International Telecommunication Union

ITU-T Technical Paper

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(27 October 2017)

HSTP-H812-FHIR
Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload for trial implementation

ITU-T

International Telecommunication Union

Change Log

This document contains Version 1 of the ITU-T Technical Paper HSTP-H812-FHIR on "*Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload for trial implementation*" approved at the ITU-T Study Group 16 meeting held in Macau, China, 16-27 October 2017.

Editors:

Daidi Zhong
Chongqing University
P.R.China
Tel: +86-13696454858
E-mail: daidi.zhong@hotmail.com

Michael J. Kirwan
PCHA
USA
Tel: +1-913-207-826
Fax: +1-913-207-826
E-mail: mkirwan@pchalliance.org

Technical Paper HSTP-H812-FHIR

Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload for trial implementation

Summary

The Continua Design Guidelines (CDG) defines a framework of underlying standards and criteria that ensure the interoperability of devices and data used for personal connected health services. It also contains design guidelines (DGs) that further clarify the underlying standards or specifications by reducing options or by adding missing features to improve interoperability.

This specification defines guidelines for uploading measurements from a Personal Health Gateway (PHG) to a Health and Fitness Service (H&FS). The uploaded measurements are represented using a resource model consistent with that of HL7 Fast Healthcare Interoperability Resources (FHIR). Although measurements are uploaded using a different encoding and data model than defined in H.812.1, the information content of the delivered measurement is the same.

This Technical Paper is planned to be issued in the future as Recommendation ITU-T H.812.5 as part of the "ITU-T H.810 interoperability design guidelines for personal connected health systems" subseries that covers the following areas:

– ITU-T H.810 – Interoperability design guidelines for personal connected health systems: System overview

– ITU-T H.811 – Interoperability design guidelines for personal connected health systems: Personal health devices interface design guidelines

– ITU-T H.812 – Interoperability design guidelines for personal connected health systems: Services interface design guidelines

– ITU-T H.812.1 – Interoperability design guidelines for personal connected health systems: Services interface: Observation upload capability

– ITU-T H.812.2 – Interoperability design guidelines for personal connected health systems: Services interface: Questionnaires capability

– ITU-T H.812.3 – Interoperability design guidelines for personal connected health systems: Services interface: Capability exchange capability

– ITU-T H.812.4 – Interoperability design guidelines for personal connected health systems: Services interface: Authenticated Persistent Session capability

– ITU-T H.812.5 – Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload (planned)

– ITU-T H.813 – Interoperability design guidelines for personal connected health systems: Healthcare information system interface design guidelines

This Technical Paper is based on anticipated directions within the HL7 FHIR community. As such it is a trial implementation specification and subject to change based both on gained experience and final design decisions within HL7 for its FHIR specification.

This document uses the term "placeholder" when specific aspects, such as URIs, need to be in place but have not yet been established. See clause A.1.1.1 for additional information.

The placeholder markers in this document may be given final values at different points in time. The interested reader should check https://members.pchalliance.org/document/dl/1038 for the current status of the placeholders.

Table of Contents

Page

Page

Page

List of Tables

Page

Page

Page

List of Figures

Page

0 Introduction

The Continua Design Guidelines (CDG) defines a framework of underlying standards and criteria that ensure the interoperability of devices and data used for personal connected health. They also contain additional design guidelines that further clarify the underlying standards or specifications by reducing options or by adding missing features to improve interoperability.

This document defines guidelines for uploading measurements from a Personal Health Gateway (PHG) to a Health and Fitness Service (H&FS). The uploaded measurements are represented using a resource model consistent with that of HL7 Fast Healthcare Interoperability Resources (FHIR). Although measurements are uploaded using a different encoding and data model than defined in H.812.1, the information content of the delivered measurement is the same.

This document is part of the "ITU-T H.810 interoperability design guidelines for personal health systems" subseries. See [ITU-T H.810] for more details.

This version is based on anticipated directions within the HL7 FHIR community. As such it is a trial implementation specification (issued as an ITU-T Technical Paper) and subject to change based both on gained experience and final design decisions within HL7 for its FHIR specification.

This document uses the term "placeholder" when specific aspects, such as URIs, need to be in place but have not yet been established. See clause A.1.1.1 for additional information.

The placeholder markers in this document may be given final values at different points in time. The interested reader should check https://members.pchalliance.org/document/dl/1038 for the current status of the placeholders.

0.1 Organization

0.2 Organization

This Certified Capability Class (CCC) guideline is organized in the following manner:

Clause 0-5: Introduction and Terminology - Provides an overview of how H.812.5 is structured

Clause 6**: Use Cases –** A descriptive scenario that motivates the class of problems that FHIR observation upload is addressing.

Clause 7: FHIR Observation Upload Overview – A technical overview of the observation upload process.

Clause 8: Behavioural Model – Details of the observation upload process

Annex A: Mapping from ISO/IEEE 11073 Personal Health Device representation to FHIR resource representation

0.3 CDG Guideline Releases and Versioning

Information on releases and versioning of these guidelines can be found in Clause 0.2 of [H.810]

0.4 What's New

Initial Release of the H.812.5 CDG document.

Technical Paper HSTP-H812-FHIR

Interoperability design guidelines for personal health systems: Services interface: FHIR Observation Upload for trial implementation

1 Scope

This guidelines document defines four Continua Certified Capability Classes associated with uploading a measurement using the FHIR data model defined by HL7 [HL7-FHIR-MODEL]. Two of the capability classes address uploading a sensor measurement when the H&FS supports a FHIR server. In the context of this document a FHIR server is a H&FS that exposes the FHIR API defined by HL7[HL7-FHIR-API]. The remaining two capability classes document how to upload a sensor measurement to a H&FS that does not support a FHIR server. In this case, the FHIR data model is used, but no assumption is made relative to the FHIR server.

The Continua Certified Capability Classes defined in this document are:

– FHIR Observation Server – A H&FS that exposes the HL7 FHIR API, and can receive a sensor measurement from a FHIR Observation Client.

– FHIR Observation Client – A PHG that uses the exposed HL7 FHIR API of the H&FS, in a manner defined herein, to transfer sensor measurements.

– FHIR Observation Reporting Server – A H&FS that requires the complete context of a measurement to be contained in the received application data packet. In FHIR this means that the received message contains a complete bundle. A complete bundle is one in which all resources associated with the measurement are present.

– FHIR Observation Reporting Client– A PHG that bundles all resources associated with a given sensor measurement into a single application data packet.

2 References

All referenced documents can be found in Clause 2 of [H.810]

[H.810] Recommendation ITU-T H.810 (2017), *Interoperability design guidelines for personal connected health systems: Introduction*.

3 Definitions

This document uses terms defined in [H.810]

4 Abbreviations and Acronyms

This document uses abbreviations and acronyms defined in [H.810]

5 Conventions

This document follows the conventions defined in [H.810]

6 FHIR Use Cases

In the Continua Design Guidelines (CDGs), capability classes are created to address use cases that meet specific market needs. The four Fast Healthcare Interoperability Resources (FHIR) capabilities classes defined in this document address uploading measurements from a Personal Health Gateway (PHG) to a Health & Fitness Service (H&FS). In that aspect, the FHIR capability classes defined herein serve the same purpose as the capability classes defined in H.812.1. The FHIR capability classes, however, address the additional market requirement for alignment with the wider industry movement toward the use of JavaScript Object Notation (JSON), the FHIR data model, and REST oriented service APIs.

These Guidelines define four FHIR capability classes to address two different business use cases.

The FHIR Observation Server and the FHIR Observation Client are employed when the H&FS supports a FHIR server. This mode of operation is designed for the common use case (e.g. Patient Health Record, Document Sharing, Decision Support), and if employed properly is expected to be more efficient in terms of the network bandwidth consumed for a given upload.

In some business applications, it is not desirable to store patient health information in a FHIR server, but it is still desirable to use the FHIR data model. In this case, the FHIR Observation Reporting Server and FHIR Observation Reporting Client are used to bundle all aspects of a measurement into a single message. The bundling process defined for these capability classes requires that the PHG, acting as the FHIR Observation Reporting Client, place all the FHIR resources needed for a measurement in a single bundle, no external references are allowed. Since all needed information is contained in the bundle, the H&FS, acting as the FHIR Observation Reporting Server, can forward or translate the sensor message without needing to access patient information in a FHIR server. The FHIR Observation Reporting capability classes allow for business relationships to be formed in which the H&FS is only partially trusted.

6.1 Managing Patient Identity

When a H&FS supports a FHIR server, the PHG must properly reference the patient resource in any observation resource being uploaded. In the FHIR protocol, this is done by having the PHG provide the Logical ID of the patient resource to the FHIR server. How the PHG obtains the Logical ID of the patient resource may be a challenge for a deploying organization. The scenarios in this clause highlight supported methods by which this Logical ID can be obtained. Other methods may work as well.

NOTE – These scenarios apply only to the FHIR Observation Client and FHIR Observation Server.

6.1.1 Scenario #1

In this scenario illustrated in Figure 6-1, the administrative team responsible for the H&FS provides the Logical ID for the Patient Resource. The Logical ID is communicated to the party responsible for configuring the PHG in the home environment. Once the PHG is configured with the appropriate logical identification of the Patient Resource, a measurement can be uploaded using the configured Logical ID of the Patient Resource. An important aspect of this scenario is that the PHG never needs any personal information, and personal information is never seen on the wire.

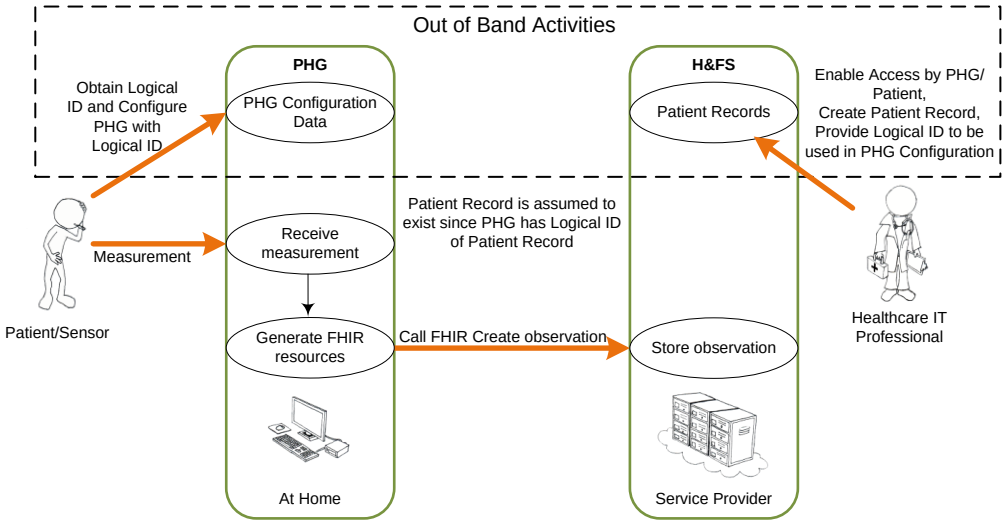

Figure 6-1 – Scenario #1 FHIR Observation Upload

6.1.2 Scenario #2

In scenario 2, the administrative team for the H&FS does not provide the patient with the FHIR Logical ID of the Patient Resource. Instead, other information, which identifies the patient and is herein called the Patient Designator, is used to establish the identity of the patient in the uploading of a measurement.

NOTE – An insurance card where the card's issuer (an insurance company) is an assigning authority, and the account number on the card identifies the patient to the insurance company) is an example of a patient designator. The backend FHIR server would be configured with this information in *Patient.identifier.system* and *Patient.identifier.value* within the Patient Resource.

The Patient Designator can represent a wide range of different forms of patient identity, appropriate to different situations. The Patient Designator contains information about who the assigning authority is, and how the patient is identified by that assigning authority. In general, the Patient Designator is a method some organizations may use to identify a patient, and by itself does not expose personal information. In many cases, the Patient Designator may be familiar to the patient through other communications with the organization. As shown in Figure 6-2, the Patient Designator is used to generate a FHIR Patient Resource and the PHG takes the responsibility of specifying the Logical ID of this resource, which must be unique when used in the FHIR server. The PHG uses the generated Logical ID to create a Patient Resource on the FHIR server, or to confirm the pre-existence of the Patient Resource. Once the PHG has obtained and validated the Logical ID of the Patient Resource on the FHIR server, it can upload measurements and reference the Patient Resource via the Logical ID.

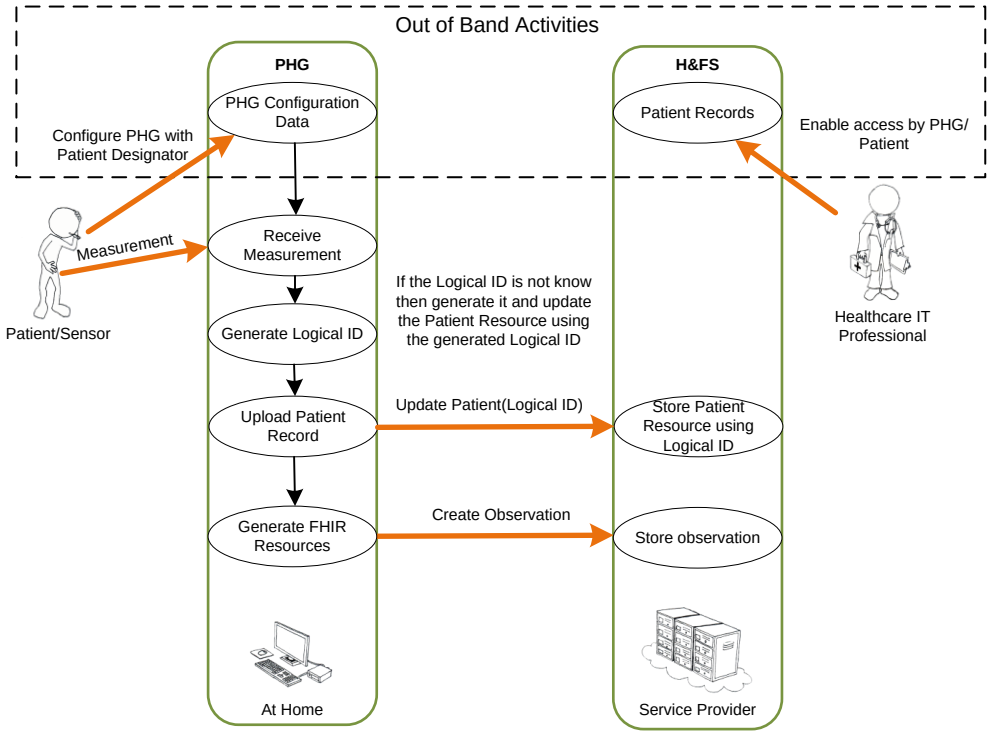

Figure 6-2 – Scenario #2 FHIR Observation Upload

6.1.3 Scenario #3

In the third scenario, the patient is provided with the Patient Designator, but in contrast to scenario 2, the Patient Resource Logical ID is generated by the FHIR server on the H&FS. The PHG obtains the generated Logical ID by identifying the Patient Resource using the Patient Designator. The patient resource must have been previously provisioned on the FHIR server, or the PHG must be allowed to create a new Patient Resource. The patient experience is the same in this scenario as in scenario 2. This scenario is included since an organization running an H&FS may not be comfortable with the idea of the PHG defining the Logical ID for a Patient Resource on the FHIR server for which they are responsible.

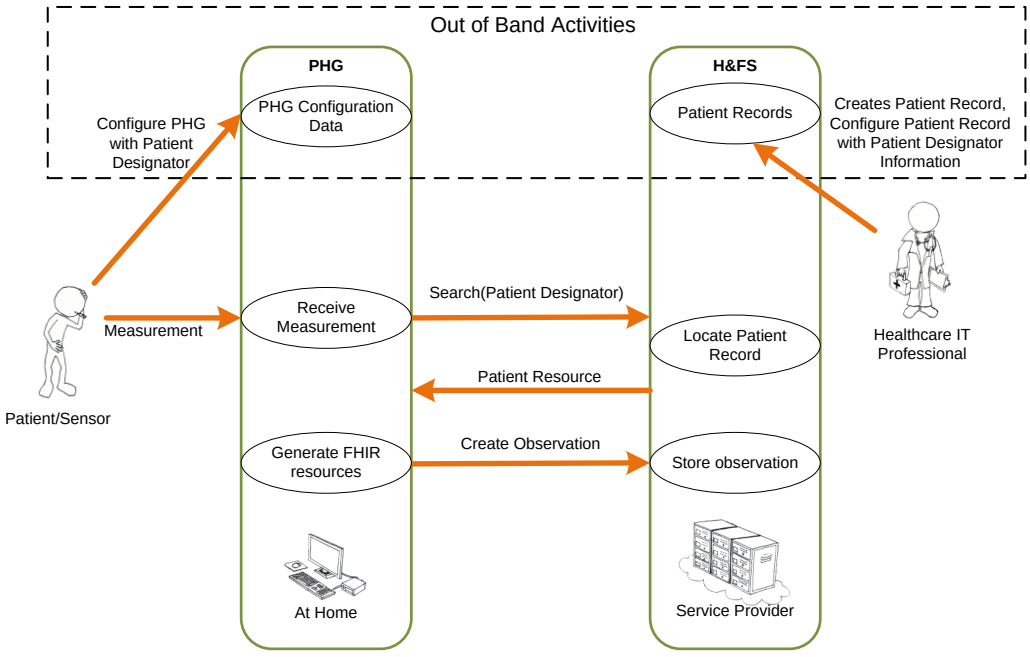

Figure 6-3 – Scenario #3 FHIR Observation Upload

6.1.4 Scenario #4

In this final scenario, the identification of the patient resource and its associated Logical ID is established using OAuth. Inherent in the use of OAuth is an authentication procedure of some type that defines the access rights granted to the PHG. The authentication procedure establishes a user identity that is granted access to the resources associated with a specific patient record. A FHIR GET operation requesting all patient resources returns nothing or a single patient resource and its associated Logical ID since the PHG is only authorized to see one patient record. The Logical ID is then used to upload the measurement. The primary advantage of this approach is that the patient does not have to be provided with a Patient Designator or a Logical ID.

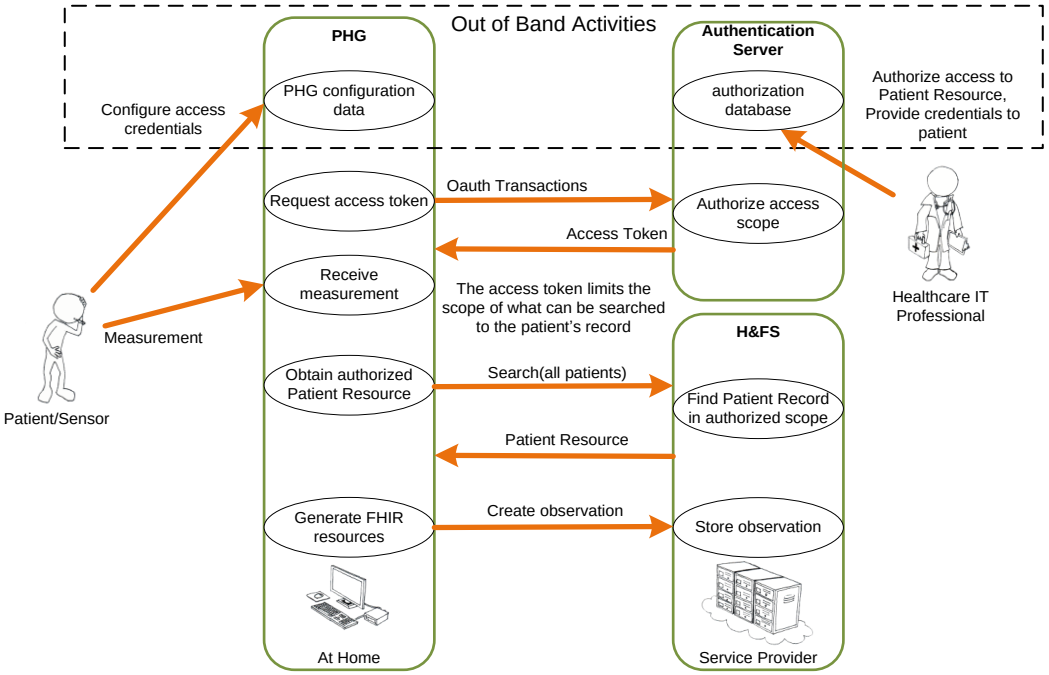

Figure 6-4 – Scenario #4 FHIR Observation Upload

7 Introduction (Informative)

The HL7 FHIR specifications define a collection of resources that can be used to support a broad spectrum of Healthcare workflows. This CDG document specifies how to use the HL7 FHIR resources to model observations that can be received from a Continua PHD, and how to securely upload the observations to an H&FS. This CDG also identifies how to map the ISO/IEEE 11073 fields and values into the corresponding FHIR resource representation.

The FHIR Certified Capabilities Classes are defined to be part of the Services Interface. As depicted in the Continua Architecture (Figure 7-1**)**, the Services Interface is associated with communication between the PHG and the H&FS.

NOTE – The reader should be aware that the Continua Architecture represents a functional architecture created for defining behaviour between components. It does not define how a given system is to be deployed. For instance, a logical PHG may reside inside a sensor device.

Figure 7-1 depicts FHIR usage within the Continua Architecture.

