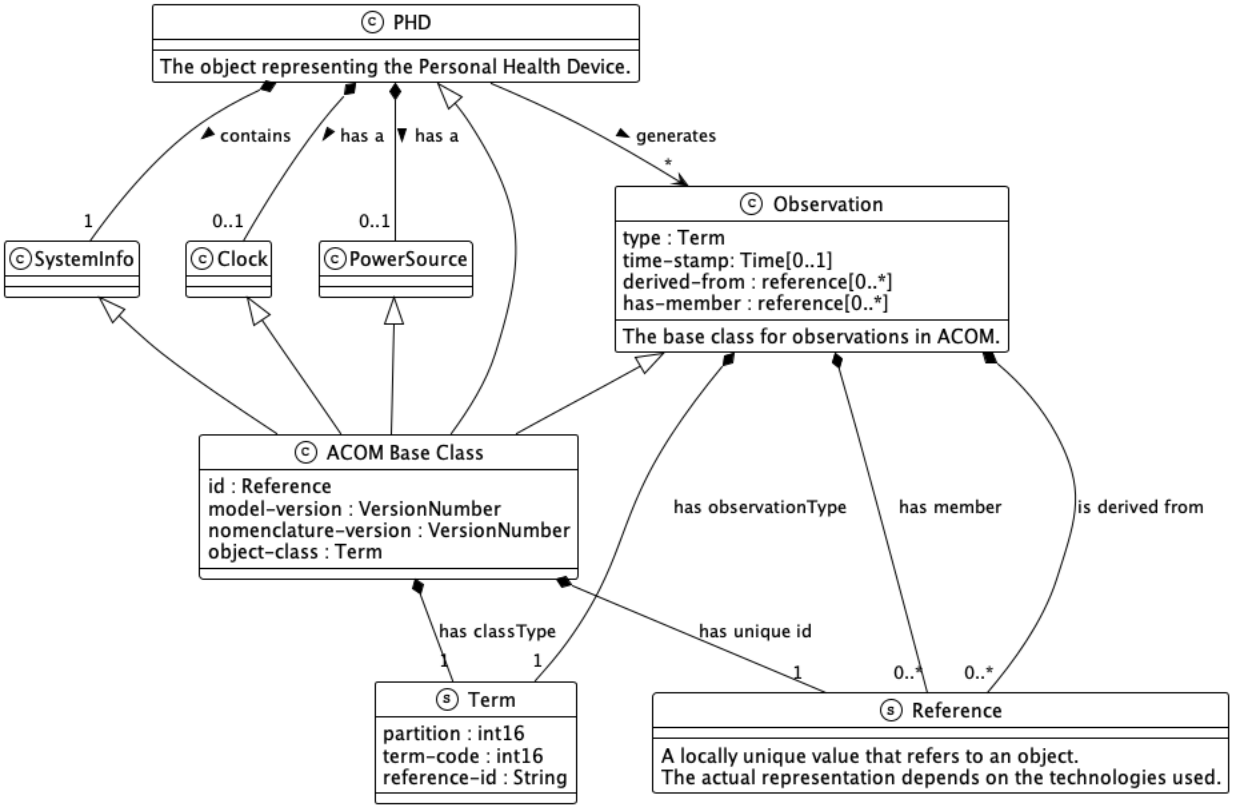

Figure A.1: ACOM basic classes

A.2 ACOM observations

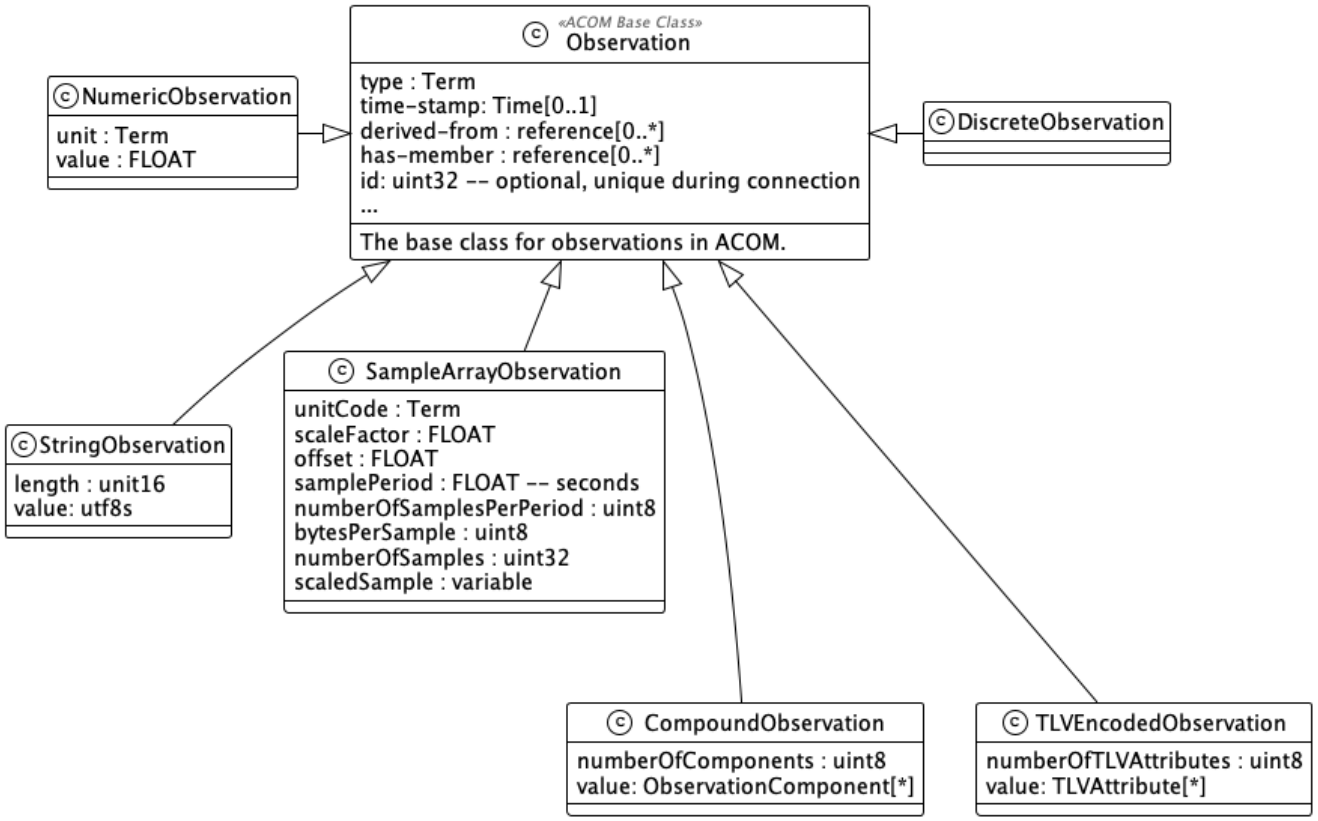

Figure A.2: ACOM observations

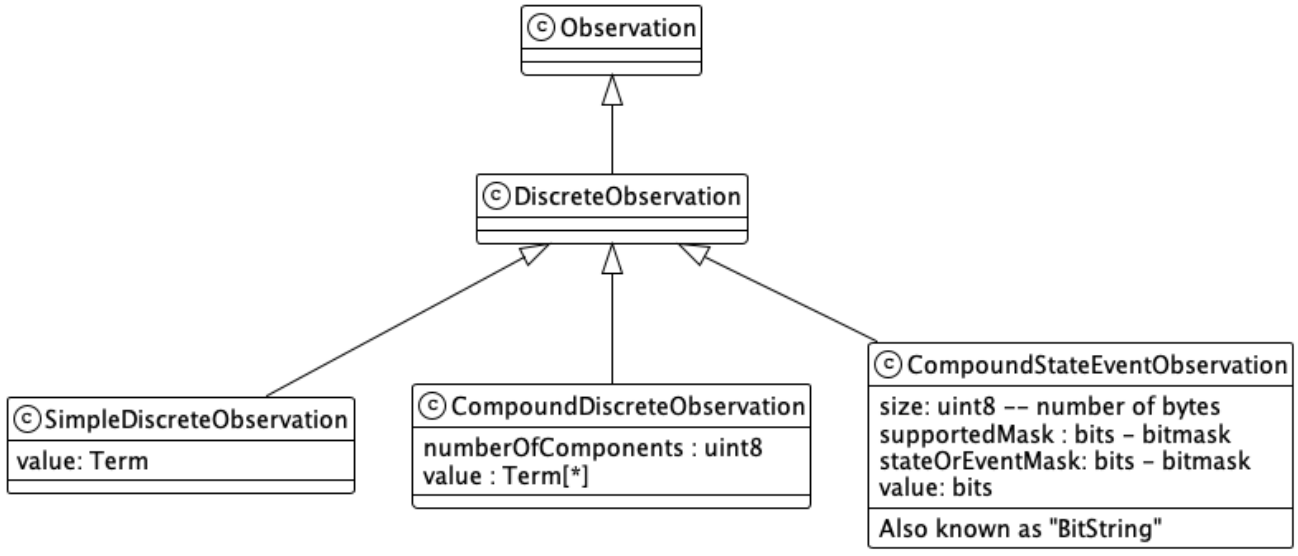

Figure A.3: ACOM Discrete Observations

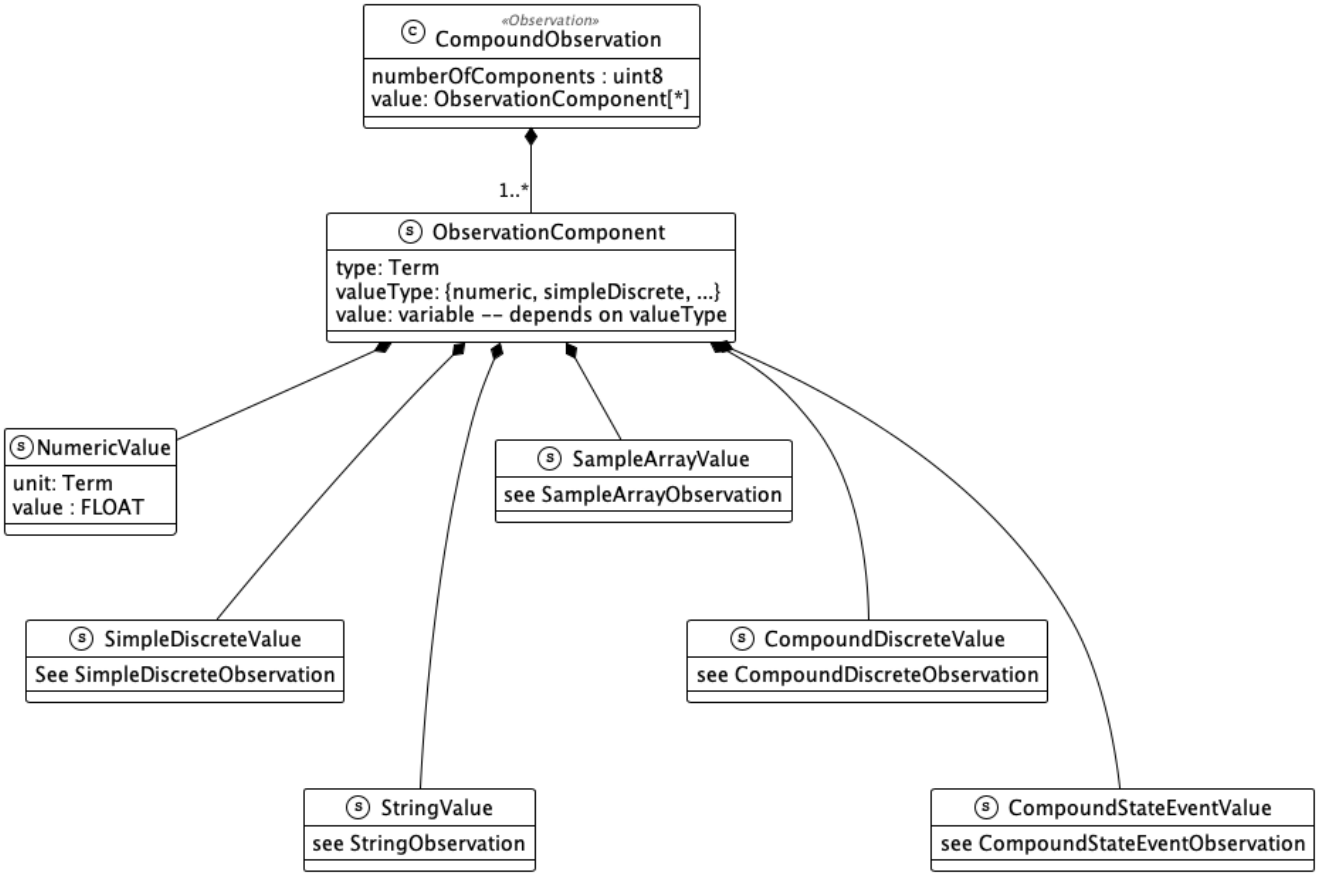

Figure A.4: ACOM Compound Observation

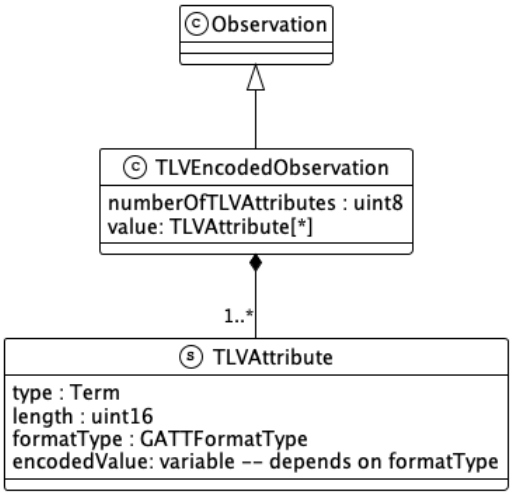

Figure A.5: ACOM TLV-encoded Observation

Appendix B GHS device specializations

This appendix gives details for a set of common GHS/ACOM device specializations in terms of the MDC codes that are used to encode the device specializations and the generated observations and provides a UML model of the specializations.

B.1 Thermometer

This section covers the information model for a thermometer and the relevant MDC codes.

- Device specialization = MDC_DEV_SPEC_PROFILE_TEMP.
- Observation type(s) = MDC_TEMP_BODY for a generic body temperature thermometer, or a more specific code from the following set:
  - MDC_TEMP_AXILLA – Armpit
  - MDC_TEMP_EAR – Ear (usually ear lobe)
  - MDC_TEMP_FINGER – Finger
  - MDC_TEMP_GIT – Gastro-intestinal Tract
  - MDC_TEMP_ORAL – Mouth
  - MDC_TEMP_RECT – Rectum
  - MDC_TEMP_TOE – Toe
  - MDC_TEMP_TYMP – Tympanum (ear drum)
- The temperature observation is a numeric observation. The MDC code for the unit in which the temperature is measured is MDC_DIM_DEGC or MDC_DIM_FAHR.

A simple UML diagram is shown in Figure B.1.