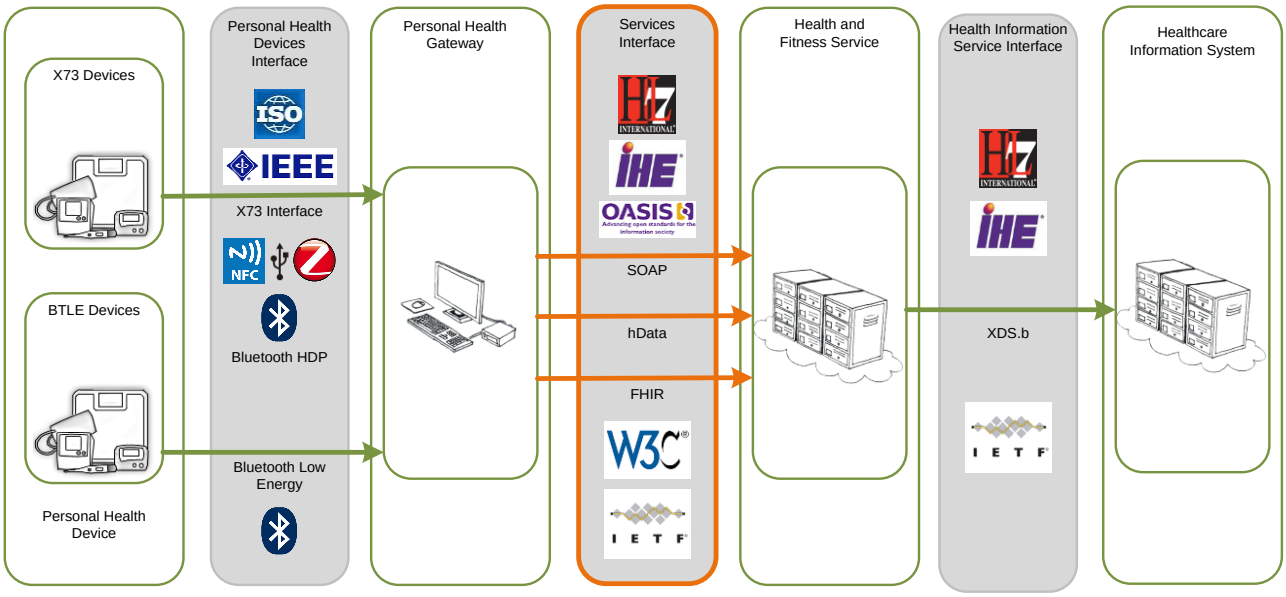

Figure 7-1 – Continua Reference Architecture

A FHIR observation upload is normally preceded by a PHD delivering an observation to a PHG. The communications between the PHD and the PHG allows two or more ISO/IEEE 11073 objects types to be constructed by the PHG. The first object type is the Medical Device System (MDS) of the PHD, which provides information about the PHD. The second object type is the ISO/IEEE 11073 metric object type which represents the observation (measurement) taken by the PHD. There may be multiple metric objects types from a single measurement process of a PHD.

The PHG maps the ISO/IEEE 11073 objects types from the PHD along with its own MDS [H.812.1] and configured patient information to FHIR resources. The mapping may create instances of three FHIR resource types:

– Patient resource: Demographic and administrative information about the patient.

– DeviceComponent resource: Characteristics, operational status and capabilities for [a component of] a healthcare device.

– Observation resource: Measurements or assertions about a subject. Can be a patient or a device.

See Annex A for detailed information on mapping between ISO/IEEE 11073 objects and FHIR resources.

7.1 Security Framework

FHIR resources are uploaded to the H&FS in the context of an encrypted TCP connection with authorization to upload being provided through OAuth [OAuth 2.0]. To upload the FHIR resources, the PHG must have an OAuth bearer token. If it does not have a valid bearer token, the PHG communicates with the OAuth Server to obtain one.

Continua certification is designed to ensure interoperability between components marketed by different organizations. To ensure that a minimum level of interoperability is possible when OAuth is used for securing the uploading of measurements with FHIR, a certified PHG is required to support one of: (1) client credential, (2) resource owner, (3) authorization code, or (4) implicit grant types. A certified H&FS is required to support both the client credential and the resource owner credential authorization grants. Further, if the H&FS claims to work with browser based PHGs, then it must also support Authorization Code and Implicit grant types.

NOTE – Continua certification means that the certified product has support for the required grant types and provides a mechanism by which the grant types could be enabled. Continua certification does not mean that a certified product, when deployed, must enable that support.

In addition to the required OAuth grant types, this specification profiles the usage of the *Assertion Framework for OAuth Client Authentication* [OAuth Assertion] in conjunction with *JSON Web Token Profile for OAuth Client Authentication and Authorization Grants* [OAuth JWT] to enable assertion based authorization of PHGs. The use of JSON web token based OAuth authorization is optional for both the PHG and the H&FS.

Although OAuth provides a standardized mechanism by which authorization can be granted to a PHG for uploading a message, it does not define how the FHIR resources within the H&FS are secured. The business logic associated with a given H&FS is ultimately responsible for ensuring the security of the information in the H&FS; this CDG document does not specify how FHIR security is achieved by the H&FS.

7.2 Example Uploads from a FHIR Observation Client

The examples in this section illustrate the exchanges that would take place between a PHG and a H&FS for each of the different scenarios listed in clause 6. The examples assume the use of a resource owner access grant in the OAuth exchange, and that the PHG is using Capability Exchange. The Examples in this section also assume that the upload is taking place between a FHIR Observation Client and FHIR Observation Server.

To simplify the sequence diagrams neither the Capability Exchange nor the OAuth exchange sequences are shown in full. For reference, the full steps are listed below.

Capability Exchange:

1. The PHG is configured with the URL base address where it can obtain the root.xml file.
2. The PHG issues a GET to obtain the root.xml file.
3. The PHG builds the URL to the atom feed from the content of the root.xml and issues an http GET to obtain the atom feed for the OAuthDescriptors.
4. The PHG uses the link reference in entry element of the returned atom feed to obtain the URL of the OAuthDescriptor itself.
5. The PHG issues a GET to obtain the OAuthDescriptor

OAuth Exchange when using a Resource Owner Grant type:

1. The PHG OAuth client application has been pre-configured with a client_id and associated client_secret.
2. The PHG OAuth client application sends an OAuth Authorization request to the resource owner.

   NOTE – This step may be implemented by the PHG OAuth client code looking up configuration information that has been provided by the resource owner, or by the PHG OAuth client code displaying an input window in which the resource owner enters credential information. This document does not require that the PHG implement a visible OAuth exchange to an authorization server's authorization endpoint.
3. The PHG OAuth client code obtains the resource owner credentials
4. The PHG issues a request to the authentication server's token endpoint with the grant type parameter set to indicate the user of a resource owner password. The resource owner's credentials are provided along with the client_id and client_secret, and optionally the scope.

5. The authentication server returns the bearer token and possibly modified scope to the PHG OAuth client code.

In clauses 7.2.1 – 7.2.4, each scenario identified in clause 6 is illustrated with numbered steps and a sequence diagram (Figure 7-2 to Figure 7-5).

7.2.1 Scenario #1

1. The H&FS is configured with a Patient Resource, generating a FHIR Logical ID.
2. The PHG is loaded with the FHIR Logical ID for the Patient Resource created on the H&FS, the H&FS URLs, the client id, the resource owner id, and the password.
3. The PHG performs Capability Exchange with the H&FS to discover the H&FS FHIR upload capabilities.
4. An observation is taken on the health sensor; the sensor communicates the measurement to the PHG.
5. The PHG receives the measurement and associates the measurement with a patient to obtain the Patient Record Logical ID.
6. The PHG recognizes that it does not have a valid access token or refresh token. The PHG sends a resource owner grant request to the OAuth authorization server's token endpoint which includes the client id, the resource owner credentials, and the scope.
7. The OAuth token endpoint validates the credentials in the context of the scope and sends an authorization grant in the form of a bearer token.
8. The PHG creates the necessary FHIR resources from the received sensor measurement and internal PHG data.
9. The created FHIR resources are uploaded to the FHIR resource server URL obtained from the Capability Exchange Process.
10. The H&FS validates the authorization using the bearer token, and if access is granted stores the FHIR resources in the context identified by the Patient Resource Logical ID.
11. The H&FS server returns the appropriate HTTP response.

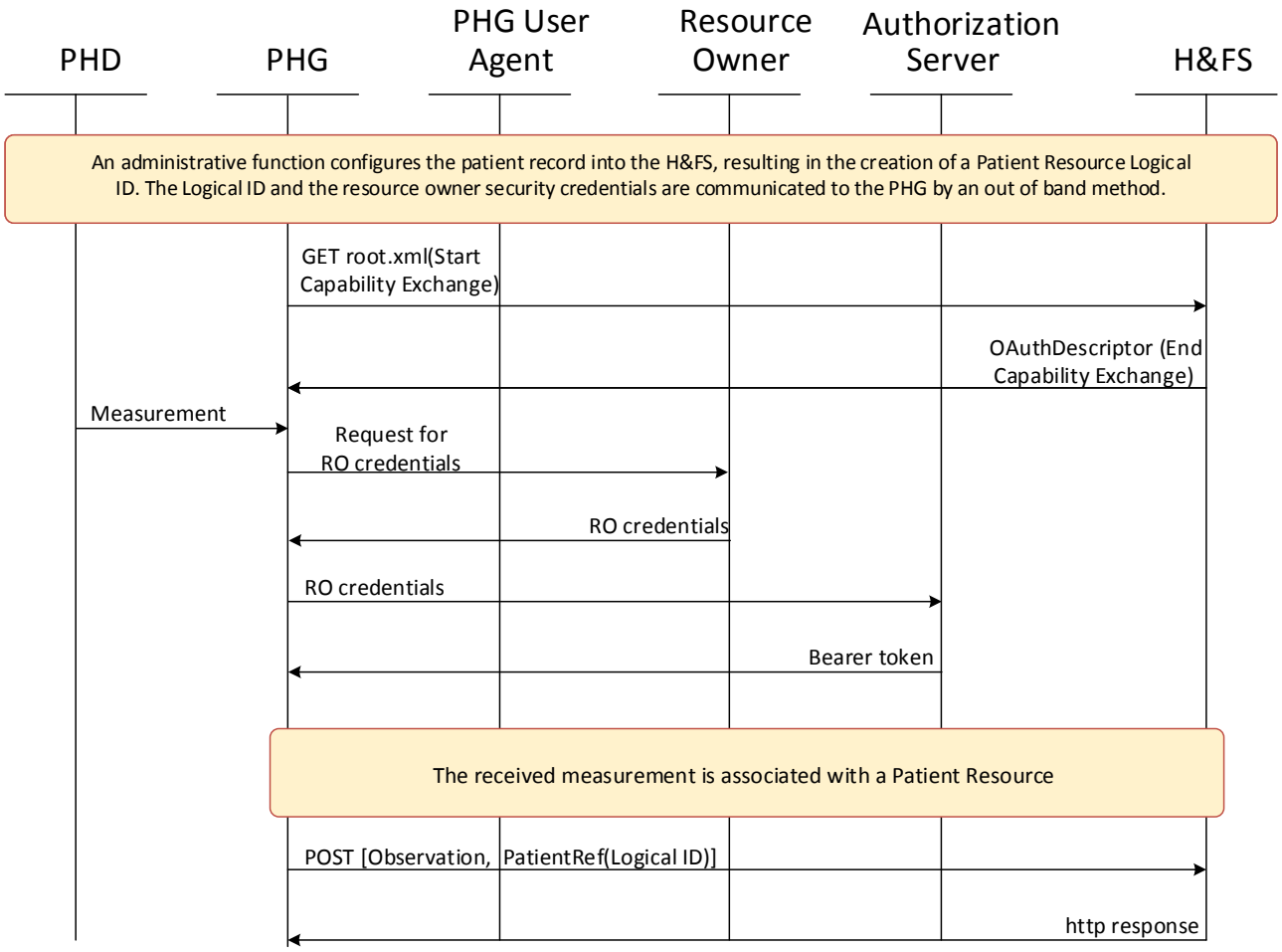

Figure 7-2 – Scenario #1 Sequence Diagram

7.2.2 Scenario #2

1. The PHG is loaded with the Patient Designator, the H&FS URLs, the client id, the resource owner id, and the password.
2. The PHG performs Capability Exchange with the H&FS to discover the H&FS FHIR upload capabilities.
3. The PHG receives a measurement from the sensor.
4. The PHG recognizes that it does not have a Logical ID for the patient associated with the measurement and creates a FHIR Patient Resource, generating a Logical ID in the process.
5. The PHG recognizes that it does not have a valid access token or refresh token. The PHG sends a resource owner grant request to the OAuth authorization server's token endpoint which includes the client id, the resource owner credentials, and the scope.
6. The Authentication Server validates the credentials in the context of the scope and sends an authorization grant in the form of a bearer token.
7. The PHG sends the Patient Resource to the H&FS in conjunction with the access token.
8. The PHG creates the necessary FHIR resources to upload the measurement from the received sensor measurement and internal PHG information.
9. The FHIR resources are uploaded to the URL obtained from the Capability Exchange Process using the Logical ID of the Patient Resource to provide patient context.

10. The H&FS validates the authorization using the bearer token, and if access is granted stores the FHIR Observation Resource, and, if provided, the DeviceComponent resource in the context identified by the Patient Resource Logical ID.

11. The H&FS server returns the appropriate HTTP response.

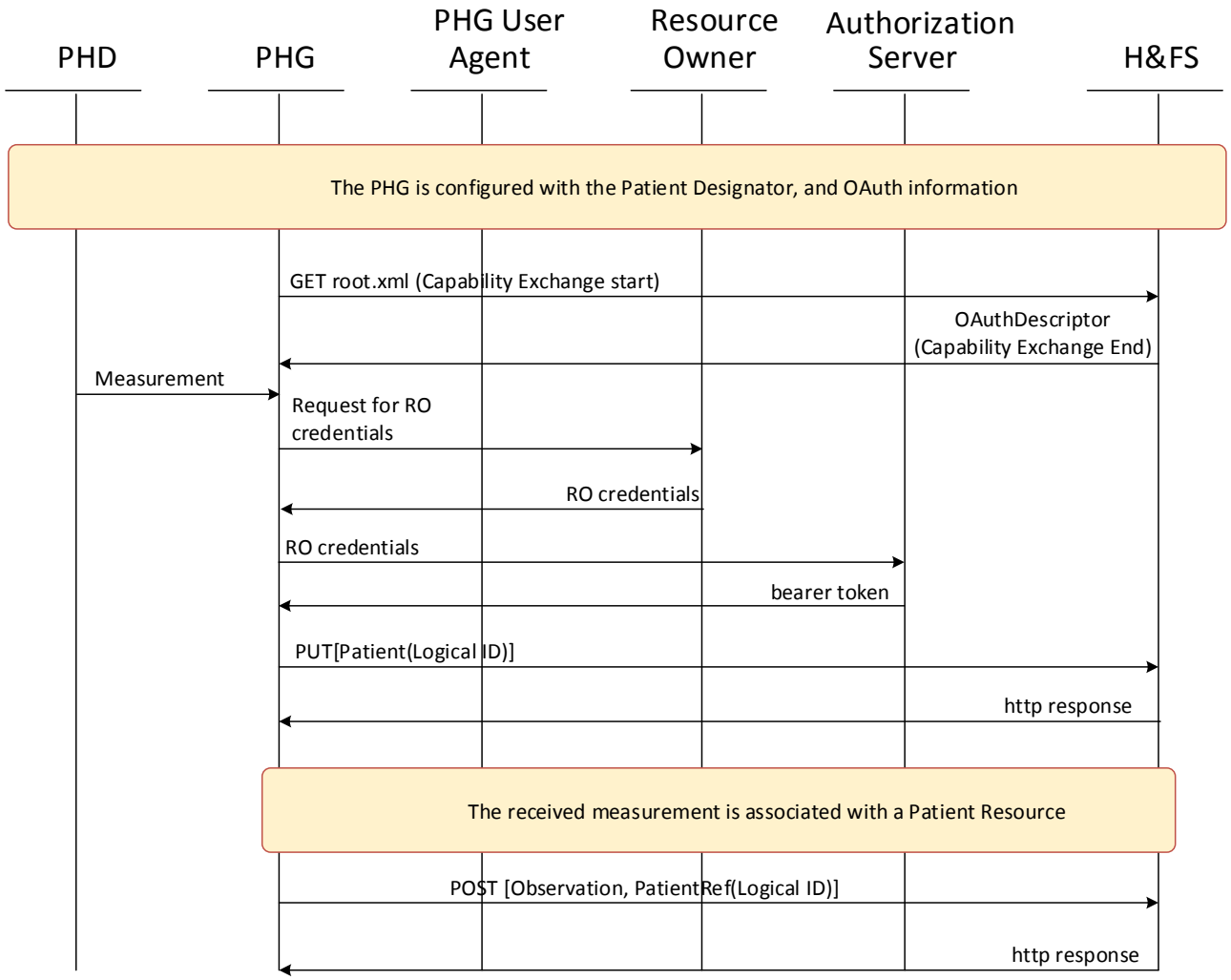

Figure 7-3 – Scenario #2 Sequence Diagram

7.2.3 Scenario #3

1. The H&FS is configured with a Patient Resource, generating a FHIR Logical ID.

2. The PHG is loaded with the Patient Designator, the H&FS URLs, the client id, the resource owner id, and the password.

3. The PHG performs Capability Exchange with the H&FS to discover the H&FS FHIR upload capabilities.

4. The PHG recognizes that it does not have a valid access token or refresh token. The PHG sends a resource owner grant request to the OAuth authorization server's token endpoint which includes the client id, the resource owner credentials, and the scope.

5. The Authentication Server validates the credentials in the context of the scope and sends an authorization grant in the form of a bearer token.

6. A measurement is taken on the health sensor; the sensor communicates the measurement to the PHG.

7. The PHG receives the measurement and associates the measurement with a patient.

8. The PHG creates the necessary FHIR resources from the received sensor measurement and internal PHG data.
9. The PHG retrieves the Patient Resource from the H&FS using the Patient Designator information and extracts the Patient Resource Logical ID.
10. The PHG uses the Patient Resource Logical ID in building the Observation Resource, and if needed, DeviceComponent resource. These resources are uploaded to the URL obtained from the Capability Exchange Process.
11. The H&FS validates the authorization using the bearer token, and if access is granted stores the FHIR Observation Resource, and, if provided, the DeviceComponent resource in the context identified by the Patient Resource Logical ID.
12. The H&FS server returns the appropriate HTTP response.

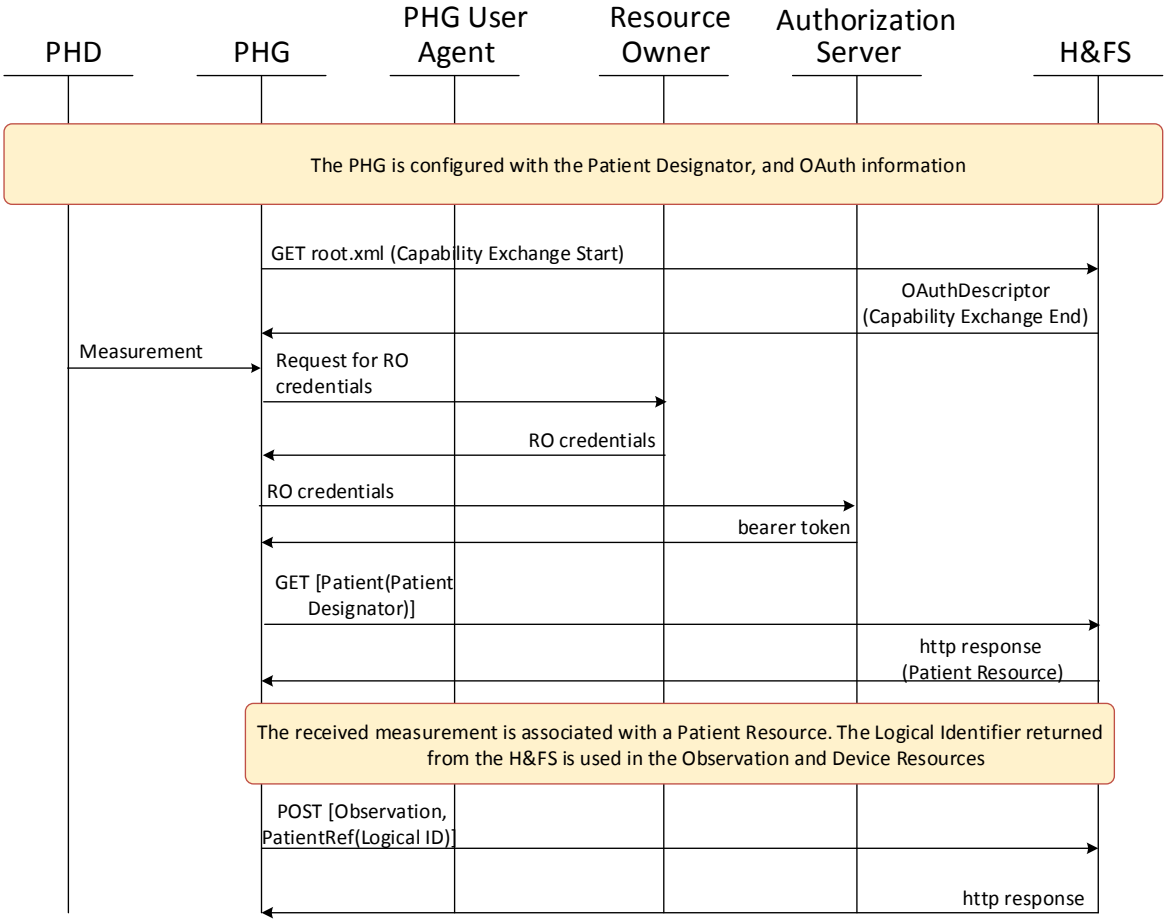

Figure 7-4 – Scenario #3 Sequence Diagram

7.2.4 Scenario #4

1. The H&FS is configured with a Patient Resource, generating a FHIR Logical ID. An authorized scope is established in which there is only one FHIR patient resource.
2. The PHG is loaded with the H&FS URLs, the client id, the resource owner id, and the password.
3. The PHG performs Capability Exchange with the H&FS to discover the H&FS FHIR upload capabilities.
4. The PHG recognizes that it does not have a valid access token or refresh token. The PHG sends a resource owner grant request to the OAuth authorization server's token endpoint which includes the client id, the resource owner credentials, and the scope.

5. The Authentication Server validates the credentials in the context of the scope and sends an authorization grant in the form of a bearer token.
6. A measurement is taken on the health sensor; the sensor communicates the measurement to the PHG.
7. The PHG receives the measurement and associates the measurement with a patient.
8. The PHG creates the necessary FHIR resources from the received sensor measurement and internal PHG data.
9. The PHG retrieves the available Patient Resource from the H&FS based on its authorized scope of access and extracts the Patient Resource Logical ID.
10. The PHG uses the Patient Resource Logical ID in building the Observation Resource, and if needed, DeviceComponent resource. These resources are uploaded to the URL obtained from the Capability Exchange Process.
11. The H&FS validates the authorization using the bearer token, and if access is granted stores the FHIR Observation Resource, and, if provided, the DeviceComponent resource in the context identified by the Patient Resource Logical ID.
12. The H&FS server returns the appropriate HTTP response.

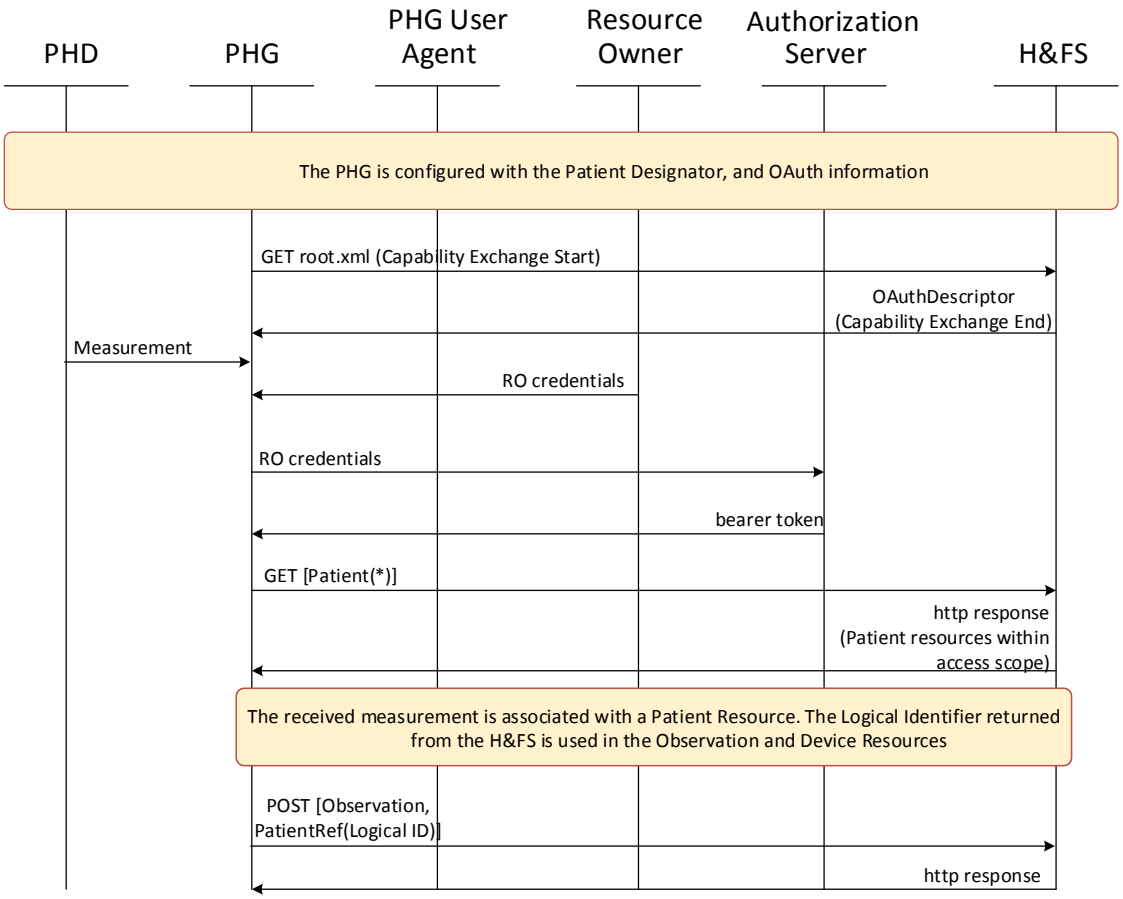

Figure 7-5 – Scenario #4 Sequence Diagram

7.3 Example Upload from a FHIR Observation Reporting Client

A PHG supporting only a FHIR Observation Reporting Client, or communicating with a H&FS that only supports a FHIR Observation Reporting Server does not need to obtain a consistent understanding of the patient resource Logical ID. There is, therefore, only one use case to consider, which is a simple upload of a complete measurement in which all the FHIR resources are provided in a single FHIR bundle.

7.3.1 Bundle Upload

1. The PHG is loaded with the H&FS URLs, the client id, the resource owner id, and the password.
2. The PHG performs Capability Exchange with the H&FS to discover the H&FS FHIR upload capabilities.
3. The PHG recognizes that it does not have a valid access token or refresh token. The PHG sends a resource owner grant request to the OAuth authorization server's token endpoint which includes the client id, the resource owner credentials, and the scope.
4. The Authentication Server validates the credentials in the context of the scope and sends an authorization grant in the form of a bearer token.
5. A measurement is taken on a Personal Health Device (PHD); the PHD communicates the measurement to the PHG.
6. The PHG receives the measurement and associates the measurement with a patient.
7. The PHG creates the necessary FHIR resources from the received sensor measurement and internal PHG data. The internal PHG data includes the Patient Designator. The created FHIR resources are bundled into a single application level packet data unit.
8. These resources are uploaded to the URL obtained from the Capability Exchange Process.
9. The H&FS validates the authorization using the bearer token, and if access is granted accepts the message
10. The H&FS server returns an HTTP response indicating acceptance or rejection of the message.

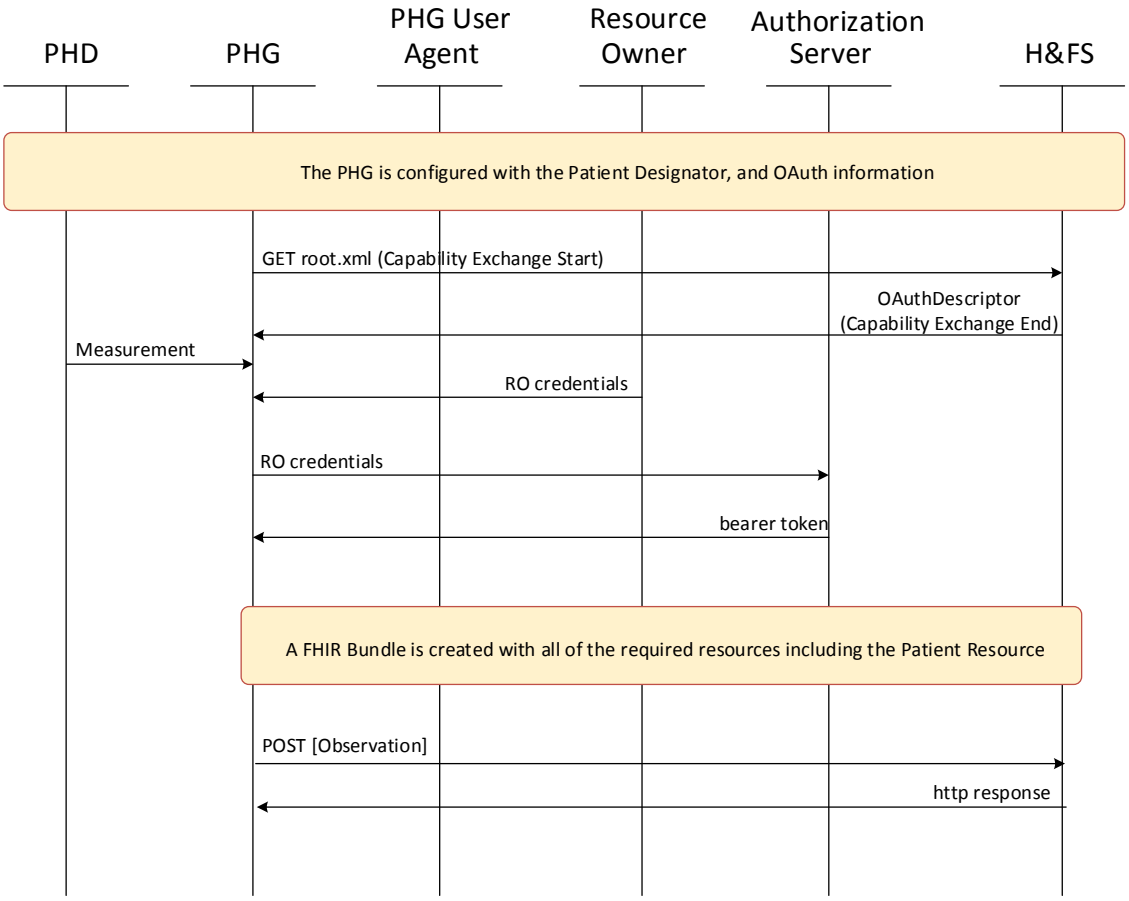

Figure 7-6 – Upload with Complete Bundle

7.4 Use of JWT

The Jason Web Token Profile for OAuth 2.0 Client Authentication and Authorization Grants [OAuth JWT] can be used to allow a PHG to access resources on a H&FS based on an existing trust relationship without having a direct user-approval step or exposing confidential information to the PHG. To take advantage of the benefits of the JWT, the signing certificate must be loaded onto the PHG. These Guidelines do not proscribe a method by which the PHG is provisioned with a signing certificate. A common pattern, however, is to use a security token service (STS). A STS requires some type of trust relationship with an organization providing the signing certificate, which is typically manifested by the exchange of key material with the STS. WS-Trust [OASIS.WS-Trust] is one available standard for a security token service.

NOTE – If the motivation for using a JWT is to avoid disclosure of a key to the PHG, then the use of a STS may not be desirable as the STS typically requires the PHG to have a key. To address this case, a manual method for configuring the PHG with the JWT is suggested.

8 Behavioural Model (Normative)

This clause provides implementation guidelines for conformant operation of a:

– FHIR Observation Server Continua Certified Capability Class

– FHIR Observation Client Continua Certified Capability Class

– FHIR Observation Reporting Server Continua Certified Capability Class

– FHIR Observation Reporting Client Continua Certified Capability Class

In the Continua Architecture, the FHIR clients operate in the context of a PHG and communicate with a FHIR server operating in the context of a H&FS. Exchanges between a PHG and a H&FS take place over the service interface, and the H&FS exposes these capabilities via Capability Exchange.

8.1 Capability Exchange

A H&FS with a FHIR Observation Server or a FHIR Observation Reporting Server **shall** implement Capability Exchange as specified in [H.812.3]. The root.xml file for a system supporting a FHIR Observation Reporting Server is the same as the root.xml on a system supporting a FHIR Observation Server, except for the name of the Continua Certified Capability Class. Knowing the supported Continua Certified Capability Class is sufficient for the PHG to properly upload observations.

The OAuthDescriptor is the resource exposed in Capability Exchange that enables the PHG to upload FHIR observations to the H&FS. The OAuthDescriptors on a H&FS are made visible via an Atom Feed in Capability Exchange. A PHG **shall** support Capability Exchange, including the ability to obtain the OAuthDescriptor, as defined in [H.812.3].

The root.xml file for a FHIR Observation Reporting Server is given in Figure 8-1.

```
<profile>
  <!—The location of the document (HCP) describing this profile -->
  <id>FHIR-Observation-Reporting-Server-4C</id>
  <reference>
    http://handle.itu.int/11.1002/3000/hdata/fhir/2017/01/h.812.5.pdf
  </reference>
</profile>
<resourceType>
  <resourceTypeID>OAuthDescriptor</resourceTypeID>
  <!—reference document for OAuthDescriptor  -->
  <reference>
    http://handle.itu.int/11.1002/3000/hdata/fhir/2017/01/h.812.5.pdf
  </reference>
  <representation>
    <mediaType>application/json</mediaType>
  </representation>
</resourceType>
<section>
  <!—Relative path to OAuthDescriptor resource -->
  <path>atom feed/of/OAuthDescriptor</path>
  <profileID>FHIR-Observation-Reporting-Server-4C</profileID>
  <resourceTypeID>OAuthDescriptor</resourceTypeID>
  <resourcePrefix>true</resourcePrefix>
</section>
```

Figure 8-1 – root.xml components for a FHIR Observation Reporting Server

The root.xml file for a FHIR Observation Server is given in Figure 8-2.

```
<profile>
  <!—The location of the document (HCP) describing this profile -->
  <id>FHIR-Observation-Server-4C</id>
  <reference>
    http://handle.itu.int/11.1002/3000/hdata/fhir/2017/01/h.812.5.pdf
  </reference>
</profile>

<resourceType>
  <resourceTypeID>OAuthDescriptor</resourceTypeID>
  <!—reference document for OAuthDescriptor  -->
  <reference>
    http://handle.itu.int/11.1002/3000/hdata/fhir/2017/01/h.812.5.pdf
  </reference>
  <representation>
    <mediaType>application/json</mediaType>
  </representation>
</resourceType>

<section>
  <!—Relative path to OAuthDescriptor resource -->
  <path>atom feed/of/OAuthDescriptor</path>
  <profileID>FHIR-Observation-Server-4C</profileID>
  <resourceTypeID>OAuthDescriptor</resourceTypeID>
  <resourcePrefix>true</resourcePrefix>
</section>
```

Figure 8-2 – root.xml components for a FHIR Observation Server

The Atom Feed for the OAuthDescriptor, shown in Figure 8-3, applies to both servers.

```
<?xml version="1.0" encoding="UTF-8" ?>
    <feed>
        <title>H.812.5 OAuthDescriptor Resource</title>
        <link rel="self" href="https://dev1.pcha.com/pchaFhir/hData" />
        <author>
            <name>CODE for Healthcare</name>
        </author>
        <id>https://dev1.pcha.com/pchaFhir/hData</id>
        <updated>2017-03-03T00:22:02Z</updated>
        <entry>
            <title>CODE for Healthcare OAuthDescriptor</title>
            <link rel="alternate"
                  href="https://dev1.pcha.com/pchaFhir/hData/@1" />
            <id>https://dev1.pcha.com/pchaFhir/hData/@1</id>
            <updated>2017-06-03T00:22:02Z</updated>
            <summary type="text">OAuthDescriptor</summary>
        </entry>
     </feed>
```

Figure 8-3 – Atom Feed for OAuthDescriptor

A H&FS with a FHIR Observation Reporting Server or a FHIR Observation Server **shall** provide the OAuthDescriptor in JSON or XML format, and **shall** set the value of <mediaType> element in the <resourceType> element for a OAuthDescriptor to either application/json or application/xml based on the representation used.

8.1.1 OAuthDescriptor

The OAuthDescriptor is a resource passed from the H&FS to the PHG during CapabilityExchange. The informational content of the OAuthDescriptor allows the H&FS to provide configuration information to the PHG which simplifies the user experience.

Table 8-1 – Elements of the OAuthDescriptor resource provided by H&FS Application

Element	Usage
resourceServerURL	The resourceServerURL is the endpoint where the H&FS expects to receive measurements from the PHG. This element **shall** be present
authorizationEndpointURL	The Authorization server's authorizationEndpointURL is the OAuth endpoint the H&FS expects the PHG to use for OAuth authorization. This element **may** be present
tokenEndpointURL	The tokenEndpointURL is the OAuth endpoint the H&FS expects the PHG to use for obtaining the access bearer token. This element **shall** be present
grantTypes	The grantTypes element in the OAuthDescriptor **shall** contain a list of one or more strings selected from the following: "clientCredential", "resourceOwnerCredential", "implicit", "authorizationCode", "rfc7523". The grantTypes element **shall** be present in the OAuthDescriptor.

An example JSON OAuthDescriptor resource is shown in Figure 8-4.

```
{
    "grantTypes": ["resourceOwnerCredential", "rfc7523"],
    "tokenEndPointURL": "http://handle.itu.int/11.1002/3000/hdata/fhir/oauth/token",
    "resourceServerURL": "http://handle.itu.int/11.1002/3000/hdata/fhir/receiver"
}
```

Figure 8-4 – An Example JSON OAuthDescriptor Resource

8.2 OAuth Usage

This clause provides guidance on the use of OAuth for enabling authorized uploads of device measurements from a PHG to a H&FS. The intent of this clause is to provide sufficient specification, that when properly configured, a PHG that supports one of the Continua Certified Capability Classes defined herein for a PHG **shall** be able to securely upload measurements to a Health and Fitness Service that supports either the FHIR Observation Server or the FHIR Observation Reporting Server.

8.2.1 OAuth Support

All Continua Certified Capability Classes defined herein **shall** support [OAuth 2.0].

8.2.2 Authorization Grants

To foster interoperability this guideline document profiles the use of OAuth authorization grant types. There are five authorization grant types and their typical usage is summarized in Table 8-2.

Table 8-2 – Authorization Grant Usage

Authorization Grant Type	Typical Usage
client credentials	The trust relationship is with the application itself. The security credentials are known to the PHG application and are independent of the resource owner (user) of the application.
resource owner credentials	The trust relationship is with a resource owner. The security credentials may be provided by the user of the software through a UI, or via a provisioning or configuration process. The security credentials are exposed to the PHG application. Resource Owner credential could be used with a PHG that does not directly interact with a user.
authorization code	The trust relationship is with a resource owner. The PHG application does not gain access to the security credentials. The resource owner needs to be involved in the workflow.
implicit	The trust relationship is with a resource owner. The PHG application does not gain access to the security credentials. The resource owner needs to be involved in the workflow. Similar to authorization code except designed for use with JavaScript based clients in web browsers.
Extension grant (urn:ietf:params:oauth:grant-type:jwt-bearer)	Allows a security certificate to be used on a PHG. Security certificates may provide a better mechanism to manage security credentials as compared to a username/password based schemes.

A PHG **shall** support the use of one or more of the following OAuth authorization grant types:

– Resource Owner Credentials

– Client Credentials

The PHG **may** support the use of authorization code and implicit grant types.

Additionally, the PHG **may** also support the use of the extension grant with a grant_type value of: "urn:ietf:params:oauth:grant-type:jwt-bearer". This grant type is used when the PHG supports *JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants* [OAuth JWT].

A PHG can use a JWT authorization grant type when it is more desirable to distribute a certificate.

If a PHG specifies a grant_type value of "urn:ietf:params:oauth:grant-type:jwt-bearer" it **shall** conform to the requirements of [OAuth JWT].

A H&FS **shall** support each of the following Authorization Grant types:

– Resource Owner Credential

– Client Credentials

If the H&FS is intended to be used with platforms that support web or interactive interfaces it **shall** support the Authorization Code and implicit grant type.

Additionally, the H&FS **may** support the use of *JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants* [OAuth JWT]. If the H&FS supports *JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants* it **shall** indicate this in Capability Exchange by including the value "rfc7523" for the grantTypes.

8.2.3 Client Registration

How a PHG's OAuth client registers with the authentication endpoint as defined in [OAuth 2.0] is out of scope.

8.2.4 Credential Distribution

This document does not specify the manner by which resource owner or client credentials are obtained by the PHG. This is considered to be a product specific decision.

OAuth does not define how a resource owner is authenticated, it only addresses authorization. This document also leaves authentication up to the implementation.

8.2.5 Access Token Scope

This document does not place a requirement on how the scope parameter is to be used.

8.2.6 Authorization Using a JSON Web Token

8.2.6.1 Introduction

This CDG profiles RFC 7523 [OAuth JWT] to enable the use of a Jason Web Token (JWT) in an OAuth exchange with the token endpoint of an OAuth server. The use of a JWT allows a client to express an existing trust relationship via a credential based mechanism, in which there is no direct user-approval step at the authorization server. The use of a JWT provides an alternative to passwords which may simplify security credential management.

NOTE – The profiling of JWT as a client authentication mechanism, which is also defined in RFC 7523, is out of scope.

8.2.6.2 Transporting Assertions in JWTs

A PHG using a JWT to communicate assertions to a H&FS **shall** perform an HTTP POST to the token endpoint as defined in section 4 of [OAuth Assertion] as profiled below:

– The value of grant_type **shall** be "urn:ietf:params:oauth:grant-type:jwt-bearer".

– The assertion parameter **shall** contain a single JWT as specified in [OAuth JWT]

A H&FS returning an error parameter for an invalid JWT **shall** provide additional information regarding the error by using the "error_description" or "error_uri" parameters.

8.2.6.3 JWT Claims

A PHG using a JWT **shall** issue the JWT with claims as specified in clause 3 of [OAuth JWT], the PHG is required to meet the following additional requirements

– The JWT **shall** contain an "iat" (issued at) claim that identifies the time at which the claim was issued.

– The JWT **shall** contain a "jti" (JWT ID) claim that provides a unique identifier for the token.

A H&FS that advertises support for JWT in Capability Exchange **shall** conform to the requirements in [OAuth JWT].

8.3 FHIR Operations

This clause provides guidelines that profile the behaviour of the PHG and H&FS to ensure that the semantic content of an upload is consistent between all conformant implementations of PHG and H&FS.

8.3.1 Implementation Types and Interoperability

Conformant H&FSs and PHGs interoperate with each other as shown in Table 8-3.

Table 8-3 – Measurement Upload Support

	FHIR Observation Server	**FHIR Observation Reporting Server**
FHIR Observation Client	Yes [1]	No [2]
FHIR Observation Reporting Client	Yes	Yes

NOTES

[1] – Can upload measurements in an optimized manner

[2] – A given implementation of a FHIR Observation Client might send measurements as complete transaction bundles when it detects a H&FS with a FHIR Observation Reporting Server, but these Guidelines do not require that behaviour.

A FHIR Observation Client can upload to a FHIR Observation Server in an optimized fashion. However, it may not be able to upload to a FHIR Observation Reporting Server at all.

A FHIR Observation Reporting Client will be able to upload to either type of FHIR H&FS; however, it may not take full advantage of the capabilities of a FHIR Observation Server resulting in greater bandwidth consumption.

8.3.1.1 FHIR API Support for FHIR Observation Server

A FHIR Observation Server **shall** support, at minimum, the following FHIR operations as defined by [RESTful FHIR]:

– Instance Level Interactions
  - Update, including conditional update

– Type Level Interactions
  - create, including conditional create
  - search

– Whole System Interactions
  - capabilities

- transactions

Within a transaction bundle, the following interactions **shall** be supported:

– Instance Level Interactions
  - Update, including conditional update

– Type Level Interactions
  - create, including conditional create

8.3.1.2 FHIR API Support for FHIR Observation Reporting Server

A FHIR Observation Reporting Server **shall** support the FHIR *create* call (http POST) of a syntactically correct FHIR transaction bundle as specified by [RESTful FHIR] in which all references can be resolved to resources in the bundle (complete FHIR bundle).

NOTE – Within the transaction bundle additional FHIR operations are allowed (see 8.3.1.1), the H&FS **shall not** return an error if the operations specified within the bundle are one of the supported operations for a transaction bundle as identified in clause 8.3.1.1.

8.3.1.3 Obtaining FHIR Measurement Server Type

The PHG **shall** use Capability Exchange to determine the type of measurement receiver.

8.3.2 Measurement Uploads

There are three FHIR defined resources that are associated with a measurement upload, the Patient Resource, the DeviceComponent Resource, and the Observation Resource. These guidelines do not address the behaviour or semantic content of any other FHIR resource.

8.3.2.1 Patient Resource

The Patient Resource links the measurement information to the patient. The Logical ID of the Patient Resource identifies the Patient Resource, and indirectly the patient. The PHG must know the Logical ID of the Patient Resource, or must be able to provide a Patient Designator which will allow the Logical ID to be located.

8.3.2.1.1 Uploading to a FHIR Observation Server

Clauses 6.1.1 through 6.1.4 cover four anticipated work flows allowing a PHG to upload a Patient Resource to a FHIR Observation Server properly. Within these workflows there are two basic situations, one where the PHG is required to generate the Logical ID of the Patient Resource (scenario 2), and one where it is not (scenarios 1, 3 and 4). When the PHG generates the Logical ID, it must be unique across all Patient Resources within an H&FS defined scope. When the PHG does not generate the Logical ID, it must provide a Logical ID that is known to the H&FS, or use the Patient Designator information to allow the H&FS to locate or create the Logical ID of the Patient Resource.