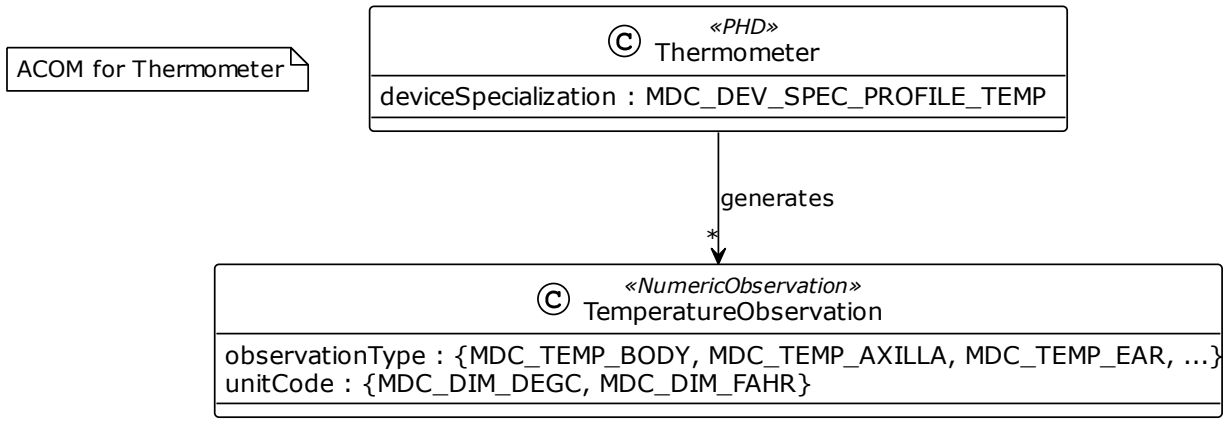

Figure B.1: Thermometer specialization

B.2 Heart rate monitor

This section covers the information model for a heart rate monitor and the relevant MDC codes.

- Device specializations:
  - MDC_DEV_SPEC_PROFILE_ECG – for devices that include an ECG waveform.
  - MDC_DEV_SPEC_PROFILE_HF_CARDIO – for devices that report energy expenditure.
  - MDC_DEV_SUB_SPEC_PROFILE_HR – for simple devices that only report HR and no ECG or energy expenditure.
- Supported observations:
  - Heart rate
  - RR-interval - optional
  - Energy expended - optional
  - ECG waveform - optional
  - ECG Context Data Trigger - optional
- Heart rate: numeric
  - Type: MDC_ECG_HEART_RATE_INSTANT
  - Unit: MDC_DIM_BEAT_PER_MIN
- RR-interval: numeric
  - Type: MDC_ECG_TIME_PD_RR_GL,
  - Unit: MDC_DIM_TICK (1/1024s) (other time units like MDC_DIM_MILLI_SEC could be used as well)
- Energy expended:
  - Numeric: MDC_HF_ENERGY,
  - Unit: MDC_DIM_JOULES
- ECG waveform: RTSA
  - Type: MDC_ECG_ELEC_POTL_xx, with xx indicating the lead (I, II, III or V1, V2, V3, etc.) if multiple leads are supported,
  - Unit: MDC_DIM_MILLI_VOLT
- ECG device status: Discrete Event - bit string 16
  - Type: MDC_ECG_DEV_STAT,
  - Bits:
    - leadwire-loss(0)
    - leadsignal-loss(1)
    - leadwire-loss-first-lead(2)
    - leadsignal-loss-first-lead(3)

- leadwire-loss-second-lead(4)
- leadsignal-loss-second-lead(5)
- leadwire-loss-third-lead(6)
- leadsignal-loss-third-lead(7)

- ECG Context Data Trigger: Compound Discrete Event
  - Type: MDC_ECG_EVT_CTXT_GEN,
  - Value set:
    - MDC_ECG_EVT_CTXT_USER,
    - MDC_ECG_EVT_CTXT_PERIODIC,
    - MDC_ECG_EVT_CTXT_DETECTED,
    - MDC_ECG_EVT_CTXT_EXTERNAL

A UML diagram of the heart rate monitor is shown in Figure B.2.