If a PHG is provided with the Logical ID of the Patient Resource for a given patient, the FHIR Observation Server is indicating that it already has that Patient Resource. In this case the PHG **shall not** upload (FHIR create, conditional create, or update operation) a Patient Resource to the H&FS.

If the Sending PHG is not provided with the Logical ID of a Patient Resource, the PHG will have to have Patient Designator information, which is the information that will allow the PHG to provide values for the *Patient.identifier.system* and *Patient.identifier.value* in the Patient Resource.

Patient Record Logical ID Management

When a FHIR Observation Client uploads a Patient Resource using a FHIR *update* operation, the server uses the Logical ID specified by the FHIR Observation Client. When the Patient Resource upload operation is a *create* or a *conditional create*, the Logical ID is created by the H&FS.

– A FHIR Observation Client **shall not** specify the Logical ID for the Patient Resource when performing a single-resource create or a conditional create. The FHIR Observation Client **shall** provide the Patient Designator information in the Patient Resource.

– A FHIR Observation [Reporting] Client **shall** specify the Logical ID for the Patient Resource when performing a single-resource update or when specifying an update transaction for the Patient resource in a transaction Bundle.

– A FHIR Observation [Reporting] Client **shall** specify a temporary Logical ID for the Patient Resource being created or conditionally created in a Transaction Bundle if the Patient Resource is referenced by another resource in the transaction Bundle. The FHIR Observation [Reporting] Client **shall** provide the Patient Designator information in the Patient Resource.

8.3.2.1.2 Uploading to a FHIR Observation Reporting Server

When uploading to a FHIR Observation Reporting Server the measurement **shall** be uploaded as a http POST operation with the payload containing a complete FHIR transaction bundle. Within the transaction bundle the FHIR operation on the Patient Resource **shall** be either update or conditional create. The update **shall only** be used when the Logical ID of the Patient Resource is known.

8.3.2.1.3 Generation of the Patient Resource

If the PHG needs to generate the Patient Resource, then in the Patient Resource:

– The *Patient.meta.profile* **shall** be set to the "placeholder/phdPatient"

– The *Patient.identifier.system* **shall** be a URI set to the health care identification system

– The *Patient.identifier.value* **shall** be set to the patient identifier

If the Patient Resource Logical ID is to be created by the PHG the PHG **shall** specify the Logical ID as the concatenation of field values:

Patient.identifier.value-Patient.identifier.system

Where the italicized strings represent the values associated with the named fields, and the "-" is a hyphen character. If this string is longer than 64 characters, it **shall** be truncated to 64 characters by removing characters from the end of the string; the logical id is restricted to 64 characters by FHIR.

If this string contains any characters other than A-Z, a-z, 0-9, "-", or ".", they **shall** be replaced by a "." (period).

If the Patient Resource is to be uploaded using a conditional create, a logical id **shall not** be specified. The search parameter **shall** be the health care identification system and the patient identifier placed in the Patient.identifier element as specified above. The client obtains the server generated logical id in the response.

An example of a Patient Resource for an update transaction that is consistent with the patient identifier information in a PCD-01 observation upload is shown in Figure 8-5:

```
"resource":{
    "resourceType":"Patient",
    "meta":{
        "profile":"placeholder/phdPatient"
    }
    "id":"234987sisId-1.2.3.4.5.6.7.8.10",
```

```
    "identifier":[
    {
        "type":
        {
            "coding":[
            {
                "system":"http://hl7.org/fhir/v2/0203",
                "code":"MR"
            } ]
        },
        "system":"urn:oid:1.2.3.4.5.6.7.8.10",
        "value":"234987sisId"
    } ],
    "name":[
    {
        "family":["Longstrump"],
        "given":["Pippi","Ulla"]
    }
}
```

Figure 8-5 – An example of Patient Resource update transaction

8.3.2.2 DeviceComponent Resource

When a DeviceComponent Resource is uploaded as a single resource or in a bundle the PHG **shall** use the FHIR update or conditional create operations. The Logical ID of the DeviceComponent Resource **shall** be unique on the H&Fs. If an update transaction is specified, the Logical ID is set to the IEEE systemId-Transport Address as indicated in Annex A.

8.3.2.3 Observation Resource

When an Observation Resource is uploaded, either as a single resource or in a bundle, the PHG **shall** use the create or conditional create operations.

8.3.2.3.1 Duplicate Observations

Personal Health Devices (PHD) often store data locally, allowing the device to be used while not in range of a PHG. Not all devices that store local data delete the data after delivering it to the PHG. This non-removal of duplicate data can lead to duplicate data being pushed into the Continua ecosystem.

The PHG **should** attempt to filter out duplicate data when sending device measurements to a H&FS.

A PHG **should** filter out duplicate measurements from the sensor device(s) using an implementation defined filtering algorithm.

Once the received measurements have passed through the duplicate filtering process, a PHG **shall** use a conditional create to upload the measurement unless one or more of the following conditions are true, in which case a create is used:

– The measurement is received by the PHG and there is no associated timestamp for the measurement.

– The measurement received by the PHG is a live measurement. The PHG **shall** identify a live measurement based on the difference between the time of reception of the measurement, and the corrected timestamp provided by the PHD. If the corrected timestamp indicates that the measurement was taken by the PHD within an application defined expiration period, then the measurement is considered to be live. These guidelines do not proscribe a value for the

expiration period as it depends on the sensor itself, and the way the sensor is used. A suggested value for the expiration period is sixty (60) seconds.

8.3.2.4 Payload Format

A PHG **shall** use either JSON or XML when uploading a measurement. A H&FS **shall** support uploads in both JSON and XML format.

9 Normative Guidelines

The tables in this clause list the guidelines for the four Continua Certified Capability Classes associated with uploading a device observation using FHIR.

9.1 Requirements Common to both H&FS FHIR Capability Classes

This clause addresses the conformance requirements that are common to both the FHIR Observation Server and the FHIR Observation Reporting Server. The term H&FS is used to designate both the FHIR Observation Reporting Server and the FHIR Observation Server.

Table 9-1 – Requirements Common to both H&FS FHIR Capability Classes

Name	Description	Comments
FHIR-H&FS-oauth-2.0-required	A H&FS **shall** support [OAuth 2.0]	
FHIR-H&FS-oauth-grantTypes	A H&FS **shall** support the following Authorization Grant types: – Resource Owner Credential – Client Credentials	Other grant types may also be supported
FHIR-H&FS-oauth-grantTypes-interactive-grants	A H&FS **shall** conditionally support the Authorization Code and implicit grant type if the H&FS is intended to be used with platforms that support web or interactive interfaces.	Conditional support is determined by developer
FHIR-H&FS-CE-supported	A H&FS with a FHIR Observation Server or a FHIR Observation Reporting Server **shall** implement Capability Exchange as specified in [H.812.3].	A Continua H&FS simplifies provisioning of PHGs through Capability Exchange. Non Continua FHIR servers may require other means to obtain provisioning information.
FHIR-H&FS-CE-HCP-reference	The value of the <reference> child element in the <profile> element of the root.xml file, which points to the Hdata Content Profile document, **shall** point to the latest revision of this document that the implementation supports.	The resource at the provided URL is what determines the latest supported version.
FHIR-H&FS-CE-resourceType	A <resourceType> element with a <resourceTypeID> child element set to the value OAuthDescriptor **shall** be present in the root.xml file.	

Name	Description	Comments
FHIR-H&FS-CE-resourceType-reference	The <reference> element for the resourceType **shall** point to the latest revision of this document that the implementation supports.	
FHIR-H&FS-CE-resourceType-representation	The <mediaType> element in the <representation> element for the resourceType **shall** be set to application/json or application/xml.	
FHIR-H&FS-CE-resourceType-consistency-xml	If the mediaType is set to application/xml the H&FS **shall** use XML to represent Capability Exchange information as defined in [H.812.3].	
FHIR-H&FS-CE-resourceType-consistency-json	If the mediaType is set to application/json the H&FS **shall** use JSON to represent Capability Exchange information as defined in [H.812.3].	
FHIR-H&FS-CE-section-resourceTypeID	The <resourceTypeID> element in the section **shall** be set to OAuthDescriptor.	
FHIR-H&FS-CE-section-atom-feed	When the base path is concatenated as a prefix to the contents of the <path> child element in the <section> element, the H&FS **shall** return an atom feed that lists the OAuthDescriptor resources.	
FHIR-H&FS-CE-OAuthDescriptor-entries	The H&FS **shall** provide a OAuthDescriptor for each distinct FHIR uploading service it exposes.	
FHIR-H&FS-CE-section-not-empty	The atom feed returned by H&FS **shall** have at least one OAuthDescriptor listed.	
FHIR-H&FS-OAuthDescriptor	The H&FS **shall** return a OAuthDescriptor as specified in Table 8-1.	There are multiple conformance requirements in referenced table
FHIR-H&FS-RFC7523	A H&FS **may** include the "rfc7523" string in the grantTypes element of the OAuthDescriptor. If it does it **shall** conform to the requirements of IETF RFC 7523.	
FHIR-H&FS-RFC7523-invalid-jwt	A H&FS returning an error parameter for an invalid JWT **shall** provide additional information regarding the error by using the "error_description" or "error_uri" parameters.	
FHIR-H&FS-no-error-bundle	For a syntactically correct transaction bundle in which there are no external references the H&FS **shall not** return an http error due to unsupported FHIR operations.	
FHIR-H&FS-xml-and-json-support	A H&FS **shall** support uploads in both JSON and XML format.	

Name	Description	Comments
FHIR-H&FS-OPS-API-Create	A H&FS **shall** support the FHIR create API call (http POST) of a syntactically correct FHIR transaction bundle as specified by [RESTful FHIR] in which all references can be resolved to resources in the bundle (complete FHIR bundle).	

9.2 Requirements Common to both PHG FHIR Capability Classes

This clause addresses the conformance requirements that are common to both the FHIR Observation Client and the FHIR Observation Reporting Client, which are listed in Table 9-2. To simplify description within this table, the term PHG is to be understood to represent both the FHIR Observation Client and FHIR Observation Reporting Client Continua Certified Capability Classes.

Table 9-2 – Requirements Common to both PHG FHIR Capability Classes

Name	Description	Comments
FHIR-PHG-oauth-2.0-required	A PHG implementing the FHIR-Observation-Uploader **shall** support [OAuth 2.0]	
FHIR-PHG-oauth-grantTypes	A PHG **shall** support the use of one or more of the following OAuth Authorization Grant types: – Resource Owner Credentials – Client Credentials – Authorization Code – Implicit	
FHIR-PHG-conditional-rfc7523-support	A PHG **may** send a grant_type value of: "urn:ietf:params:oauth:grant-type:jwt-bearer" to an H&FS that has the 'rfc7523' value in the grantTypes element of the OAuthDescriptor. If it does, it **shall** conform to the requirements of RFC 7523.	
FHIR-PHG-use-of-rfc7523	A PHG **shall not** send a grant_type value of: "urn:ietf:params:oauth:grant-type:jwt-bearer" to an H&FS that does not list the 'rfc7523' value in the grantType element of the OAuthDescriptor	
FHIR-PHG-rfc7523-jwt-bearer	A PHG using a bearer JWT as defined in [OAuth JWT] **shall** issue an access token request as defined in section 4 of [OAuth Assertion] with the following parameters and values: – grant_type **shall** be "urn:ietf:params:oauth:grant-type:jwt-bearer". – assertion **shall** contain a single JWT	
FHIR-PHG-rfc7523-claims	A PHG using a JWT **shall** issue the JWT with claims as specified in clause 3 of [OAuth JWT].	

Name	Description	Comments
FHIR-PHG-rfc7523-claims-iat	The JWT **shall** contain an "iat" (issued at) claim that identifies the time at which the claim was issued	
FHIR-PHG-rfc7523-claims-jti	The JWT **shall** contain a "jti" (JWT ID) claim that provides a unique identifier for the token.	
FHIR-PHG-discover-server-type	A PHG **shall** have an implementation of Capability Exchange that conforms to [H.812.3] including the Atom Feed to enable access to the OAuthDescriptor	A PHG implementation must provide a way to use Capability Exchange to obtain the OAuthDescriptor of a H&FS.
FHIR-PHG-FOS-patient-resource-no-upload	If the PHG is provided the Logical ID of the Patient Resource, the PHG **shall not** upload a Patient Resource to the H&FS.	Applies when uploading to a FHIR Observation Server
FHIR-PHG-patient-resource-no-logical-id-on-creates	A PHG **shall not** specify the Logical ID for the Patient Resource when performing a single-resource create or a conditional create of the Patient Resource. The PHG **shall** provide the Patient Designator information in the Patient Resource.	
FHIR-PHG-patient-resource-logical-id-on-update	A PHG **shall** specify the Logical ID for the Patient Resource when performing a single-resource update.	
FHIR-PHG-patient-resource-temporary-logical-id	A PHG **shall** specify a temporary Logical ID for the Patient Resource being created or conditionally created in a Transaction Bundle if the Patient Resource is referenced by another resource in the transaction Bundle. The PHG **shall** provide the Patient Designator information in the Patient Resource.	
FHIR-PHG-FORS-upload-complete-bundle-using-post	When uploading to a FHIR Observation Reporting Server the measurement **shall** be uploaded as a http POST operation with the payload containing a complete FHIR transaction bundle.	
FHIR-PHG-FORS-upload-complete-bundle-ops	Within a transaction bundle the FHIR operation on the Patient Resource **shall** be either update or conditional create. The update **shall only** be used when the Logical ID of the Patient Resource is known.	
FHIR-PHG-patient-resource-gen-pid	If the PHG needs to generate the Patient Resource, then in the Patient Resource the *Patient.identifier.system* **shall** be a URI set to the health care identification system and the *Patient.identifier.value* **shall** be set to the patient identifier component of the Patient Descriptor	

Name	Description	Comments
FHIR-PHG-patient-resource-gen-pid-structure	If the Patient Resource Logical ID is to be created by the PHG the PHG **shall** specify the Logical ID as the concatenation of field values: *Patient.identifier.value-Patient.identifier.system* Where the italicized strings represent the values associated with the named fields, and the "-"is a hyphen character. If this string is longer than 64 characters, it **shall** be truncated to 64 characters by removing characters from the end of the string. If this string contains any characters other than A-Z, a-z, 0-9, "-", or ".", they **shall** be replaced by a "." (period).	
FHIR-PHG-patient-resource-upload-conditional-create	If the Patient Resource is to be uploaded using a conditional create, a logical id **shall not** be specified. The search parameter **shall** be the health care identification system and the patient identifier placed in the Patient.identifier element as specified above.	
FHIR-PHG-devicecomponent-upload	When a DeviceComponent Resource is uploaded as a single resource or in a bundle the PHG **shall** use the FHIR update or conditional create operations.	
FHIR-PHG-devicecomponent-id	The Logical ID of the DeviceComponent Resource **shall** be unique on the H&Fs and **should** contain the IEEE systemId of the device this component is to represent. If the systemId is not available, then the Logical ID **should** be set to a UUID conformant to RFC 4122.	
FHIR-PHG-obsres-upload	When an Observation Resource is uploaded, either as a single resource or in a bundle, the PHG **shall** use the create or conditional create operations.	
FHIR-PHG-dup-filter	The PHG **should** attempt to filter out duplicate data when sending measurements to a H&FS using an implementation defined filtering algorithm.	

Name	Description	Comments
FHIR-PHG-dup-use-of-conditional-create	A PHG **shall** use a conditional create to upload measurement unless one or more of the following conditions are true, in which case a create **shall** be used (1) The measurement is received by the PHG and there is no associated timestamp for the measurement. (2) The measurement received by the PHG is a live measurement.	See clause 8.3.2.3.1 for additional details.
FHIR-PHG-use-of-xml-and-json	A PHG **shall** format measurement payloads in JSON or XML.	
PHG Requirements from Annex A		
FHIR-PHG-Nomenclature-Encoding	PHGs **shall** map Nomenclature codes to CodeableConcept data types as specified in A.1.5.1.	Nomenclature codes are frequently mapped to CodeableConcept data types.
FHIR-PHG-ASN.1-Encoding	PHGs **shall** map ASN.1 BITs fields as specified in A.1.5.2.	This mapping allows ASN.1 BITs field to be mapped to codes which can then be mapped to CodeableConcept data types in FHIR. BITs fields otherwise have no corresponding FHIR data type.
FHIR-PHG-MderFloat-Encoding	PHGs **shall** encode MderFloat representations as specified in A.1.5.3.	This section describes the means of capturing the precision indicated in the MderFloat encoding.
FHIR-PHG-PHG-Properties-Mapping	PHGs **shall** map their static properties into a DeviceComponent resource as specified in A.3.	
FHIR-PHG-Sensor-Properties-Mapping	PHGs **shall** map the static sensor properties expressed by a PHD's MDS object attributes into one or more DeviceComponent resources as specified in A.4.	In most cases, only a single DeviceComponent resource is needed, but if multiple specializations are supported, more than one DeviceComponent is needed.

Name	Description	Comments
FHIR-PHG-Coincident-Time-Stamp-Generation	PHGs **shall** generate Coincident Time Stamp Observation resources as specified in A.5.	A Coincident Time Stamp Observation is not always necessary. Measurements reporting no time stamp are an example of where a Coincident Time Stamp is not needed.
FHIR-PHG-Measurement-Mapping	PHGs **shall** map measurements to Observation resources as specified in A.6.	Mapping IEEE 11073-20601 measurements to Observation resources involves attention to many details. The measurement mapping is based upon the object and attribute type and not the value. This approach allows a PHG to handle both current and future IEEE 11073 measurements without software updates as long as the base classes in IEEE 11073 stay compatible with those in 11073-20601 v3.
FHIR-PHG-Sync-qualified-time	A PHG **shall** be capable of synchronizing to qualified time.	The PHG is to map the sensor time line to qualified time. Qualified time is any time synchronized to UTC with or without knowledge of local time. See the [H.812.1] section on timestamping for additional details.

9.3 Requirements Specific to the FHIR Observation Server

This clause addresses the conformance requirements that are specific to the FHIR Observation Server. Additional requirements apply to this certified capability class, see clause 9.1.

Table 9-3 – Requirements Specific to the FHIR Observation Server

Name	Description	Comments
FOS-CE-profile-id	The value of the child element <id> in the <profile> element **shall** be set to "FHIR-Observation-Server-4C".	
FOS-CE-section-profileID	The value of the child element <profileID> in the <section> element **shall** be set to "FHIR-Observation-Server-4C".	
FOS-supported-FHIR-operations	A FHIR Observation Server **shall** support, at minimum, the following FHIR operations as defined by [RESTful FHIR]: – Instance Level Interactions o Update – Type Level Interactions o create, including conditional create o search – Whole System Interactions o Capabilities o transactions	

9.4 Requirements Specific to the FHIR Observation Reporting Server

This clause addresses the conformance requirements that are specific to the FHIR Observation Reporting Server. Additional requirements apply to this certified capability class, see clause 9.1.

Table 9-4 – Requirements Specific to the FHIR Observation Reporting Server

Name	Description	Comments
FORS-CE-profile-id	The value of the child element <id> in the <profile> element **shall** be set to "FHIR-Observation-Reporting-Server-4C"	
FORS-CE-section-profileID	The value of the child element <profileID> in the <section> element **shall** be set to "FHIR-Observation-Reporting-Server-4C".	

9.5 Requirements Specific to the FHIR Observation Client

This clause addresses the conformance requirements that are specific to the FHIR Observation Client. This CDG does not mandate how a FHIR Observation Client optimizes its measurement uploads with a FHIR Observation Server, however, if the PHG can only send measurements as complete transaction bundles it is considered a FHIR Observation Reporting Client. Additional requirements apply to this certified capability class, see clause 9.2.

Table 9-5 – Requirements Specific to the FHIR Observation Client

Name	Description	Comments
FHIR-FOC-FORS-must-use-bundle	When a H&FS advertises a FHIR Observation Reporting Server the PHG **shall** only send measurements that are fully contained in a complete transaction bundle.	The FOC may upload measurements to a FORS but is not required to do so.
FHIR-FOC-FOS-optimize-upload	A FHIR Observation Client **should** optimizes its measurement uploads with a FHIR Observation Server by using single resource interactions that avoid repeating data already uploaded	

9.6 Requirements Specific to the FHIR Observation Reporting Client

This clause addresses the conformance requirements that are specific to the FHIR Observation Reporting Client. Additional requirements apply to this certified capability class, see clause 9.2.

Table 9-6 – Requirements Specific to the FHIR Observation Reporting Client

Name	Description	Comments
FHIR-FORC-FOS-support	A PHG implementing only a FHIR Observation Reporting Client **shall** be able to upload measurements to a H&FS advertising a FHIR Observation Server Continua Certified Capability Class.	A conformant FOS can handle the FHIR transaction bundle of a FORC. The FORC can therefore deliver the measurement to the FOS successfully
FHIR-FORC-supports-FORS	A PHG claiming support for a FHIR Observation Reporting Client **shall** be able to upload a measurement to a H&FS advertising a FHIR Observation Reporting Server Continua Certified Capability Class.	This is the interoperability baseline

Annex A

ISO/IEEE 11073 to FHIR Resource Mapping

(This annex forms an integral part of this document.)

A.1 Mapping from ISO/IEEE 11073-20601 to FHIR resources

This Annex requires an understanding of both ISO/IEEE 11073-20601 [20601] and the HL7 FHIR specifications. Appendix I of this document provides an informative background for some of the relevant aspects of the HL7 and IEEE specifications as they relate to this Annex. Readers less familiar with FHIR and ISO/IEEE 11073-20601 concepts may find it advantageous to read Appendix I.2 before reading this Annex.

A.1.1 General Notes on Mapping Tables

The mapping tables do not capture all aspects of the translation from IEEE 11073 to FHIR due to difficulties associated with placing these aspects in tabular format. When reading the tables, the following points should be taken into account:

– Every resource referred to in this mapping shall have a *.meta.profile element set to the particular structure definition (profile) the resource is following.

– Most entries containing a CodeableConcept data type that require an IEEE 11073-10101 code allow alternative coding systems. In these cases, the IEEE 11073-10101 coding entry shall occur first. When LOINC is required, as it is by FHIR for vital signs, it shall occur second.

– When component entries are required in addition to component entries from compound or ASN1 BITs enumeration measurements, the additional component entries come after the compound or ASN1 BITs entries.

A.1.1.1 Placeholders

This specification requires the use of a three value sets and nine custom structure definitions (profiles). At the time of writing this document URIs for these value sets used in the *.coding.system element and profiles used in the *.meta.profile element have not been finalized. In the mapping below placeholders are used instead of the actual URIs. The word 'placeholder' occurs in the URI to remind the implementer that these URIs have not yet been determined. The affected URIs are as follows:

Table A-1 – Placeholders

Parameter	Placeholder
ASN1 Value Set	placeholder/fhir/IEEE.ASN1
PCHA PHD certification codes	placeholder/fhir/reg-cert-codes
PCHA H&FS certification codes	placeholder/fhir/reg-cert-wan-codes
PhdParentDeviceComponent profile	placeholder/phdParentDeviceComponent
PhdChildDeviceComponent profile	placeholder/phdChildDeviceComponent
PhgDeviceComponent profile	placeholder/phgDeviceComponent
PhdNumericObservation profile	placeholder/phdNumericObservation
PhdCompoundNumericObservation profile	placeholder/phdCompoundNumericObservation
PhdRtsaObservation profile	placeholder/phdRtsaObservation
PhdCodedEnumerationObservation profile	placeholder/phdCodedEnumerationObservation

PhdBitsEnumerationObservation profile	placeholder/phdBitsEnumerationObservation
PhdStringEnumerationObservation profile	Placeholder/phdStringEnumerationObservation
PhdCoincidentTimeStampObservation profile	placeholder/phdCoincidentTimeStampObservation

A.1.2 Terminologies and Conventions

The following notations, conventions and terms are used in this Annex:

A.1.2.1 FHIR Operations

This specification will use the following RESTFul FHIR operation terminologies:

– **update**: the uploading of a resource using an HTTP PUT where the client specifies the logical id of the resource. The resource may or may not exist on the FHIR server. If it does not exist, the resource is created. If it does exist, the resource is replaced by the uploaded resource and the metadata version number is updated by the FHIR server.

– **create:** the uploading of a resource using an HTTP POST where the FHIR server generates the final logical id of the resource. The resource is assumed not to exist and a new resource will be created. Even if the resource does exist, this operation will cause the generation of a new resource with its own metadata history.

– **conditional create:** the uploading of a resource using an HTTP POST with a special FHIR-defined header element determining a search parameter on a certain element in the resource. If one such resource is found to exist, the resource is left alone and a 200 response code is returned. If no such resource is found to exist the resource is created and a 201 response code is returned. The FHIR server generates the final logical id of the resource. If more than one resource is found an error code is returned.

These operations may be applied to individual resources if the H&FS supports the CDG FHIR Observation Server or specified individually for each resource within a transaction Bundle. A 'complete' transaction Bundle is supported by both the CDG FHIR Observation Server and CDG FHIR Observation Reporting Server classes.

A.1.2.2 ASN.1

Abstract Syntax Notation 1 (ASN.1) is used extensively in [ISO/IEEE 11073-20601]. This annex follows the conventions of [ISO/IEEE 11073-20601] in using both the IEEE 11073-20601 ASN.1 names and ASN.1 notation to identify IEEE attributes. In particular:

– An IEEE attribute is specified by its ASN.1 name (e.g. Production-Specification or *Basic-Nu-Observed-Value*)

– If a named element in a complex structure is a primitive, it can be assigned a value, the named element is italicized.

– If the attribute itself is a primitive, the attribute name will be in italics, for example, *Basic-Nu-Observed-Value*

– Dots are used to separate levels in a complex structure

– *Partition* is used to represent the most significant 16 bits of a 32-bit nomenclature code

– *Term code* is used to represent the least significant 16 bits of a 32-bit nomenclature code

– Some ASN.1 values are determined by another field in the ASN.1 struct, for example in the Production-Specification. If a specific value in such as case is indicated, that specific value is attached by an underscore. For example, the production specification values are indicated by Production-Specification.*prod-spec.* However, that value takes on one of several possible

meanings depending upon the value of another element. If the value being indicated is the serial number, it is indicated by Production-Specification.*prod-spec_serialnumber*.

In actual practice the only way one knows that the Production-Specification.*prod-spec* is a serial number is to examine the Production-Specification.*spec-type* element value. A *spec-type* value of 1indicates that the *proc-spec* is a serial number.

– A FHIR resource is indicated by its resource name, for example the DeviceComponent resource.

– An element in the resource is indicated by following its hierarchal structure as defined on the FHIR website, http://hl7.org/fhir/resourcelist.html, for each resource. For readability, a dot is used to separate the element names of this structure. If the element is a primitive (takes a value) and is not itself a structure, it is indicated in italics. For example, the 'code' value of a DeviceComponent resource 'type' field is indicated by DeviceComponent.type.coding.*code*.

– The direct mapping of an attribute value to a FHIR resource element is indicated by using an equal sign. Thus Observation.valueQuantity.*value = Basic-Nu-Observed-Value*.

Handling Bluetooth low energy (LE) attributes is done by semantically mapping its characteristic values to [ISO/IEEE 11073-20601] attributes and using the guidelines provided in this document to generate the FHIR payload. The Transcoding White Paper [Bluetooth PHDT] provides the characteristic-to-IEEE attribute mapping.

A.1.3 Protocol dependent Information

[ISO/IEEE 11073-20601] defines concepts whose sole purpose is to aid in the exchange of APDUs such that the data in the APDUs can be reconstructed into measurements. The PM Store, PM Segment, Scanner, Attribute Value Map, scan event report formats (fixed, variable, group), object handles and config ids are all examples of these concepts. These items are not reported in the FHIR resources. One will note that the Bluetooth Low Energy transcoding white paper [Bluetooth PHDT] is also careful to only map Bluetooth LE features to [ISO/IEEE 11073-20601] features that are important for sensor properties and measurements. There is no mapping, for example, of the Bluetooth LE RACP to PM Stores and PM Segments or Bluetooth LE characteristic descriptors to [ISO/IEEE 11073-20601] attributes since these items are all aids in the protocol transfer of information and not the information itself.

A.1.4 CDG Nomenclature

Additional nomenclature codes not defined in [ISO/IEEE 11073-20601] are defined in the CDG to describe sub-structures of attributes that have no nomenclature codes. For example, the serial number, firmware revision, hardware revision, software revision, etc. are all sub-elements of the Production-Specification attribute. Codes are defined in the CDG for each of these elements to map them to PCD-01 messages. These codes are needed for the CodeableConcept entries in the FHIR resources when mapping these complex attributes. These CDG codes will NOT be seen on the wire in the exchange between the sensor and the PHG and therefore must be added by the PHG during the mapping. The following Table lists the codes for the sub-structures of the Production-Specification and Reg-Cert-Data-List attributes and the PHG indicator.

Table A-2 – Additional Nomenclature Codes

Partition	Nomenclature Code's Common Name	Code
MDC_PART_INFRA	MDC_MOC_VMS_MDS_AHD	7693
MDC_PART_INFRA	MDC_REG_CERT_DATA_CONTINUA_VERSION	8064
MDC_PART_INFRA	MDC_REG_CERT_DATA_CONTINUA_CERT_DEV_LIST	8065
MDC_PART_INFRA	MDC_REG_CERT_DATA_CONTINUA_REG_STATUS	8066
MDC_PART_INFRA	MDC_REG_CERT_DATA_CONTINUA_AHD_CERT_LIST	8067
MDC_PART_INFRA	MDC_ID_MODEL_NUMBER	7681
MDC_PART_INFRA	MDC_ID_MODEL_MANUFACTURER	7682
MDC_PART_INFRA	MDC_ID_PROD_SPEC_UNSPECIFIED	7683
MDC_PART_INFRA	MDC_ID_PROD_SPEC_SERIAL	7684
MDC_PART_INFRA	MDC_ID_PROD_SPEC_PART	7685
MDC_PART_INFRA	MDC_ID_PROD_SPEC_HW	7686
MDC_PART_INFRA	MDC_ID_PROD_SPEC_SW	7687
MDC_PART_INFRA	MDC_ID_PROD_SPEC_FW	7688
MDC_PART_INFRA	MDC_ID_PROD_SPEC_PROTOCOL	7689
MDC_PART_INFRA	MDC_ID_PROD_SPEC_GMDN	7690

A.1.5 ISO/IEEE 11073-20601 General Object/Attribute Mapping

This Annex provides detailed guidelines for the mapping of [ISO/IEEE 11073-20601] objects and attributes to FHIR resources. The guidelines in this annex apply to any sensor whose information and/or observations are mapped to [ISO/IEEE 11073-20601] objects and attributes even if they are not [ISO/IEEE 11073-20601] sensors. An example would be Bluetooth Low Energy sensors whose transactions have been mapped to the necessary [ISO/IEEE 11073-20601] objects and attributes in the Transcoding White Paper [Bluetooth PHDT].

A.1.5.1 Nomenclature Code Mapping

The mapping of [ISO/IEEE 11073-20601] to FHIR is primarily through the IEEE nomenclature codes. The codes describe what the measurement is, the units, and can be the measurement itself. In FHIR these codes are mapped to the appropriate CodeableConcept elements. This encoding occurs so frequently it is worth indicating in detail how the encoding is done.

CodeableConcept Encoding for Nomenclature codes:

A nomenclature code is always associated with a partition. In attributes that only report the term code portion of the nomenclature code, the partition is implicit; for example, the *Unit-Code* attribute value always comes from partition 4 (MDC_PART_DIM). In other cases, the partition is obtained by default from the Type attribute value's partition. Additional attributes may further change this default. In the end nomenclature code information is mapped to a CodeableConcept element that is defined as

– CodeableConcept

coding 0..*

- ***system*** This is always **urn:iso:std:iso:11073:10101** for MDC
- *version* if present 1
- ***code*** partition * 2^{16} + term code
- *display* ref_id (suggested) + additional application-selected text.
- *userSelected* if present always false.

text if present, application selected.

FHIR is flexible. None of the above elements are required by the FHIR specification. However, the bold elements are required in this CDG mapping specification. It should also be pointed out that the 0..* cardinality of the coding element does not refer to additional coding concepts but *alternative coding systems* for the same concept, for example LOINC or SNOMED CT. For CDG mapping, only the entrants for the IEEE 11073-20601 nomenclature coding system are required (referred to as MDC codes in this document). The reason for this requirement is that these nomenclature codes are provided by the sensor device and there is no introduced ambiguity due to translation. Providing *additional* coding system descriptions are up to the application; however, if an alternative coding system is represented, it must come from a recognized standardized mapping specification such that all translations to alternative coding systems are done consistently. The additional coding systems are additional; they do not replace the MDC entrants.

FHIR requires that all vital signs measurements record the LOINC code. If this requirement is not rescinded, then both MDC and LOINC codes will be needed in those cases.

Example: A weigh scale contains a body mass object. The TYPE attribute value for this object gives the partition value, which is 2, and the nomenclature code, which is 57664.The reference id for code 57664 in partition 2 (SCADA) is MDC_MASS_BODY_ACTUAL. The reference Id as a string must be looked up; it is not present on the wire in an [ISO/IEEE 11073-20601] transaction. Thus, the encoded value for the code element is $2*2^{16} + 57664 = 188736$. Note that in HEX the value is 0x0002E140. The partition is the most significant two bytes and the nomenclature code the least significant two bytes. The final CodeableConcept entrants would be as follows:

```
      "code":{
                "coding":[
                    {
                        "system":"urn:std:iso:11073:10101",
                        "code":"188736",
                        "display":"MDC_MASS_BODY_ACTUAL Body mass"
                    },
                ]
            },
```

Without the optional display name, the entrant would appear as follows:

```
      "code":{
                "coding":[
                    {
                        "system":"urn:std:iso:11073:10101",
                        "code":"188736",
                    },
                ]
            },
```

which is not as easy for a human reader to interpret.

Given that body mass is a vital sign, FHIR would require that the LOINC code also be present:

```
      "code":{
                "coding":[
                    {
                        "system":"urn:std:iso:11073:10101",
                        "code":"188736",
                        "display":"MDC_MASS_BODY_ACTUAL Body mass"
                    },
                    {
                        "system":"http://loinc.org",
                        "code":"8310-5",
                    },
                ]
            },
```

A.1.5.2 ASN.1 BITS: ASN1 Vocabulary set

Though not as ubiquitous as nomenclature codes, the mapping of an [ISO/IEEE 11073-20601] ASN.1-BITs field to FHIR resources requires the use of a translation. There is currently no means to represent ASN.1-BITs fields in either FHIR or C-CDA templates. The translation is to create a new HL7 'vocabulary' and follow all the logistic procedures of defining that vocabulary and registering it so it can be referenced and used. The vocabulary defines a set of codes for each ASN.1 bit setting in use. New codes will be added as new specializations are developed and new ASN.1 BITs fields are defined.

The vocabulary set will be a set of codes constructed as follows for Enumeration metric ASN.1 BITs measurements:

– (type-attribute value).mder-bit-position

For all other cases (Power status, MdsTimeInfo capabilities, Regulation status, etc.):

– attribute-id.*value*/attribute-component-id.*value*.mder-bit-position

The Type attribute value, which is a 32-bit integer computed from the Type.*partition* and Type.*term-code* entries), is used when the ASN.1 BITs field is a measurement; one of the two Enumeration observational BITs fields attributes. Every metric object has a Type attribute that defines what the measurement is.

The attribute-id or attribute-component-id is used when the ASN.1 BITs field being mapped is not a measurement reported through the Enumeration metric object; for example, the Power status or time capabilities from the Mds-Time-Info attribute. The attribute-id-component is used when only a sub-structure of the attribute contains the BITs field. Attribute-component-ids are not sent as part of the IEEE 11073-20601 exchange protocol and thus need to be defined as required. Attribute-component-ids are assigned an IEEE 11073-10101 nomenclature code. These assignments were made to allow one to report static device properties in PCD-01 V2 messaging. An example of an attribute that is assigned several attribute-component-ids is the Mds-Time-Info attribute. A component id is assigned for the capabilities, time synchronization, synchronization accuracy, and the resolutions of the possible time clocks. These codes will now be used in FHIR for the same reason.

The bit-position in both cases is the Mder bit position of the bit where bit 0 in Mder-encoding is the high-order bit of the actual integer. The bit-position will range from 0 to 15 for 16-bit BITs and from 0 to 31 for 32-bit BITs.

This vocabulary model is chosen since it uniquely defines the BIT concept *and it can be created from protocol for measurements* without any additional support information. It is also extensible in that a PHG will be able to generate the proper vocabulary code for future ASN.1 BITs measurements. Since the vocabulary set defines a code, the codes can be mapped to FHIR CodeableConcept and related elements. The PHG **shall** map ASN.1 BITs field using this encoding. In the display element of the CodeableConcept the implementation **should** use the ASN.1 names defined in the specifications if known. Providing the ASN.1 name indicates that the PHG 'knows' this specialization as the ASN.1 names are not sent by the device on the wire.

To describe whether the bit was set or cleared or unknown, the HL7 FHIR Boolean value set, http://hl7.org/fhir/v2/0136, **shall** be used in the valueCodeableConcept or related element. The code is a simple 'y' for set and 'n' for not set. If the bit is masked because the sensor does not support the setting, the value element **shall not** be sent and the dataAbsentReason **shall be** set to 'unknown'.

Set bits **shall** be sent. Cleared bits do not need to be sent. In most cases, the specializations should have specified the ASN.1 BIT setting such that the cleared bit is the default case that would be assumed if no BITs measurement were sent. Unfortunately, there are a few cases where that is not true. The Independent Living specialization has unfortunately used a BITs measurement to describe a true-false case where each case is important; for example, door opened or door closed. OIDs would have been more appropriate in that case.

To handle cases where multiple bits may be set, ASN.1 BITs measurements settings are mapped to Observation.component elements where there will be one component for each bit setting reported.

Example:

A pulse oximeter sends a 'Device and sensor annunciation status' measurement. The Type attribute value for this measurement is MDC_PULS_OXIM_DEV_STATUS which has a term code 19532 in the partition SCADA (2). The 32-bit code value is then $2 * 2^{16} + 19532$ or 150604. The defined ASN.1 bits for this measurement in the pulse oximeter specialization are:

Table A-3 – Description of ASN.1 items

ASN.1 Item	Description
sensor-disconnected	Agent reports that the sensor is disconnected from the instrument.
sensor-malfunction	Agent reports that the sensor is malfunctioning or faulting.
sensor-displaced	Agent reports that the sensor is not properly attached or has been dislodged, and accurate measurement is, therefore, prevented.
sensor-unsupported	An unsupported sensor is connected to the agent.
sensor-off	Agent reports that sensor is not connected to the user.
sensor-interference	Agent reports that there is interference due to ambient light or electrical phenomena.
signal-searching	Signal analysis is currently in progress prior to measurement availability.
signal-pulse-questionable	Agent determines that a questionable pulse is detected.
signal-non-pulsatile	Agent detects a nonpulsatile signal.
signal-erratic	Agent reports that the signal is erratic or is not plausible.
signal-low-perfusion	Agent reports a consistently low perfusion condition exists.
signal-poor	Agent reports a poor signal exists, possibly affecting accuracy.
signal-inadequate	Agent reports that the incoming signal cannot be analysed or is inadequate for producing a meaningful result.
signal-processing-irregularity	Agent has determined that some irregularity has been detected while processing the signal.
device-equipment-malfunction	A general device fault has occurred in the agent.
device-extended-update	An extended display update is currently active.

The vocabulary set values and ASN.1 code names would then be

Table A-4 –Code to ANS.1 name

Code	ASN.1 name
150604.0	sensor-disconnected
150604.1	sensor-malfunction
150604.2	sensor-displaced

150604.3	sensor-unsupported
150604.4	sensor-off
150604.5	sensor-interference
150604.6	signal-searching
150604.7	signal-pulse-questionable
150604.8	signal-non-pulsatile
150604.9	signal-erratic
150604.10	signal-low-perfusion
150604.11	signal-poor
150604.12	signal-inadequate
150604.13	signal-processing-irregularity
150604.14	device-equipment-malfunction
150604.15	device-extended-update

A received pulsatile quality of 0x0040 (bit 7) would be mapped in the code and value elements of the component element as follows:

```
"code":{
    "coding":[
        {
            "system":"placeholder/fhir/IEEE.ASN1*",
            "code":"150604.7",
            "display":"MDC_PULS_OXIM_DEV_STATUS.signal-pulse-questionable
                 (sensor determines that a questionable pulse is detected.)"
        },
    ]
},
"valueCodeableConcept":{
    "coding":[
        {
            "system":"http://hl7.org/fhir/v2/0136",
            "code":"y",
            "display":"questionable pulse is detected"
        },
    ]
}
```

A received pulsatile quality of 0x0140 (bit 7 and bit 9) would be mapped in an Observation resource as follows (note that the individual bit settings are reported in components).

```
{
        "resource":{
            "resourceType":"Observation",
      "meta":{
         "profile":"placeholder/phdBitsEnumerationObservation"
                              }
            "id":"OBR_1:1.0.0.15",
            "identifier":[
                {
                   "value":"150604-01400-20120628145624.000-04:00",
                },
            ],
            "status":"final",
            "code":{
                "coding":[
                    {
                        "system":"urn:std:iso:11073:10101",
```

```
                        "code":"150604",
                        "display":"MDC_PULS_OXIM_DEV_STATUS"
                    },
                ]
            },
            "subject":{
                "reference":"Patient/28da0026bc42484-1.2.3.5.6.77"
            },
            "effectiveDateTime":"2012-06-28T14:56:24.000-04:00",
            "device":{
                "reference":"Device/0022D6014AFBD418-0355E37B11AD"
            },
            "related":[
                {
                    "target":{
                        "reference":"Observation/OBR_1:1.0.0.13"
                    }
                }
            ],
            "component":[
                {
                    "code":{
                        "coding":[
                            {
                                "system":"placeholder/fhir/IEEE.ASN1*",
                                "code":"150604.7",
                                "display":"MDC_PULS_OXIM_DEV_STATUS.signal-
                                                pulse-questionable"
                            },
                        ],
                    },
                    "valueCodeableConcept":{
                        "coding":[
                            {
                                "system":"http://hl7.org/fhir/v2/0136",
                                "code":"y",
                                "display":"questionable pulse is detected"
                            },
],
                    },
                },
                {
                    "code":{
                        "coding":[
                            {
                                "system":"placeholder/fhir/IEEE.ASN1*",
                                "code":"150604.9",
                                "display":"MDC_PULS_OXIM_DEV_STATUS.signal-
                                                erratic"
                            },
                        ],
                    },
                    "valueCodeableConcept":{
                        "coding":[
                            {
                                "system":"http://hl7.org/fhir/v2/0136",
                                "code":"y",
                                "display":"signal erratic"
                            },
],
                    },
                },
            ],
        },
    },
```

A.1.5.3 FLOAT and SFLOAT Mapping

[ISO/IEEE 11073-20601] encodes many of its attribute values as 32-bit FLOAT or 16-bit SFLOATs. The decoding is straightforward. The 32-bit float contains an 8-bit exponent and a 24-bit mantissa, which can be independently treated as, signed integers for their respective bitness. Thus, an 8-bit exponent with value 0x02 is the integer 2 while 0xFE is -2. However, what is often missed in the translation is the additional semantic meaning of the exponent; it defines not only the 'power' but also the *precision*. The mantissa has the number of significant figures. The exponent gives the location of the decimal point. Whenever the exponent is positive, the reported value is an integer; it has no fractional component. Thus, if the mantissa is 2 and the exponent 3, the value reported is 2000 (2×10^3) with one significant figure. If the mantissa is 20 and the exponent 3, the value is reported is 20000 (20×10^3) with two significant figures. On the other hand, if the exponent is negative, it indicates the number being reported has fractional decimal places of precision. Thus, if the mantissa is 2 and the exponent -3, the value reported is 0.002 (2×10^{-3}). If the mantissa is 2000 and the exponent -3, the value reported is 2.000 and not 2 or 2.0 or 2.00. The '-3' indicates three decimal places of fractional precision.

– The FHIR mapping of quantities **shall** encode FLOAT and SFLOATs per the indicated decimal precision.

A.1.6 FHIR logical ids and identifiers

The logical id is used to reference one resource from another. It is unique on a server and only means something to that server. When the resource upload operation is an 'update', the server uses the logical id specified by the uploader. When the resource upload operation is a 'create', the logical id is created by the server. In a single-resource create upload, the logical id shall not be specified. In a transaction Bundle, the logical id may need to be specified even if the individual Bundle.entry.request method is a create since other resources in the Bundle may need to reference that resource. In this case the 'temporary id' is used and the server changes the logical id as it sees fit, but when doing so it will consistently update all the references to that resource in the Bundle to the server-specified logical id.

The logical id is limited to 64 characters and can only contain certain characters.