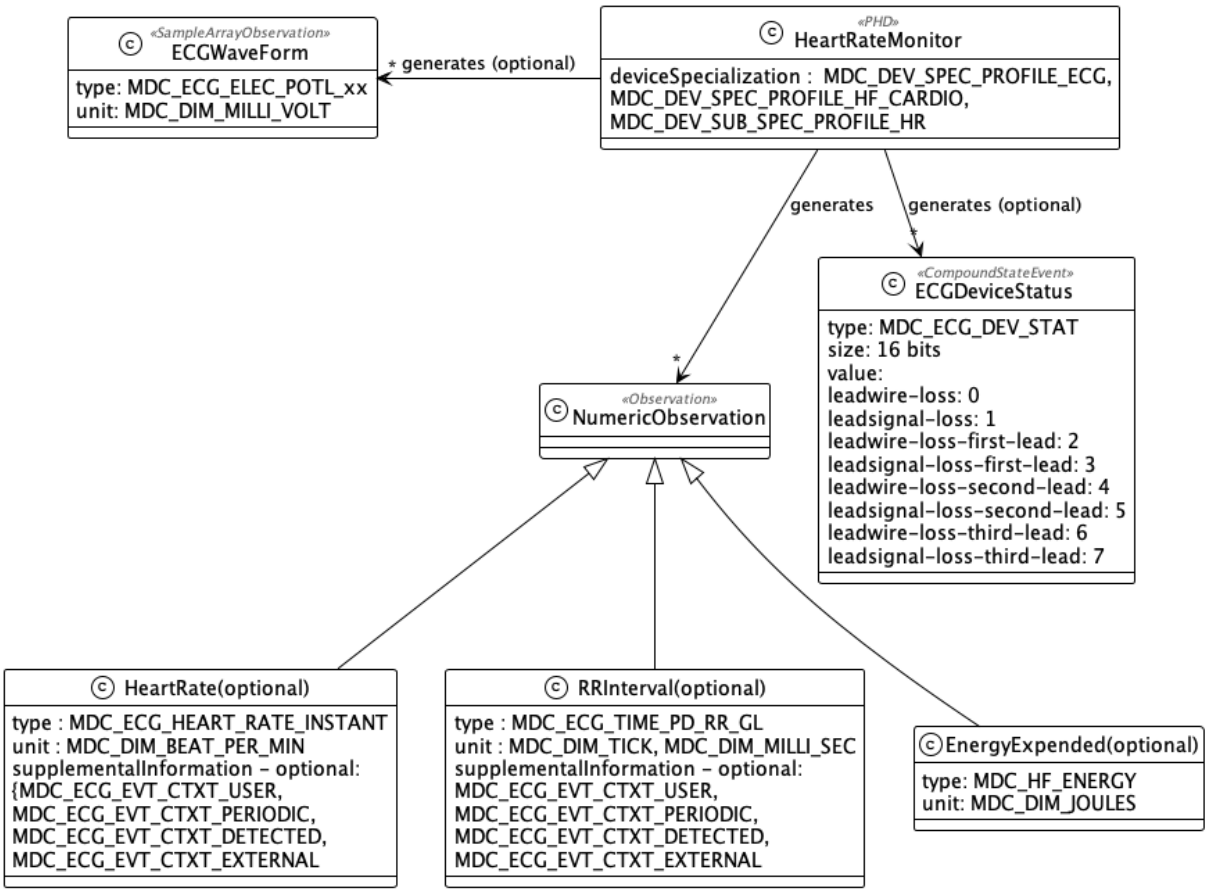

Figure B.2: Heart rate monitor specialization

B.3 Blood pressure monitor

This section covers the information model for a blood pressure monitor and the relevant MDC codes.

- Device specialization:
  - MDC_DEV_SPEC_PROFILE_BP
- Supported observations:
  - Blood pressure measurement
  - Pulse rate measurement
  - Blood pressure measurement status - optional
- Blood pressure measurement:
  - Compound observation
  - Type: MDC_PRESS_BLD_NONINV
  - Supplemental information (e.g., body location): - optional
    - MDC_LOEXT_LEG
    - MDC_LOEXT_LEG_L
    - MDC_LOEXT_LEG_R
    - MDC_LOEXT_THIGH
    - MDC_LOEXT_THIGH_L
    - MDC_LOEXT_THIGH_R
    - MDC_UPEXT_FOREARM
    - MDC_UPEXT_FOREARM_L
    - MDC_UPEXT_FOREARM_R
    - MDC_UPEXT_ARM_UPPER
    - MDC_UPEXT_ARM_UPPER_L
    - MDC_UPEXT_ARM_UPPER_R
  - Number of components: 2 or 3
    - Systolic blood pressure: Numeric component
      - Type: MDC_PRESS_BLD_NONINV_SYS
      - Unit: MDC_DIM_MMHG, MDC_DIM_KILO_PASCAL
    - Diastolic blood pressure: Numeric component
      - Type: MDC_PRESS_BLD_NONINV_DIA

        - Unit: MDC_DIM_MMHG, MDC_DIM_KILO_PASCAL
    - Mean blood pressure: Numeric component - optional
        - Type: MDC_PRESS_BLD_NONINV_DIA
        - Unit: MDC_DIM_MMHG, MDC_DIM_KILO_PASCAL
- Pulse rate measurement:
  - Numeric observation
  - Type: MDC_PULSE_RATE_NON_INV
  - Unit: MDC_DIM_BEAT_PER_MIN
- Blood pressure measurement status:
  - Compound state/event observation
  - Type: MDC_BLOOD_PRESSURE_MEASUREMENT_STATUS
  - Size: 16 bits
  - Derived from - optional: blood pressure measurements, pulse rate measurements
  - Value:
    - bit 0: body-movement
    - bit 1: cuff-too-loose
    - bit 2: irregular-pulse
    - bit 3: pulse-under-range-limit
    - bit 4: pulse-over-range-limit
    - bit 5: improper-body-position
    - other bits: RFU

The UML diagram is shown in Figure B.3.