The identifier element is a means of uniquely identifying the resource content and it does ***not*** change from server to server. This specification makes extensive use of the identifier element and specifies its content. An important use of the identifier element in this specification is to avoid resource duplication.

A.1.7 Profiles: Customized Structure Definitions

Customized Structure Definitions were formally called 'profiles'. There appears to be a move in FHIR to move away from the overloaded 'profile' definition.

By specifying this mapping, the CDG defined profiles. To formally comply with FHIR these profiles need to be defined in a specified manner and resources conforming to these profiles need to indicate so in the resource meta.*profile* element.

The following profiles are defined for this mapping:

– PhdParentDeviceComponent

  This profile is used for mapping the sensor device data

– PhdChildDeviceComponent

  This profile is used when the sensor device supports multiple specializations and/or IEEE 11073-20601 sub-profiles (not to be confused with FHIR profiles)

– PhgDeviceComponent

This profile is used for the mapping of the PHG data.

– PhdNumericObservation

  This profile is used when the measurement is a simple numeric metric

– PhdCompoundNumericObservation

  This profile is used when the measurement is a compound numeric metric

– PhdRtsaObservation

  This profile is used when the measurement is an RTSA metric

– PhdCodedEnumerationObservation

  This profile is used when the measurement is an MDC coded (a simple OID) enumeration metric

– PhdBitsEnumerationObservation

  This profile is used when the measurement is an ASN.1 BITs enumeration metric

– PhdCoincidentTimeStampObservation

  This profile is used when describing the coincident time stamp information

Note that the full URLs for these profiles have not yet been defined. When they are, these URLs are entered in the resource meta.profile element.

A.2 CDG FHIR Data Model

This mapping assumes that the DeviceComponent has been updated to include a [0..] identifier cardinality and a DeviceComponent.property element. It also assumes that the Observation.device reference has been updated to include the DeviceComponent.*

A.2.1 FHIR Resources

The CDG mapping of PHD data requires the use of the Patient, DeviceComponent, and Observation resources. All required entries in the DeviceComponent and Observation resources are populated by data obtained directly or indirectly through protocol. The mapping application will need to supply the information in the Patient resource. Use of additional FHIR resources is up to the application.

A.2.1.1 PHG Resources

A DeviceComponent resource is used to map the PHG properties. The PHG in this case refers to the endpoint that is responsible for handling the transactions over the Continua services interface.

A.2.1.2 PHD Resources

In PCD-01, OBX segments are used to report both the static and dynamic properties of the sensor endpoint. In FHIR, the static properties are reported in the DeviceComponent resource whereas the dynamic properties such as the battery level and power status are reported using Observation resources.

A.2.1.3 Coincident Time Stamp

The PHG is responsible for mapping the sensor measurement time stamps to UTC plus offset to local time. The sensor is responsible for providing a contiguous unbroken time line and if it cannot, it indicates a Date-Time-Adjustment or time fault.

The FHIR equivalent of the PCD-01 coincident time stamp is reported in an Observation resource. This critical resource reports a 'measurement' of the sensor's current time taken by the PHG. Every measurement Observation resource will point to a coincident time stamp Observation if the

measurement reports a timestamp. If the PHD does not report a timestamp with the measurement, the PHG uses the time of reception as the time of the measurement and no Coincident Time Stamp Observation exists for that measurement.

The coincident time stamp indicates

– if the PHG has corrected the measurement time stamps of the sensor and if so by how much

– the mapping of sensor relative times to wall clock times

– if the PHG has created time stamps for the measurements using its time of reception (there will be no coincident time stamp for that measurement)

– if the measurement time stamps are unreliable due to a sensor time fault

– if the sensor time stamps have been used unmodified because the sensor has superior time synchronization than the PHG.

The coincident time stamp semantics are given in H.812.1 and discussed in Appendix I of this document.

A.2.1.4 Metric Measurement Resources

All metric measurements are reported using Observation resources.

A.2.1.5 Mapping Tables

To make the mapping simpler to read, the mapping is placed in tables as follows:

– The '**Resource**' column gives the FHIR hierarchy for the resource element being mapped. Only primitives can take a value, so the row showing the data type will have no value or requirement.

– The '**Value**' column gives the algorithm one uses to populate the 'Resource' element.

– To the far right of the algorithm entry is a column that gives the requirement as one of **R**, **S**, **O**, and **Z**.

R indicates the entry is required

S indicated the entry is strongly suggested

O indicates the entry is optional but if entered it **shall** be as entered in the 'Value' column

Z indicates that the entry is required but the specific situation indicated in the 'Value' column is not. The use of **Z** occurs only in a single table describing the mapping for RTSA data.

The Table structure is illustrated below:

Table A-5 – Table Structure

Resource	Value	
FHIR element being populated	Algorithm to generate the primitive element value from the IEEE 11073-20601 attributes.	**R**,**S**,**O**, or **Z**

A.3 PHG Properties Encoding Guidelines

For the purposes of mapping, the PHG is treated *as if it had* an MDS object with MDS attributes. This treatment does NOT mean the PHG must support an MDS object. The system id, serial number, model name, manufacturer number, current time, etc. are all treated as if they existed in the analogous attributes they would be present in on a sensor device. The PHG **shall** map the following information:

– The information that would be in a Reg-Cert-Data-List attribute

– The accuracy, if known, and time-synchronization information that would be in a Mds-Time-Info attribute

All other PHG MDS attributes such as the production specification, battery level, etc. are not required to be mapped.

These hypothetical attributes are only for the convenience of describing the encoding, and have no meaning outside of that purpose. There is no requirement that these attributes or the MDS object exist on the PHG.

These properties are mapped to a DeviceComponent resource which is designated as the PHG-DeviceComponent resource to distinguish it from DeviceComponent resources containing the sensor properties.

A.3.1 FHIR-Specific Requirements

FHIR requires that certain elements be present that are not related to information content.

A.3.1.1 Resource Name

All FHIR resources have a *resourceType* element. For the PHG-DeviceComponent, it is "DeviceComponent".

Example:

```
"resourceType":"DeviceComponent"
```

A.3.1.2 PHG-DeviceComponent Resource logical id

All FHIR resources have a logical *id* element which is an alpha-numeric string. The logical id is used to identify the resource in searches and when referenced by other resources. There are no specific rules for the creation of this value. A logical id **shall** be specified by the client when the client uses the FHIR 'update interaction' using the following string:

"[system Id as 16-digit HEX string].[transport address]"

where the transport address is

– a 12-digit HEX string for Bluetooth without the 0x prefix

– a 16-digit HEX string for ZigBee without the 0x prefix,

– a 4-digit VID HEX string followed by a 4-digit PID HEX string for USB separated by a dot without the 0x prefix.

If the transport address is not known, it shall be set to all 0's.

Update example:

```
"id":"7EEDABEE34ADBEEF.00f1FE09b155"  (Bluetooth)
```

A.3.1.3 Profile

The meta.profile entry **shall** contain "placeholder/phgDeviceComponent".

NOTE – The actual profile URL is not yet specified.

A.3.2 PHG Data Mapping

The sub clauses and tables in this section provide the details on how IEEE information is presented in the FHIR resources. In most tables the FHIR resource is on the left side of the table and the value to put into the table is in the adjacent column to the right. In some tables, there is a column with the

values "R", "S", or "O". These letters indicate if the mapping is required to be present ("R"), or should be present ("S"), or is optional ("O").

A.3.2.1 PHG System Id

The PHG shall encode its system id with the formatting in Table A-6 as specified in the *Guidelines for 64-bit Global Identifier (EUI-64)* found in [b-IEEE GL-EUI-64].

Table A-6 – PHG System Id Encoding

DeviceComponent Resource Structure		Value	
identifier.		If an additional identifier is also used, this element **shall** occur first	**R**,**S**,**O**, or **Z**
	use	"official"	R
	system	"urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680"	R
	value	*The PHG system Id as a 16-digit HEX string without the 0x prefix and each byte separated by dashes*	R

Example:

```
"identifier":[
                {
                    "use":"official",
                    "system":"urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680",
                    "value":"7E-ED-AB-EE-34-AD-BE-EF",
                }
            ],
```

More than one identifier is allowed. It should be something that uniquely identifies the PHG. Applications may add other identifiers such as the Bluetooth or ZigBee address.

A.3.2.2 PHG type

The DeviceComponent.type entry for the PHG is a CodeableConcept and is coded as follows:

Table A-7 – PHG Type Encoding

DeviceComponent Resource Structure			Value	R,S,O, or Z
type.				
	coding.		If an alternative coding is also used, this coding element **shall** occur first	
		code	531981	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	*"*MDC_MOC_VMS_MDS_AHD*" plus optional text*	R

Example:

```
            "type":{
                "coding":[
                    {
```

```
                        "system":"urn:std:iso:11073:10101",
                        "code":"531981",
                        "display":"MDC_MOC_VMS_MDS_AHD (CDG PHG)"
                    }
                ]
            },
```

Note that the display element is required in this case but only the reference id part of it. The inclusion gives a clear indication to the reader that this DeviceComponent resource refers to the gateway.

A.3.2.3 Production Specification

Reporting of production specification information by the PHG is optional. The manufacturer name and model number are included in the Production Specification list even though the IEEE 11073-20601 representation has them in their own attribute.

If these values are reported the DeviceComponent.productionSpecification element **shall** be used with Table A-8 and **shall** be encoded as in Table A-9.

Table A-8 – Production Specification Codes

Quantity to Enter	IEEE 11073 code	IEEE 11073 reference id
Model number	531969	MDC_ID_MODEL_NUMBER
Manufacturer name	531970	MDC_ID_MODEL_MANUFACTURER
Continua Version	532352	MDC_REG_CERT_DATA_CONTINUA_VERSION
Unspecified	531971	MDC_ID_PROD_SPEC_UNSPECIFIED
Serial number	531972	MDC_ID_PROD_SPEC_SERIAL
Part number	531973	MDC_ID_PROD_SPEC_PART
Hardware revision	531974	MDC_ID_PROD_SPEC_HW
Software revision	531975	MDC_ID_PROD_SPEC_SW
Firmware revision	531976	MDC_ID_PROD_SPEC_FW
Protocol	531977	MDC_ID_PROD_SPEC_PROTOCOL
Global Medical Device Nomenclature (GMDN)	531978	MDC_ID_PROD_SPEC_GMDN

Table A-9 – PHG Production Specification Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
productionSpecification.		
specType		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	Table A-8 *'IEEE 11073 code' for the 'quantity to enter'*	R
system	"urn.iso.std.iso:11073:10101"	R
display	Table A-8 *'IEEE 11073 reference id' for the 'quantity to enter' plus any optional text*	S
productionSpec	*The production spec value*	R

A.3.2.3.1 FHIR Production Specification Requirement

In version 3.0.1, the FHIR specification made the DeviceSpecificationSpecType value set 'extensible' for the productionSpecification.specType.coding.*code* element. "Extensible' means that if a production specification parameter is defined in the DeviceSpecificationSpecType value set, the code from it **shall** be used. Other coding systems are still allowed. This new requirement means that an additional productionSpecification.specType.coding entry must be added when the production specification is one of the following:

Table A-10 – FHIR DeviceSpecificationSpecType Codes

Quantity to Enter	FHIR code
Unspecified	unspecified
Serial number	serial-number
Part number	part-number
Hardware revision	hardware-revision
Software revision	software-revision
Firmware revision	firmware-revision
Protocol	protocol

The system value for this coding system is http://hl7.org/fhir/specification-type. When one of the above 'Quantity to Enter' parameters are to be mapped, the mapping is as follows:

Table A-11 – PHG Production Specification Encoding with DeviceSpecificationSpecType

DeviceComponent Resource Structure	Value	R,S,O, or Z
productionSpecification.	If the value being mapped has a code in the DeviceSpecificationSpecType value set	
specType		
coding.	The 11073 coding element **shall** always occur first	
code	Table A-8 *'IEEE 11073 code' for the 'quantity to enter'*	R
system	"urn.iso.std.iso:11073:10101"	R
display	Table A-8 *'IEEE 11073 reference id' for the 'quantity to enter' plus any optional text*	S
coding.		
code	Table A-10 *'FHIR code' for the 'quantity to enter'*	R
system	"http://hl7.org/fhir/specification-type"	R
productionSpec	*The production spec value*	R

Interestingly, this requirement is almost impossible to validate unless the validator knows all coding systems.

A.3.2.3.2 The Continua Version

The Continua version is present in the Reg-Cert-Data-List attribute. A PHG **shall** encode the Continua Version using the DeviceComponent.productionSpecification element as shown in Table A-12.

Table A-12 – PHG Continua Version Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
productionSpecification.		
specType.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	532352	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_REG_CERT_DATA_CONTINUA_VERSION" *plus any optional text*	S
productionSpec	*The Continua (major version).(minor version)*	R

A.3.2.4 Regulation Certification Information

The regulation certification information not mapped to the productionSpecification consists of

– The list of device specializations and transports the PHG has been *certified* for

– The list of H&FS interfaces the PHG has been *certified* for

– The regulation status of the PHG.

Note that the list of certifications is not necessarily equal to the list of supported features.

The DeviceComponent.property element is used to encode this information. The nomenclature codes used for this mapping are in Table A-12. The lists involve an array of property.value[x] fields.

The DeviceComponent.property element is defined as follows:

```
property.type                    CodeableConcept    [1..1]
        .valueQuantity           SimpleQuantity[0..*]
        .valueCode               CodeableConcept    [0..*]
```

Only a set of valueQuantity or a set of valueCode entries **shall** be populated. No property entry is to have both a valueQuantity and valueCode entry.

A.3.2.4.1 The List of Continua Certified Transports and Specializations

A PHG **shall** encode the list of transports and specializations it has been Continua certified for, but *only* if it has been certified, using Table A-13 as indicated in Table A-14.

Table A-13 – Transport Tcode Values

Transport	Tcode
Continua version 1.0	0
USB	1
Bluetooth HDP	2
ZigBee	3
Bluetooth Low Energy	4
NFC	5

Table A-14 – PHG Continua Certified Transports and Specializations Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	532353	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_REG_CERT_DATA_CONTINUA_CERT_DEV_LIST" *plus any optional text*	S
For each certified specialization:		
valueCode.		
coding.		
code	*The code value of the certified transport and specialization given by:* *Tcode * 8192 +16-bit mdc profile code – 4096* *Where the Tcode is given in Table A-13 and the 16-bit mdc profile code is one of the MDC_DEV_*_SPEC_PROFILE_* term codes representing the specialization*	R
system	"placeholder/fhir/reg-cert-codes" *(placeholder)*	R
display	*any optional text*	O

A.3.2.4.2 The List of Continua Certified H&FS Interfaces

A PHG **shall** encode the Continua Health & Fitness interfaces it has been certified for, but *only* if it has been certified, with help from Table A-15, as indicated in Table A-16.

Table A-15 – Health & Fitness Interface Codes

Health & Fitness Interface	code	Reference name
PCD-01 web services	0	observation-upload-soap
Consent enabled PCD-01 web service	1	consent-enabled-soap
Capability exchange	2	capabilities
PCD-01 upload using hData	3	observation-upload-hdata
Consent enabled PCD-01 using hData	4	consent-enabled-hdata
Questionnaire CDA	5	questionnaire
Authenticated Persistent Sessions	6	aps
FHIR resource upload	7	observation-upload-fhir

Table A-16 – PHG Continua Certified H&FS Interface Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	532355	R

DeviceComponent Resource Structure	Value	R,S,O, or Z
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_REG_CERT_DATA_CONTINUA_AHD_CERT_LIST" *plus optional text*	S
For each certified interface:		
valueCode.		
coding.		
code	*one of the codes from Table A-15*	R
system	"placeholder/fhir/reg-cert-wan-codes"*(placeholder)*	R
display	*The reference name for the interface code in Table A-15 corresponding to the code above plus any optional text*	S

A.3.2.4.3 The Regulation Status

The Regulation status is an ASN.1 BITS field with currently only one defined bit. A PHG **shall** encode its regulation status. The ASN.1 bit field mapping to codes is used. The bits **shall** be encoded as indicated in Table A-17.

Table A-17 – PHG Regulation Status Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	532354.0	R
system	"placeholder/fhir/IEEE.ASN1" *(placeholder)*	R
display	"regulation-bit-field" *plus any optional text*	S
valueCode.		
coding.		
code	"y" *(if set)* "n" *(if cleared)*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	"unregulated" *or* "regulated" *plus any optional text*	R

Note that this bit setting can be confusing as it is defined in the negative; a *set* bit is unregulated. For this reason, the display element of the yes/no binary coding element is required.

A.3.2.5 Time Synchronization

The time synchronization gives the PHG's method of time synchronization *if it is currently synchronized*. If the PHG is able to synchronize by a certain method but it is currently not synchronized, the reported time synchronization would be none and not the method it is capable of using to synchronize.

A property is used to report the time synchronization state. The list of possible time synchronization methods is defined in Table A-31 'Time Synchronization Related Nomenclature Codes". A PHG **shall** report its current synchronization state. The PHG **shall** encode its time synchronization state as follows:

Table A-18 – PHG Time Synchronization Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
name.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68220	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_TIME_SYNC_PROTOCOL" *plus optional text*	S
valueCode.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	*32-bit code of time synchronization method if synchronized otherwise* 532224 *(MDC_TIME_SYNC_NONE)*	R
system	"urn.iso.std.iso:11073:10101"	R
display	*reference id of synchronization code (see Table A-31) plus any additional optional text*	S

A.3.2.6 Time Info Information

A.3.2.6.1 Time Synchronization Accuracy

This measure provides an indication of the deviation of the actual time to that of the synchronized time in units of microseconds. A property is used to report this value. It **shall not** be reported if not known. If reported, the property **shall** be encoded as indicated in Table A-19.

Table A-19 – PHG Time Synchronization Accuracy Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
name.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68221	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_TIME_SYNC_ACCURACY" *plus any optional text*	S
valueQuantity.		
value	*accuracy in microseconds*	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

A.3.2.6.2 Time Resolution of PHG clock

The PHG **may** encode the time resolution of its clock. The time resolution represents the smallest difference between two time stamps the PHG can represent. All PHGs must effectively support a base offset time so the time resolution is coded using the base offset time resolution identifier code. If it is reported, it **shall** be reported in units of microseconds as indicated in Table A-20.

Table A-20 – PHG Time Resolution Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
name.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68226	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_TIME_RES_BO" *plus any optional text*	S
valueQuantity.		
value	*resolution in microseconds*	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

A.4 Sensor Properties Encoding Guidelines

In a IEEE 11073-20601 sensor the device properties are present in the MDS object. However, this specification makes no requirement that the mapped device have such an object. Any sensor device that can provide the information required to be mapped by this specification is permissible. CDG-compliant BTLE devices are examples of such non IEEE 11073-20601 sensors that may be able to be mapped according to these guidelines.

The static properties of the sensor are mapped to the FHIR DeviceComponent resource.

A.4.1 FHIR-Specific Requirements

FHIR requires that certain elements be present that are not related to information content.

A.4.1.1 Resource Name

All FHIR resources have a resourceType element. For the sensor mapping the resourceType is "DeviceComponent".

Example:

```
"resourceType":"DeviceComponent"
```

A.4.1.2 DeviceComponent Resource logical id

All FHIR resources have a logical *id* element which is an alpha-numeric string. The logical id is used to identify the resource in searches and when referenced by other resources. A FHIR application must treat this logical id as an opaque value and it only has meaning on a given server. There are no specific rules for the creation of this value. A logical id is always specified. If the logical id is

specified by the client (a PUT update transaction), the logical id **shall** be specified by the following string:

```
"[systemId as 16-digit HEX string].[transport address]"
```

where the transport address is

– a 12-digit HEX string for Bluetooth without the 0x prefix
– a 16-digit HEX string for ZigBee without the 0x prefix,
– a 4-digit VID HEX string followed by a 4-digit PID HEX string for USB separated by a dot without the 0x prefix.

If the system Id is not known, which may happen with some badly behaved BTLE devices, the 16-digit HEX string shall be set to all 0s. If the transport address is not known, it shall be set to all 0's.

Update example:

```
"id":"0022D6014AFBD418.00f1BA09b155"  (Bluetooth)
```

A.4.1.3 Profile

The meta.profile entry **shall** contain "placeholder/phdParentDeviceComponent".

NOTE – The actual profile URL is not yet specified.

A.4.2 Sensor Data Mapping

A.4.2.1 Sensor System-Id

The Sensor *System-Id* **shall** be placed in the DeviceComponent.identifier with the formatting indicated in Table A-21 as specified in the *Guidelines for 64-bit Global Identifier (EUI-64)* found in [b-IEEE GL-EUI-64].

Table A-21 – Sensor System-Id Encoding

DeviceComponent Resource Structure		Value	R,S,O, or Z
identifier.		If an additional identifier is also used, this element **shall** occur first	
	use	"official"	R
	system	"urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680"	R
	value	The MDS *System-Id* as a 16-digit HEX string without the 0x prefix and each byte separated by dashed	R

Example:

```
"identifier":[
        {
            "use":"official",
            "system":"urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680",
            "value":" 00-22-D6-01-4A-FB-D4-18",
        }
    ],
```

More than one identifier is allowed. It should be something that uniquely identifies the sensor. Applications may add other identifiers such as the Bluetooth or ZigBee address.

A.4.2.2 System-Type-Spec-List: Sensor type

The System-Type-Spec-List contains a list of the specializations that the sensor device supports. Because it may contain multiple specializations, multiple DeviceComponent resources may be needed.

A.4.2.2.1 Top-Level DeviceComponent

This DeviceComponent is always present. It contains the identifier, type, and productionSpecification elements. It also contains the DeviceComponent.*parent* element that points to the PHG-DeviceComponent. This resource is the one that specifies the phdParentDeviceComponent profile.

A.4.2.2.2 Child DeviceComponents

Child DeviceComponents are only present when there are multiple specializations or when there are specialization sub-profiles such as the step counter sub-profile of the cardiovascular specialization. When one has only multiple specializations, there will be a child DeviceComponent for each specialization, the child DeviceComponent.parent element will point to the top-level parent DeviceComponent, and the child DeviceComponent.type element will contain the specialization information. When one has a specialization sub-profile, the child DeviceComponent.parent element will point to the DeviceComponent containing the specialization it is a sub-profile of and the child DeviceComponent.type element will contain the specialization sub-profile. In this case it is possible to have a child of a child if the sensor also supports multiple specializations in addition to at least one specialization sub-profile. In this case the sub-profile child DeviceComponent would point to the child DeviceComponent representing the specialization which would point to the top-level parent DeviceComponent containing the specialization 'hydra'.

A.4.2.2.3 Profile

When a child DeviceComponent is present its meta.profile entry **shall** contain "placeholder/phdChildDeviceComponent".

NOTE – The actual profile URL is not yet specified.

A.4.2.2.4 Single Entry

If the System-Type-Spec-List contains a single entry, its value is mapped to the top-level DeviceComponent.type element. A System-Type-Spec-List contains the type code where the partition is understood to be 'infra' which has a value 4 as well as the specialization version. The top-level DeviceComponent.type is a CodeableConcept and **shall** be populated as follows:

Table A-22 – Single Entry Sensor System-Type-Spec-List Encoding

DeviceComponent Resource Structure			Value	R,S,O, or Z
type.				
	coding.		If an alternative coding is also used, this coding element **shall** occur first	
		code	8 * 2^{16} + System-Type-Spec-List[0].*type*	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	*"*MDC_DEV_*_SPEC_PROFILE_**" plus optional text*	S

Example:

```
"type":{
    "coding":[
        {
            "system":"urn:std:iso:11073:10101",
            "code":"528404",
            "version":"1",
            "display":" MDC_DEV_SPEC_PROFILE_BCA (Body composition analyzer)"
        }
    ]
},
```

A.4.2.2.5 Multiple Entries

If the system-type-spec-list contains multiple entries there will be at least one specialization and one or more of either specializations, sub-profiles, or both. A sub-profile is a refinement of a specialization and when there is a sub-profile, both the specialization and the sub-profile are required to be in the System-Type-Spec-List. If there is more than one *specialization*, the code "hydra" indicating multiple specializations is used in the top-level DeviceComponent. The actual specializations in the System-Type-Spec-List are encoded in separate child DeviceComponent resources. In these 'additional' child DeviceComponent resources only the child DeviceComponent.type and child DeviceComponent.*parent* elements are required to be populated.

If there is only one specialization and multiple sub-profiles, the single specialization is coded into the top-level DeviceComponent. If there are multiple specializations and at least one sub-profile, the sub-profile will be a child of a child DeviceComponent. A sub-profile DeviceComponent will never be a parent to another child DeviceProfile.

The 'hydra' specialization **shall** be encoded in the top-level DeviceComponent.type as indicated in Table A-23.

Table A-23 – HYDRA Specialization Encoding

DeviceComponent Resource Structure			Value	R,S,O, or Z
type.				
	coding.		If an alternative coding is also used, this coding element **shall** occur first	
		code	528384	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	*"*MDC_DEV_SPEC_PROFILE_HYDRA*" plus optional text*	S

Example:

```
"type":{
    "coding":[
        {
            "system":"urn:std:iso:11073:10101",
            "code":" 528384",
            "display":" MDC_DEV_SPEC_PROFILE_HYDRA (Multiple specializations)"
        }
    ]
},
```

A.4.2.2.6 Remaining Specializations and Sub-Profile Encoding

Child DeviceComponents are used to code each specialization contained in the System-Type-Spec-List when there is more than one entry and the top-level DeviceComponent is "hydra". See Figure A-1. If the top-level DeviceComponent is a specialization, the child DeviceComponents are used to encode the sub-profiles. For each specialization, the child DeviceComponent.type element **shall** be encoded as in Table A-22.

In addition, the child DeviceComponent.*parent* **shall** reference the parent DeviceComponent resource. The parent DeviceComponent could be a child or a top-level DeviceComponent.

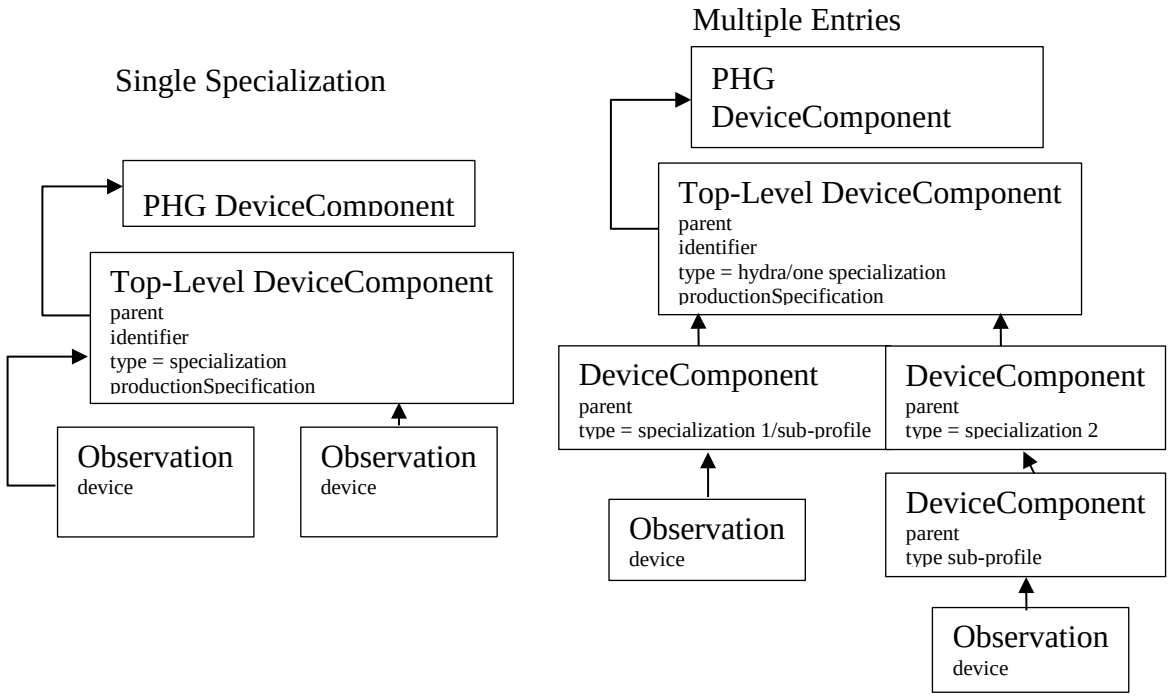

Figure A-1 – Single vs Multiple Specializations

In the multiple entry case, if the PHG encounters a specialization or sub-profile it does not know, the child DeviceComponent **shall** be generated, but the DeviceComponent.parent element shall be populated with the top-level DeviceComponent.

A.4.2.3 Production-Specification and System_Model Attributes

Reporting of the System-Model manufacturer name and model number and Production-Specification serial number and firmware revision **shall** be done if the sensor sends them. Note that the DeviceComponent.productionSpecification element is used for all these fields even though the IEEE 11073 sensor reports them in different attributes. Reporting the remaining production specification values is optional. If reported, the top-level DeviceComponent.productionSpecification element is used with information from Table A-24 and **shall** be encoded as indicated in Table A-25.

Table A-24 – Sensor Production Specification Mapping

Quantity to Enter	Code	Reference Id	IEEE 11073 attribute value
Model number	531969	MDC_ID_MODEL_NUMBER	System-Model.*model*
Manufacturer name	531970	MDC_ID_MODEL_MANUFACTURER	System-Model.*manufacturer*
Serial number	531972	MDC_ID_PROD_SPEC_SERIAL	*.prod_spec_serial-number*
Part number	531973	MDC_ID_PROD_SPEC_PART	*.prod_spec_part-number*
Hardware revision	531974	MDC_ID_PROD_SPEC_HW	*.prod_spec_hw-revision*

Quantity to Enter	Code	Reference Id	IEEE 11073 attribute value
Software revision	531975	MDC_ID_PROD_SPEC_SW	*.prod_spec_ sw-revision*
Firmware revision	531976	MDC_ID_PROD_SPEC_FW	*.prod_spec_fw-revision*
Protocol	531977	MDC_ID_PROD_SPEC_PROTOCOL	*.prod_spec_protocol-revision*
Global Medical Device Nomenclature (GMDN)	531978	MDC_ID_PROD_SPEC_GMDN	*.prod_spec_prod-spec-gmdn*
Unspecified	531971	MDC_ID_PROD_SPEC_UNSPECIFIED	*.prod_spec_unspecfied*

Table A-25 – Sensor Production Specification Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
productionSpecification.		
specType		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	Table A-24 *'code' for the 'quantity to enter'*	R
system	"urn.iso.std.iso:11073:10101"	R
display	*Table A-24 'reference id' for the 'quantity to enter' plus any optional text*	S
productionSpec	*The production spec value*	R

A.4.2.3.1 Additional FHIR Production Specification Requirement

In version 3.0.1, the FHIR specification made the DeviceSpecificationSpecType value set 'extensible' for the productionSpecification.specType.coding.*code* element. "Extensible' means that if a production specification parameter is defined in the DeviceSpecificationSpecType value set, the code from it **shall** be used. Other coding systems are still allowed. This new requirement means that an additional productionSpecification.specType.coding entry must be added when the production specification is one of the quantities to enter as specified in Table A-10. The system value for this coding system is http://hl7.org/fhir/specification-type. The mapping in this case is as indicated in Table A-26.

Table A-26 – PHD Production Specification Encoding with DeviceSpecificationSpecType Requirement

DeviceComponent Resource Structure	Value	R,S,O, or Z
productionSpecification.	If the value being mapped has a code in the DeviceSpecificationSpecType value set	
specType		
coding.	The 11073 coding element **shall** always occur first	
code	*Table A-8 'IEEE 11073 code' for the 'quantity to enter'*	R
system	"urn.iso.std.iso:11073:10101"	R
display	*Table A-8 ' IEEE 11073 reference id' for the 'quantity to*	S

DeviceComponent Resource Structure				Value	R,S,O, or Z
				enter' plus any optional text	
		coding.			
			code	*Table A-10 'FHIR code' for the 'quantity to enter'*	R
			system	"http://hl7.org/fhir/specification-type"	R
	productionSpec			*The production spec value*	R

A.4.2.3.2 The Continua Version

The Continua Version is present as a major and minor version in the Reg-Cert-Data-List attribute if the auth-body element has the value 2. The Continua Version is coded in the productionSpecification element. If the sensor provides this attribute the Continua Version **shall** be encoded as indicated in Table A-27.

Table A-27 – PHD Continua Version Encoding

DeviceComponent Resource Structure				Value	R,S,O, or Z
productionSpecification.					
	specType.				
		coding.			
			code	532352	R
			system	"urn.iso.std.iso:11073:10101"	R
			display	"MDC_REG_CERT_DATA_CONTINUA_VERSION" *plus any optional text*	S
	productionSpec			*The Continua (major version).(minor version)*	R

A.4.2.4 Reg-Cert-Data-List Attribute

The regulation certification information not mapped to the productionSpecification consists of

– The list of device specializations and transports the PHD has been *certified* for

– The regulation status of the PHD.

Note that the list of certifications is not necessarily equal to the list of supported features.

The top-level DeviceComponent.property element is used to encode this information.

A.4.2.4.1 The List of Continua Certified Specializations

The list of Continua Certified specializations is present as a list of Continua defined codes in the Reg-Cert-Data-List attribute if the auth-body element has the value 2. If the sensor provides this attribute, the specializations **shall** be encoded as indicated in Table A-28.

Table A-28 – Sensor Continua Certified Transports and Specializations Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	532353	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_REG_CERT_DATA_CONTINUA_CERT_DEV_LIST" *plus any optional text*	S
For each certified specialization:		
valueCode.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	*The code value of the certified transport and specialization given by:* *Tcode * 8192 +16-bit mdc profile code – 4096* *Where the Tcode is given in Table A-13 and the 16-bit mdc profile code is one of the MDC_DEV_*_SPEC_PROFILE_* term codes representing the specialization*	R
system	"placeholder/fhir/reg-cert-codes" *(placeholder)*	R
display	*any optional text*	O

A.4.2.4.2 The Regulation Status

The Regulation status is present as an ASN.1 BITS field in the Reg-Cert-Data-List if the auth-body element has the value 2. Currently there is only one defined bit. If the sensor provides this attribute, the bits **shall** be encoded using the ASN.1 BITs field mapping to codes as indicated in Table A-29.

Table A-29 – PHG Regulation Status Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	532354.0	R
system	"placeholder/fhir/IEEE.ASN1" *(placeholder)*	R
display	"regulation-bit-field" *plus any optional text*	S
valueCode.		
coding.		
code	"y" *(if set)* "n" *(if cleared)*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	"unregulated" *or* "regulated" *plus any optional text*	R

Note this bit setting is particularly confusing as it is defined in the negative; a *set* bit is unregulated. For this reason, the display element of the yes/no binary coding element is required.

A.4.2.5 The Mds-Time-Info Attribute

A sensor is required to support an Mds-Time-Info attribute if it supports any type of real time clock. This attribute contains information about the properties of the real time clock and its state of synchronization.

If the sensor provides this attribute, the state of time synchronization **shall** be reported. The time synchronization accuracy **shall** also be reported if the value reported indicates that it is known.

Reporting of the remaining information in this attribute is optional but it **shall not** be reported if the value indicates it is unknown.

The following Continua nomenclature codes for reporting the sub-structures of the Mds-Time-Info attribute over the H&FS interface are used in this encoding:

Table A-30 – Mds-Time-Info Sub-Structure Nomenclature Codes

32-bit code	Reference identifier	Description	partition:code
68220	MDC_TIME_SYNC_PROTOCOL	Synchronization method	1::2684
68219	MDC_TIME_CAP_STATE	Time capabilities	1::2683
68224	MDC_TIME_RES_REL_HI_RES	High resolution relative time resolution	1::2688
68223	MDC_TIME_RES_REL	Relative time resolution	1::2687
68222	MDC_TIME_RES_ABS	Absolute time resolution	1::2686
68226	MDC_TIME_RES_BO	Base offset time resolution	1::2690
68221	MDC_TIME_SYNC_ACCURACY	Time synchronization accuracy	1::2685

A.4.2.5.1 Time Synchronization

When reporting the time synchronization, it is always reported as unsynchronized unless the sensor is synchronized. The Mds-Time-Info.*time-sync-protocol* field does NOT give the state of time synchronization but the method of time synchronization supported by the sensor. To find the time synchronization one needs to look at the Mds-Time-Info.*mds-time-cap-state* field. This ASN.1 BITs field has four bits that indicate whether the sensor is currently synchronized to absolute, relative, hi-resolution relative, or base offset time, respectively. If one of these bits is set, the time synchronization is given by the code in the Mds-Time-Info.*time-sync-protocol* field, otherwise the sensor is not synchronized.

Table A-31 lists the currently defined time synchronization state codes.

Table A-31 – Time Synchronization Related Nomenclature Codes

32-bit code	Reference identifier	Description	partition:code
532224	MDC_TIME_SYNC_NONE	An uncalibrated and unsynchronized local clock source	8::7936
532234	MDC_TIME_SYNC_EBWW	A manually set time, by 'eyeball and wristwatch'	8::7946
532225	MDC_TIME_SYNC_NTPV3	Network Time Protocol Version 3.0 (RFC 1305)	8::7937
532226	MDC_TIME_SYNC_NTPV4	Network Time Protocol Version 4.0 (under dev)	8::7938
532227	MDC_TIME_SYNC_SNTPV4	Simple Network Time Protocol v4 (RFC 2030)	8::7939
532228	MDC_TIME_SYNC_SNTPV4330	Simple Network Time Protocol v4	8::7940

32-bit code	Reference identifier	Description	partition:code
		(RFC 4330)	
532229	MDC_TIME_SYNC_BTV1	Bluetooth Medical Device Profile	8::7941
532235	MDC_TIME_SYNC_USB_SOF	Synced to the 1kHz USB "start-of-frame" clock	8::7947
532230	MDC_TIME_SYNC_RADIO	Atomic Clock synchronization through RF	8::7942
532231	MDC_TIME_SYNC_HL7_NCK	Synchronized via Health Level 7 NCK (network clock)	8::7943
532232	MDC_TIME_SYNC_CDMA	CDMA mobile telecommunications synchronization	8::7944
532233	MDC_TIME_SYNC_GSM	GSM - Network Identity and Time Zone (NITZ)	8::7945

Encoding the time synchronization state involves the following algorithm:

If one of

Mds-Time-Info.*mds-time-cap-state field*_mds-time-state-abs-time-synced(8)

Mds-Time-Info.*mds-time-cap-state field*_mds-time-state-rel-time-synced(9)

Mds-Time-Info.*mds-time-cap-state field*_mds-time-state-hi-res-relative-time-synced(10)

Mds-Time-Info.*mds-time-cap-state field*_mds-time-state-bo-time-synced(13)

is set, the time synchronization is given by Mds-Time-Info.*mds-time-sync-protocol*; otherwise the sensor is unsynchronized.

If the sensor provides an Mds-Time-Info attribute, the time synchronization **shall** be encoded as follows:

Table A-32 – Sensor Time Synchronization Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68220	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_TIME_SYNC_PROTOCOL" *plus any optional text*	S
valueCode.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	*If sensor is synchronized* Mds-Time-Info.*mds-time-sync-protocol* *else* 532224 *(unsynchronized)*	R

DeviceComponent Resource Structure	Value	R,S,O, or Z
system	"urn.iso.std.iso:11073:10101"	R
display	*"*Reference identifier*" from Table A-31 corresponding to the value of the code element plus any optional text*	S

A.4.2.5.2 Time Synchronization Accuracy

The time synchronization accuracy is reported in the Mds-Time-Info.*time-sync-accuracy* field. If the value of this field is 0xFFFFFFFF it is unknown and **shall not** be reported. Otherwise, it **shall** be encoded in units of microseconds as indicated in Table A-33.

Table A-33 – Sensor Time Synchronization Accuracy Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68221	R
system	"urn.iso.std.iso:11073:10101"	R
display	"MDC_TIME_SYNC_ACCURACY" *plus any optional text*	S
valueQuantity.		
value	Mds-Time-Info.*time-sync-accuracy scaled to microseconds*	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

A.4.2.5.3 Time Capabilities

If the sensor provides an Mds-Time-Info attribute, the time capabilities **may** be reported. The capabilities are an ASN.1 BITs field. If the capabilities are reported, which capabilities are reported is up to the application. In general, only the set static cases are reported. For those capabilities that are reported, the ASN.1 BITs **shall** be mapped to the codes as shown in Table A-34.

Table A-34 – Codes for Time Capabilities Vocabulary

Bit position	Code	ASN.1 name
0	68219.0	mds-time-capab-real-time-clock
1	68219.1	mds-time-capab-set-clock
2	68219.2	mds-time-capab-relative-time
3	68219.3	mds-time-capab-high-res-relative-time
4	68219.4	mds-time-capab-sync-abs-time
5	68219.5	mds-time-capab-sync-rel-time
6	68219.6	mds-time-capab-sync-hi-res-relative-time
7	68219.7	mds-time-capab-bo-time
8	68219.8	mds-time-state-abs-time-synced*****

Bit position	Code	ASN.1 name
9	68219.9	mds-time-state-rel-time-synced*****
10	68219.10	mds-time-state-hi-res-relative-time-synced*****
11	68219.11	mds-time-mgr-set-time*****
12	68219.12	mds-time-capab-sync-bo-time
13	68219.13	mds-time-state-bo-time-synced*****
14	68219.14	mds-time-state-bo-time-UTC-aligned
15	68219.15	mds-time-dst-rules-enabled

*****These settings are dynamic, in particular the *mds-time-mgr-set-time* which indicates that the PHG shall set the sensor time, which, by protocol, must be cleared after being set.

and encoded, for each bit wished to be reported, as indicated in Table A-35.

Table A-35 – Sensor Time Capabilities Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.		
code	68219.*i*	R
system	"placeholder/fhir/IEEE.ASN1" *(placeholder)*	R
display	"ASN.1 name from Table A-34" *plus any optional text*	S
valueCode.		
code	"y" *(if set)* "n" *(if cleared)*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	*optional text*	O

A.4.2.5.4 Time Resolution

Mds-Time-Info has fields that report the resolution of its time clocks.

The Mds-Time-Info.*time-resolution-abs-time* represents the resolution of the absolute-time clock when the sensor supports an absolute time clock. If the sensor supports a base-offset time clock it represents the resolution of the base-offset time clock. The sensor is not able to support both time clocks simultaneously. Which time clock is supported is indicated by the settings of the Mder 0 and 7 bits of the Mds-Time-Info.*mds-time-caps-state*.

The sensor may *additionally* support both relative time clocks or just the relative time clocks without any 'wall time' clock. In practice, support of more than one real time clock at the same time is rare.

If the respective time resolution has a value of 0, it indicates that the resolution is unknown and it **shall not** be reported. In all other cases, the resolutions **may** be reported.

When reported, all time resolutions values **shall** be scaled to units of microseconds. When supporting absolute time, the Mds-Time-Info.*time-resolution-abs-time* is in units of $1/100^{th}$ of a second. When supporting base-offset time, the Mds-Time-Info.*time-resolution-abs-time* is in units of $1/65536^{th}$ of a second. In the base-offset case, the special value of 0xFFFF means one second. The Mds-Time-Info.*time-resolution-rel-time* is in units of $1/8^{th}$ millisecond and the Mds-Time-Info.*time-resolution-hi-res-relative-time* is in units of microseconds.

If the time resolutions are reported, each time resolution the application wishes to report **shall** be encoded as indicated in Table A-36.

Table A-36 – Sensor Time Resolution Encoding

DeviceComponent Resource Structure	Value	R,S,O, or Z
property.		
type.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	*One of* *68222 (absolute time)* *68226 (base-offset time)* *68223 (relative time)* *68224 (hi-res relative time)*	R
system	"urn.iso.std.iso:11073:10101"	R
display	*One of* "MDC_TIME_RES_ABS" "MDC_TIME_RES_BO" "MDC_TIME_RES_REL" "MDC_TIME_RES_REL_HI_RES" *plus any optional text*	S
valueQuantity.		
value	*The corresponding* Mds-Time-Info.*time-resolution-*-time* *value scaled to microseconds*	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

A.4.2.6 The Dynamic MDS attributes

The remaining informational attributes are encoded into Observation resources. Of these the following are reported in the coincident time stamp Observation:

– Date-and-Time (current absolute time)

– Base-Offset-Time (current base-offset time)

– Relative-Time (current relative time)

– HiRes-Relative-Time (current hi-resolution relative time)

The coincident time stamp Observation is conditionally required.

The remaining attributes:

– Battery-Level

– Remaining-Battery-Time

– Power-Status

are optional but if encoded **shall** follow the guideline specified here.

The MDS dynamic/observational attributes can be read at association but may change during an association and get evented to the PHG in a scan event report. However, current time attribute updates without a Date-Time-Adjustment **shall not** be reported as that may result in confusion. Time-line changes, indicated by a Date-Time-Adjustment, result in a new coincident time stamp Observation resource. Observation resources generated after that time point to the new coincident time stamp Observation.

A.4.2.6.1 Battery-Level

If reported, the MDS Battery-Level **shall** be reported in an Observation resource as shown in Table A-37.