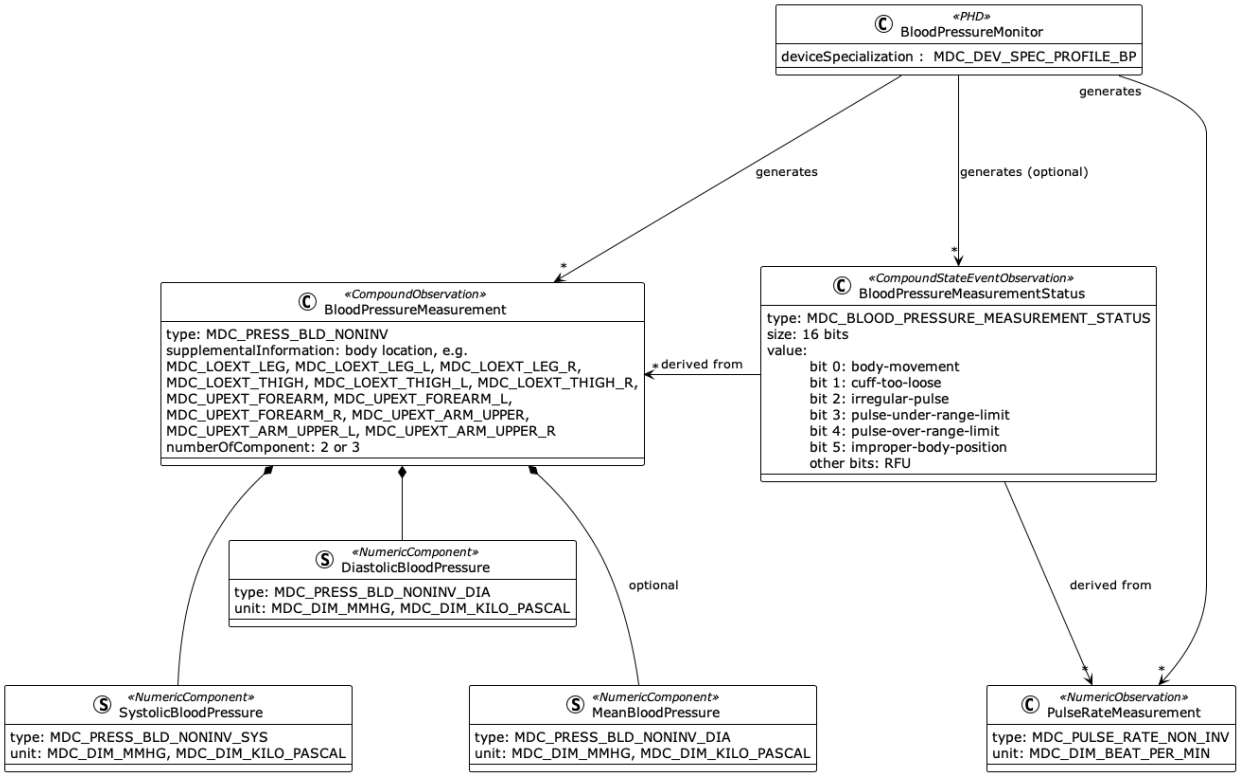

Figure B.3: Blood pressure monitor specialization

B.4 Glucose Meter

This section covers the information model for a glucose meter and the relevant MDC codes.

- Device specialization:
  - MDC_DEV_SPEC_PROFILE_GLUCOSE
- Supported observations:
  - Glucose Measurement
  - Medication Measurement - optional
  - Carbohydrate Measurement - optional
  - Exercise Intensity Measurement - optional
  - HbA1 Concentration Measurement - optional
- Glucose Measurement: Numeric Observation
  - Type:
    - MDC_CONC_GLU_CAPILLARY_WHOLEBLOOD,
    - MDC_CONC_GLU_CAPILLARY_PLASMA,
    - MDC_CONC_GLU_VENOUS_WHOLEBLOOD,

        - MDC_CONC_GLU_VENOUS_PLASMA,
        - MDC_CONC_GLU_ARTERIAL_WHOLEBLOOD,
        - MDC_CONC_GLU_ARTERIAL_PLASMA,
        - MDC_CONC_GLU_UNDETERMINED_WHOLEBLOOD,
        - MDC_CONC_GLU_UNDETERMINED_PLASMA,
        - MDC_CONC_GLU_ISF,
        - MDC_CONC_GLU_CONTROL
    - Unit: {MDC_DIM_MILLI_G_PER_DL, MDC_DIM_MILLI_MOLE_PER_L}
- Medication Measurement: Numeric Observation
    - Type: MDC_CTXT_MEDICATION
    - Unit: {MDC_DIM_MILLI_G, MDC_DIM_MILLI_L, MDC_DIM_INTL_UNIT}
    - Supplemental Information:
        - MDC_CTXT_MEDICATION_RAPIDACTING
        - MDC_CTXT_MEDICATION_SHORTACTING
        - MDC_CTXT_MEDICATION_INTERMEDIATEACTING
        - MDC_CTXT_MEDICATION_LONGACTING
        - MDC_CTXT_MEDICATION_PREMIX
    - Derived from: Glucose Measurements
- Carbohydrate Measurement: Numeric Observation
    - Type: MDC_CTXT_GLU_CARB
    - Unit: MDC_DIM_G
    - Supplemental Information:
        - MDC_CTXT_GLU_CARB_BREAKFAST or
        - MDC_CTXT_GLU_CARB_LUNCH or
        - MDC_CTXT_GLU_CARB_DINNER or
        - MDC_CTXT_GLU_CARB_SNACK or
        - MDC_CTXT_GLU_CARB_DRINK or
        - MDC_CTXT_GLU_CARB_SUPPER or
        - MDC_CTXT_GLU_CARB_BRUNCH or
        - MDC_CTXT_GLU_CARB_UNDETERMINED or
        - MDC_CTXT_GLU_CARB_OTHER or

        - MDC_CTXT_GLU_CARB_NO_ENTRY or
        - MDC_CTXT_GLU_CARB_NO_INGESTION
    - Derived from: Glucose Measurements
- Exercise Intensity Measurement: Numeric Observation
    - Type: MDC_CTXT_GLU_EXERCISE
    - Measurement Duration: FLOAT (seconds)
    - Unit: MDC_DIM_PER_CENT (percent of maximum)
    - Derived from: Glucose Measurements
- HbA1 Concentration Measurement: Numeric Observation
    - Type: MDC_CONC_HB1AC
    - Unit: MDC_DIM_PER_CENT
    - Derived from: Glucose Measurements

The UML diagram is shown in Figure B.4.

Figure B.4: Glucose meter specialization

B.5 Weight scale

This section covers the information model for a weight scale and the relevant MDC codes.

- Device specialization:
  - MDC_DEV_SPEC_PROFILE_SCALE
- Supported observations:
  - Body mass observation
  - Body length observation - optional
  - BMI observation - optional
- Body mass observation: Numeric Observation
  - Type: MDC_MASS_BODY_ACTUAL
  - Unit: MDC_DIM_KILO_G, MDC_DIM_LB
  - Patient: optional
- Body length observation: Numeric Observation
  - Type: MDC_LEN_BODY_ACTUAL
  - Unit: MDC_DIM_CENTI_M, MDC_DIM_INCH
  - Observation status: manually-entered
  - Patient: optional
- BMI observation: Numeric Observation
  - Type: MDC_RATIO_MASS_BODY_LEN_SQ
  - Unit: MDC_DIM_KG_PER_M_SQ
  - Observation status: calculation
  - Derived from: Body length observation, Body mass observation
  - Patient: optional

The UML diagram is shown in **Error! Reference source not found.**.

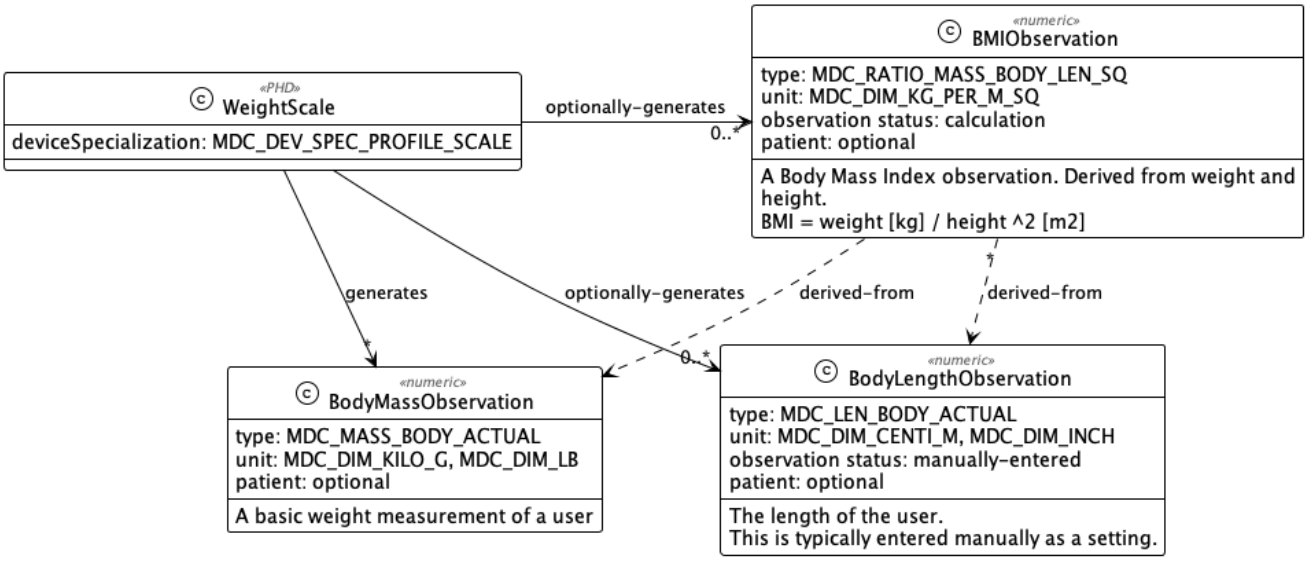

Figure B.5: Weight scale specialization

B.6 Pulse oximeter

This section covers the information model for a pulse oximeter.

- Device specialization: MDC_DEV_SPEC_PROFILE_PULS_OXIM
- Supported observations:
  - Oxygen Saturation
  - Pulse Rate
  - Pulsatile Quality (optional)
  - Plethysmographic Waveform (optional)
  - Sensor Status (optional)
- Oxygen Saturation: Numeric Observation
  - Type: MDC_PULS_OXIM_SAT_O2
  - Unit: MDC_DIM_PER_CENT
  - Supplemental information: MDC_MODALITY_FAST, MDC_MODALITY_SLOW, MDC_MODALITY_SPOT
- Pulse Rate: Numeric Observation
  - Type: MDC_PULS_OXIM_PULS_RATE
  - Unit: MDC_DIM_BEAT_PER_MIN
  - Supplemental information: MDC_MODALITY_FAST, MDC_MODALITY_SLOW, MDC_MODALITY_SPOT,

MDC_PULS_OXIM_PULS_CHAR_NOMINAL,
MDC_PULS_OXIM_PULS_CHAR_MARGINAL,
MDC_PULS_OXIM_PULS_CHAR_MINIMAL,
MDC_PULS_OXIM_PULS_CHAR_UNACCEPTABLE

- Pulsatile Quality: Numeric Observation
  - Type: MDC_SAT_O2_QUAL
  - Unit: MDC_DIM_PER_CENT
- Plethysmographic Waveform: Sample Array Observation
  - Type: MDC_PULS_OXIM_PLETH
  - Unit: MDC_DIM_DIMLESS
- Sensor Status: Compound State/Event
  - Type: MDC_PULS_OXIM_DEV_STATUS
  - Size: 16 bits
  - Value:
    - sensor-disconnected(0),
    - sensor-malfunction(1),
    - sensor-displaced(2),
    - sensor-unsupported(3),
    - sensor-off(4),
    - sensor-interference(5),
    - signal-searching(6),
    - signal-pulse-questionable(7),
    - signal-non-pulsatile(8),
    - signal-erratic(9),
    - signal-low-perfusion(10),
    - signal-poor(11),
    - signal-inadequate(12),
    - signal-processing-irregularity(13),
    - device-equipment-malfunction(14),
    - device-extended-update(15)

The UML diagram is shown in Figure B.6.

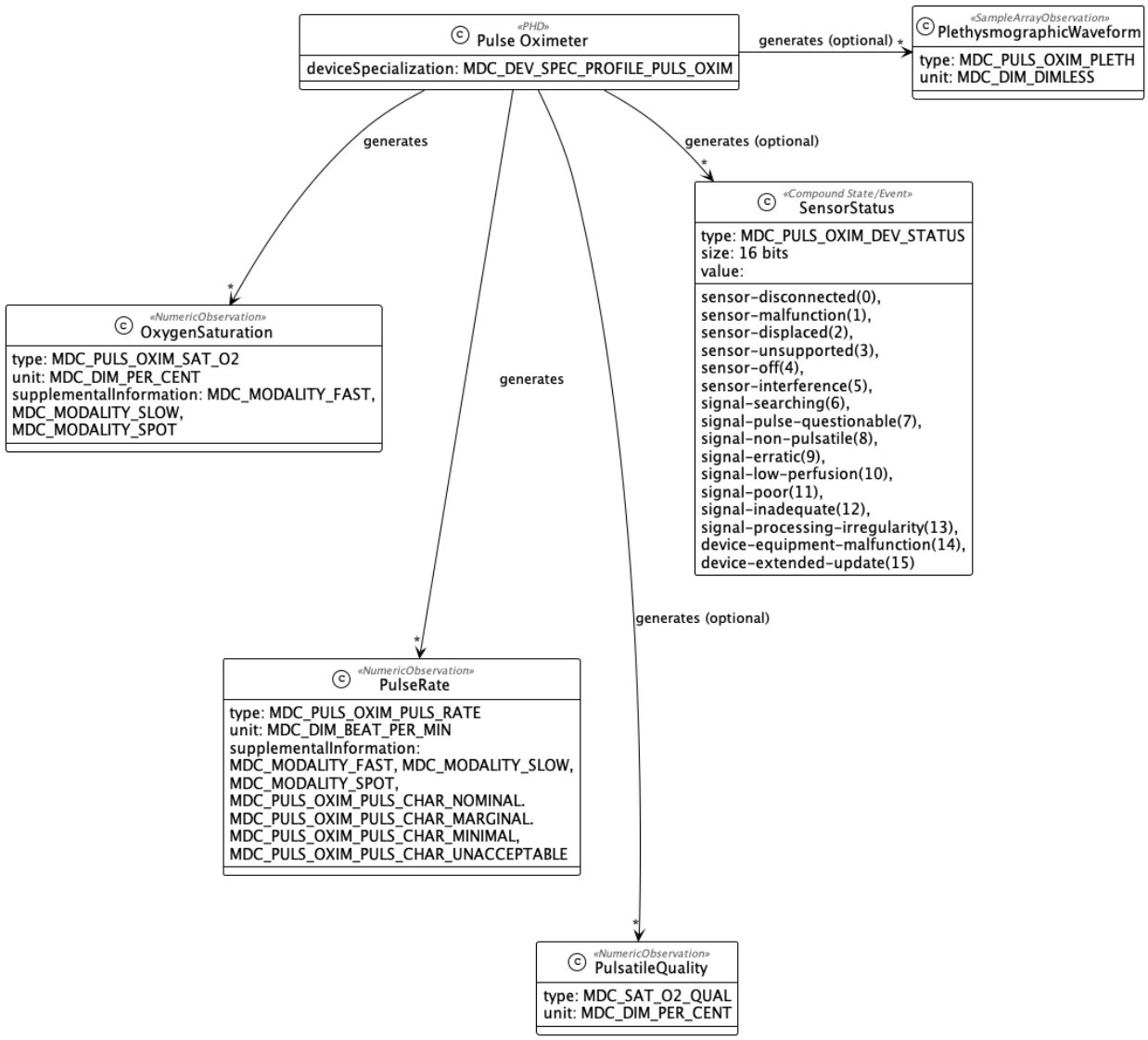

Figure B.6: Pulse oximeter specialization