Table A-37 – Battery-Level Encoding

Observation Resource			Value	R,S,O, or Z
identifier.			*Not necessary since this value is always current. Duplicate detection is not needed.*	O
status			"final"	R
code.				
	coding.		If an alternative coding is also used, this coding element **shall** occur first	
		code	67996	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	MDC_ATTR_VAL_BATT_CHARGE *plus optional text*	S
subject			*Reference to the Patient Resource*	R
effectiveDateTime			*The time of the PHG including offset to UTC at which the battery attribute was read or the value was received.*	R
valueQuantity.				
	value		Battery-Level.*value*	R
	units		*%*	S
	system		"urn.iso.std.iso:11073:10101"	R
	code		262688	R
device			Reference to the parent DeviceComponent resource	R

A.4.2.6.2 Remaining-Battery-Time

If reported, the MDS Remaining-Battery-Time **shall** be reported in an Observation resource as shown in Table A-38.

Table A-38 – Remaining-Battery-Time Encoding

Observation Resource		Value	R,S,O, or Z
identifier.		*Not necessary since this value is always current. Duplicate detection is not needed.*	O
status		"final"	R
code.			
	coding.	If an alternative coding is also used, this coding element **shall** occur first	

Observation Resource			Value	R,S,O, or Z
		code	67956	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	MDC_ATTR_TIME_BATT_REMAIN *plus optional text*	S
subject			*Reference to the Patient Resource*	R
effectiveDateTime			*The time of the PHG including offset to UTC at which the battery time remaining attribute was read or the value was received.*	R
valueQuantity.				
	value		Remaining-Battery-Time.*value*	R
	units		*either min (minutes), hr (hours), or days.*	S
	system		"urn.iso.std.iso:11073:10101"	R
	code		$4*2^{16}$ + Remaining-Battery-Time.*unit* The attribute reports the time in units of MDC_DIM_MIN (minutes), MDC_DIM_HR (hours), or MDC_DIM_DAY (days)	R
device			Reference to the parent DeviceComponent resource	R

A.4.2.6.3 Power-Status

The Power-Status reports its values in an BITS-16 format. Thus the BITs to ANS.1 BITs encoding is used. The possible codes are as indicated in Table A-39.

Table A-39 – Codes for Power Status Vocabulary

Bit position	Code	ASN.1 name
0	67925.0	onMains
1	67925.1	onBattery
8	67925.8	chargingFull
9	67925.9	chargingTrickle
10	67925.10	chargingOff

If reported, the Power-Status values **shall** be reported in an Observation resource as indicated in Table A-40.

Table A-40 – Power-Status Encoding

Observation Resource			Value	R,S,O, or Z
identifier.			*Not necessary since this value is always current. Duplicate detection is not needed.*	O
status			"final"	R
code.				
	coding.		If an alternative coding is also used, this coding element **shall** occur first	
		code	67925	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	MDC_ATTR_POWER_STAT *plus optional text*	S
subject			*Reference to the Patient Resource*	R

Observation Resource	Value	R,S,O, or Z
effectiveDateTime	*The time of the PHG including offset to UTC at which the battery time remaining attribute was read or the value was received.*	R
value[x]	Not reported	R
device	Reference to the parent DeviceComponent resource	R
For each bit that is defined and reported (only set values **shall** be reported)		
component.		
code.		
coding.		
code	*Set to the ASN1code computed from Table A-39*	R
system	"placeholder/fhir/IEEE.ASN1"	R
display	*ASN.1 name from Table A-39 plus optional text*	O
valueCodeableConcept.		
coding.		
code	"y" *if set,* "n" *if cleared*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	*optional text*	O

A.5 The Coincident Time Stamp

The coincident time stamp is reported in an Observation resource. It contains a measure of the sensor's current time taken by the PHG. The timestamp of this measurement is the PHG's current time. Measurement Observation resources point to this coincident time stamp Observation unless the sensor does not report a time stamp for that measurement. If the sensor does not use time stamps in any of its measurements, the coincident time stamp Observation resource is not generated.

The presence and/or absence of the coincident time stamp Observation with respect to an Observation indicates

- if the measurement time stamps have been corrected by the PHG and by how much:
  - the difference between the PHG and PHD current times gives the amount of the correction
- if the sensor reports no time stamps and the measurement time stamps are the time of reception by the PHG:
  - the measurement points to no coincident time stamp,
- if there has been a time fault and the measurement time stamps are questionable:
  - The PHD current time is not recorded in the coincident time stamp and the dataValueAbsent reason indicates 'unknown'
- and if the measurement time stamps are those of the sensor because the sensor is superiorly synchronized:
  - The PHG current time is not recorded in the coincident time stamp.

A.5.1 Generating the Coincident Time Stamp

A PHG obtains the current time of the sensor and records its own current time as it performs this operation. These two time stamps are referred to as the coincident time stamp pair. All CDG compliant PHD devices must be able to report a current time unless it does not use time stamps in its measurements. In addition, all CDG compliant PHD devices that store data **shall** have timestamps.

The generation of the coincident time stamp Observation and the resultant time corrections (if any) depends upon the following factors:

– Which endpoint is more accurately synchronized to NTP time
– Which of the possible time formats the sensor supports:
  - no time
  - absolute time
  - base-offset time aligned to UTC
  - base-offset time NOT aligned to UTC
  - relative time
  - high-resolution relative time
– Whether the PHG sets the sensor time.

A.5.2 FHIR-Specific Requirements

A.5.2.1 The logical id

If a logical id is needed, the Observation.id **shall** be populated with a temporary id. The coincident time stamp Observation is always 'created'.

A.5.2.2 Status

The Observation.*status* **shall** be set to 'final'.

A.5.2.3 Profile

The meta.profile entry **shall** contain "placeholder/phdCoincidentTimeStampObservation".

NOTE – The actual profile URL is not yet specified.

A.5.3 Determining Superior Synchronization

The PHG examines the Mds-Time-Info.*mds-time-cap-state* field of the Mds-Time-Info attribute to see if the sensor is synchronized to an external time source. If the capabilities indicate either that the absolute time or base offset time is synchronized, the PHG examines the Mds-Time-Info.*time-sync-protocol* field and the Mds-Time-Info.*time-sync-accuracy* field. The PHG compares these two values with its own values. All NTP time synchronization protocols are considered equal for this comparison and they are superior to all other time synchronization protocols. The time synchronization accuracy is used to break any tie. If the accuracies are equal, the sensor is considered to have the superior timeline with respect to the construction of the coincident time stamp Observation all else being equal. For all other cases, the PHG is considered to have the superior timeline.

If the sensor is not synchronized to any external source, the time synchronization protocol is treated as none regardless of the Mds-Time-Info.*time-sync-protocol* field value. The Mds-Time-Info.*time-sync-protocol* reports the *method* of synchronization *when* synchronized but for the coincident time stamp Observation, it is considered none unless the sensor is synchronized which is indicated in the Mds-Time-Info. *mds-time-cap-state*.

A.5.4 PHG has Superior Synchronization

For PHD devices this outcome is the most likely. At the time of the writing of this specification, there are no PHD sensor devices that synchronize to an external time source. The time is either set manually on the device, or the PHG is asked to set the time, or there is a factory default.

A.5.4.1 Sensor uses Absolute Time

If the sensor reports the time in absolute time format the PHG assumes that the obtained current sensor time refers to the current time zone of the PHG. The PHG **shall** encode the coincident time stamp pair as indicated in Table A-41.

Table A-41 – Absolute Time Coincident Time Stamp Encoding

Observation Resource Structure	Value	R,S,O, or Z
code.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	67975	R
system	"urn.iso.std.iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_ABS*" plus optional text*	S
effectiveDateTime	*The current time of the PHG including offset to UTC at the time of reading the sensor current time*	R
valueDateTime	*Date-and-Time* converted to a FHIR dateTime data type*. *This element is absent if there is a time fault* *This element is set to the effectiveDateTime if the PHG set the sensor time and the set is successful*	R

* NOTE – The time stamp must include the UTC offset (provided by the PHG) as it is required by FHIR, even though an absolute time has no offset.

In this case the PHG corrects the sensor measurement time stamps such that they are on the same time line as the PHG. Since the agent reports an absolute time, all reported measurement time stamps will have the current time zone. Corrections are derived from the local time differences. When the PHG sets the sensor time, the local time difference becomes zero so there is no correction.

A.5.4.2 Sensor uses Base-Offset Time

If the sensor reports the time in base offset time format the PHG assumes that the obtained current sensor time refers to the current time zone of the PHG.

– If the Mds-Time-Info.*mds-time-cap-state* indicates the sensor base-offset time is UTC aligned, the PHG **shall** correct the base time and offset independently.

– If the Mds-Time-Info.*mds-time-cap-state* indicates that the sensor base-offset time is not UTC-aligned, the PHG **shall** assume that the sensor offset is correct. This assumption allows the PHG to ascertain the offset of the base time from UTC allowing the PHG to map the sensor base offset time to UTC-aligned base offset time.

The PHG **shall** encode the coincident time stamp pair as indicated in Table A-42.

Table A-42 – Base Offset Time Coincident Time Stamp Encoding

Observation Resource Structure	Value	R,S,O, or Z
code.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68226	R
system	"urn.iso.std.iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_BO*" plus optional text*	S
effectiveDateTime	*The current time of the PHG including offset to UTC at the time of reading the sensor current time*	R
valueDateTime	*Base-Offset-Time* converted to a FHIR dateTime data type. *This element is absent if there is a time fault* *This element is set to the effectiveDateTime if the PHG set the sensor time and the set is successful*	R

The PHG corrects the sensor measurement time stamps such that they are on the same time line as the PHG.

A.5.4.3 Sensor uses Relative Time

When a sensor uses a relative time, the coincident time stamp pair allows the PHG to map the sensor measurement relative times to wall clock time. The PHG **shall** encode the coincident time stamp pair as indicated in Table A-43.

Table A-43 – Relative Time Coincident Time Stamp Encoding

Observation Resource Structure	Value	R,S,O, or Z
code.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	67983	R
system	"urn.iso.std.iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_REL*" plus optional text*	S
effectiveDateTime	*The current time of the PHG including offset to UTC at the time of reading the sensor current time*	R
valueQuantity.	Element is absent if a time fault	
value	*Relative-Time * 125* to convert to microseconds	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

The PHG computes the wall clock time from the sensor measurement time stamps by using the values in the coincident time stamp as the reference.

A.5.4.4 Sensor uses High Resolution Relative Time

When a sensor uses a high resolution relative time, the coincident time stamp pair allows the PHG to map the sensor measurement high resolution relative times to wall clock time. The PHG **shall** encode the coincident time stamp pair as indicated in Table A-44.

Table A-44 – Hi-Res Relative Coincident Time Stamp Encoding

Observation Resource Structure	Value	R,S,O, or Z
code.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	
code	68072	R
system	"urn.iso.std.iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_REL_HI_RES*" plus optional text*	S
effectiveDateTime	*The current time of the PHG including offset to UTC at the time of reading the sensor current time*	R
valueQuantity.	Element is absent if a time fault	
value	*HiRes-Relative-Time* which is in microseconds	R
units	"us" *(UCUM code for microseconds)*	R
system	"urn.iso.std.iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds)*	R

The PHG computes the wall clock time from the sensor measurement time stamps by using the values in the coincident time stamp as the reference.

A.5.5 Sensor has Superior Synchronization

When the sensor has superior time synchronization, the PHG reports the measurement time stamps from the sensor uncorrected. This situation may also happen because the PHG has somehow lost its time synchronization.

A.5.5.1 Sensor uses Base-Offset Time

The PHG **shall** encode the coincident time stamp pair as it does for the case when the PHG has superior time synchronization except the Observation.*effectiveDateTime* element is absent. The PHG uses the sensor provided time stamp from the measurements unaltered.

A.5.5.2 Sensor uses Absolute Time

The PHG **shall** encode the coincident time stamp pair as it does for the case when the PHG has superior time synchronization except the Observation.*effectiveDateTime* element is absent. The offset to UTC in the Observation.*valueDateTime* element must still be provided by the PHG.

The PHG uses the sensor provided time stamp from the measurements unaltered except for providing the offset to UTC.

A.5.5.3 PHG Sets the time

In this case, the current time of the sensor is set to the current time of the PHG and there is no correction applied to the measurement time stamps from the sensor.

A.5.6 Sending the Coincident Time Stamp

The Coincident Time Stamp is always uploaded using a create (POST) operation since its primary role is to compare the current time clocks of the sensor with the PHG at the time the sensor connects with the PHG.

If there are no Observations uploaded that reference the Coincident Time Stamp Observation the Coincident Time Stamp Observation **shall not** be sent.

A.6 Guidelines for Encoding the Metric Measurements

All measurements stemming from [ISO/IEEE 11073-20601] metric objects are mapped to FHIR Observation resources. In this specification, the mapping to Observation resources is defined in terms of these metric objects and their concomitant attributes. *There is no requirement that these objects or attributes exist on the sensor device.* Any sensor device whose data can be mapped to these objects and attributes by the PHG following these guidelines can be translated to compliant FHIR Observation resources.

Table A-45 gives a conceptual relationship between the IEEE 11073-20601 metric object concepts and the corresponding FHIR Observation resource elements.

Table A-45 – Metric Object Concepts to FHIR Observation Concepts Table

Metric Object Concepts	FHIR Observation elements
measurement type	Observation.code.coding
measurement value	
numeric	Observation.valueQuantity.*value*
units	Observation.valueQuantity.*unit*
compound numeric	
compound sub-type	Observation.component.code.coding
compound sub-value	Observation.component.valueQuantity.*value*
compound sub-units	Observation.component.valueQuantity.*unit*
RTSA	Observation.valueSampledData
Enumeration OID	Observation.valueCodeableConcept
Enumeration ASN.1	Observation.valueCodeableConcept
Enumeration string	Observation.valueString.*value*
measurement time stamp	Observation.effectiveDateTime
measurement status reports error	Observation.dataAbsentReason
supplemental types	Observation.component.code.coding = *Supplemental types MDC code* Observation.component.valueCodableConcept Repeat for each supplemental type code
time stamp is relative or hi-res relative	Observation.component.code.coding = *Relative Time stamp MDC code* Observation.component.valueQuantity.*value* Observation.component.valueQuantity.*unit*
label string attributes	Observation.text
Source handle reference	Observation.related.*target*

Table A-45 expresses the relationship between the IEEE 11073-20601 Metric Object attributes (at least in concept) and a FHIR Observation resource. Every defined Metric Object has one attribute

representing the measurement value which may be a quantity, a set of quantities, a periodic sequence of quantities, a code, or a human readable text string. In a FHIR Observation resource, the measurement value maps to the value[x] element or a set of component.value[x] elements. Every Metric Object also has a Type attribute, that describes what the measurement value is, which is mapped to a code element though the final type may be further modified by other attributes. As far as the CDG is concerned, every measurement value will have an associated time stamp (if the sensor does not provide a timestamp the CDG PHG generates one from the time of reception) which is mapped to the effectiveDateTime element. If the Metric Object exposes a *Measure-Active-Period* attribute, the time stamp and the active period are mapped to the effectivePeriod element.

The Metric Object may also have additional attributes that further describe the measurement value. For example, the Metric-Id-List attribute contains additional information describing each sub-value of a compound numeric such as the x, y, and z components of an acceleration while the Supplemental-Types attribute may indicate that these sub-values are time averages over the measurement period. These additional descriptions are mapped to FHIR Observation component elements.

A.6.1 Measurement Type

In the FHIR Observation resource, the Observation.code describes the measurement type. To obtain the measurement type from the metric object in all IEEE 11073-20601 supported situations requires the examination of the Type, *Metric-Id*, *Metric-Partition-Id*, Nu-Observed-Value and Enum-Observed-Value attributes. Of these attributes, only the Type attribute is always present. The Nu-Observed-Value and Enum-Observed-Value are complex observational attributes with their own nomenclature code sub-structure which takes precedence over the nomenclature codes from the Type and *Metric-Id* attributes. The algorithm is shown in the Measurement Type Mapping table.

In addition, if the measurement type refers to a vital sign, FHIR requires that the LOINC code for the vital sign be present. Table A-46 lists the LOINC codes for vital signs.

Table A-46 – LOINC Codes for Vital Signs

Blood Pressure (overall)	55284-4
Systolic Blood Pressure	8480-6
Diastolic Blood Pressure	8462-4
Height	8302-2
Oxygen Saturation	59408-5
Pulse Rate (all pulse rates)	8867-4
Respiration Rate	9279-1
Temperature	39156-5
Weight	8310-5

The Observation.code element **shall** be set as indicated in Table A-47, where the *termCode* is the 16 least significant bits and the *partition* is the 16 most significant bits of the 32-bit nomenclature code:

Table A-47 – Measurement Type Mapping

Observation Resource	Value	R,S,O, or Z
code.		
coding.	If an alternative coding is also used, this coding element **shall** occur first	

Observation Resource			Value	R,S,O, or Z
		code	*termCode* = Type.*code* *partition* = Type.*partition* *if Metric-Id* or Nu-Observed-Value or Enum-Observed-Value attribute exits: { *if* Nu-Observed-Value or Enum-Observed-Value exists *termCode* = Nu-Observed-Value.*metric-id* or *termCode* = Enum_Observed-Value.*metric-id* *else if Metric-Id* is present: *termCode = Metric-Id* *if Metric-Id-Partition* exists: { *partition = Metric-Id-Partition* } } *then the code is set to:* *partition* * 2^{16} + *termCode*	R
		system	"urn.iso.std.iso:11073:10101"	R
		display	*Reference Id for the code plus optional text*	S
If the measurement type is a vital sign, FHIR requires a translation to LOINC				
	coding.		If an additional alternative coding is also used, this coding element **shall** occur second	
		code	*LOINC code for the above vital sign*	R
		system	"http://loinc.org"	R
		display	*optional text*	O

A.6.2 The Measurement Values

The measurement value is placed in the *.value[x] element, where 'x' depends upon whether the value is a quantity, code, a set of periodically sampled data, or a string.

A.6.2.1 Non-Compound Numeric Value

If the measurement value attribute is a *Basic-Nu-Observed-Value* (contains an Mder SFLOAT) or *Simple-Nu-Observed-Value* (contains an Mder FLOAT) or Nu-Observed-Value (value element is an Mder FLOAT) the measurements are quantities.

A.6.2.1.1 Profile

The meta.profile entry **shall** contain "placeholder/phdNumericObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.1.2 Encoding

The attribute *value* **shall** be encoded into a string with the precision given by the Mder SFLOAT or FLOAT precision. Table A-48 illustrates how one decodes Mder floats to the encoded precision.

NOTE – A negative Mder exponent gives the number of significant figures to the right of the decimal point and a positive or 0 exponent indicates an integer with no decimal point. The mantissa gives the number of significant figures. Examples:

Table A-48 – Mder Float Decodings

Mder exponent	Mder mantissa	Encoded Value as string
-3	0	0.000
0	0	0
-3	+/-2100	+/-2.100
-2	+/-2100	+/-21.00
-1	+/-2100	+/-210.0
0	+/-2100	+/-2100
1	+/-2100	+/-21000
-3	+/-2000	+/-2.000
-2	+/-200	+/-2.00
-1	+/-20	+/-2.0

The valueQuantity **shall** be encoded as indicated in Table A-49.

Table A-49 – Single Numeric Measurement Mapping

Observation Resource	Value	R,S,O, or Z
valueQuantity.	**shall not** be present if error or a special value:	
value	*value* is one of *Basic-Nu-Observed-Value* *Simple-Nu-Observed-Value* Nu-Observed-Value.*value* *decoded from the SFLOAT or FLOAT with the precision given by the respective Mder encoding*	R
unit	*UCUM string for the unit code*	S
code	*unitTermCode = Unit-Code* *if* Nu-Observed-Value { *unitTermCode* = Nu-Observed-Value.*unit* } *then code is set to (the unit code partition is always 4)* $4 * 2^{16} + unitTermCode$	R
system	"urn:iso:std:iso:11073:10101"	R
dataAbsentReason.	**shall not** be present if no error or no special value:	
coding.		
code	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

Special notes:

– If the dataAbsentReason is present, the value[x] **shall** be absent

– If the valueQuantity is present, the dataAbsentReason **shall** be absent

– See the section on the handling of special values and the handling of errors reported by the Measurement-Status attribute or measurement status from the complex observation attributes for a mapping from the error condition to the available data absent reason codes.

A.6.2.2 Compound Numeric Value

A compound measurement contains N sub-values such as the x, y, and z components of an acceleration or the systolic, diastolic, and mean components of the blood pressure. In addition to the overall type, compound measurements need N sub-types to describe each sub-value. The N sub-values and sub-types are mapped into N Observation.component elements. If the attribute is a Compound-Basic-Nu-Observed-Value (contains N Mder SFLOATs), Compound-Simple-Nu-Observed-Value (contains N Mder FLOATs), or Compound-Nu-Observed-Value (contains N Mder FLOATs) it contains N sub-values that are quantities.

A.6.2.2.1 Profile

The meta.profile entry **shall** contain "placeholder/phdCompoundNumericObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.2.2 Encoding

The N sub-types for each of the N sub-values are obtained from the Metric-Id-List which also has N entries. There is a one to one correspondence between the N^{th} entry of the Compound attribute and the N^{th} entry of the Metric-Id-List. If there is a Compound-Nu-Observed-Value, the sub-type is given by the Compound-Nu-Observed-Value.*metric-id* value which takes precedence over the Metric-Id-List should it exist. The partition of the sub-type is given by the partition of the Type attribute unless there is a Metric-Id-Partition attribute.

The 'primary' Observation.code element **shall** be populated from the Type attribute. In this case, no other attributes need to be examined to determine the value of this entry.

The Compound measurement quantities **shall** be encoded as indicated in Table A-50.

Table A-50 – Compound Numeric Measurement Mapping

Observation Resource	Value	R,S,O, or Z
value[x]	***shall not*** *be encoded*	R
dataAbsentReason	***shall not*** *be encoded (no appropriate reason exists)*	R
For each of the N sub-entries		
component.	The components containing the compound measurements **shall** occur before any additional component elements	
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	*partition* = Type.*partition* *termCode* = Metric-Id-List[*n*].*value* *if* Compound-Nu-Observed-Value exits: { *termCode* = Compound-Nu-Observed-Value[*n*].*metric-id* } *if Metric-Id-Partition* exists: { *partition = Metric-Id-Partition*	R

Observation Resource	Value	R,S,O, or Z
	} *then the code is set to:* *partition* * 2^{16} + *termCode*	
system	"urn:iso:std:iso:11073:10101"	R
display	*Reference identifier for code plus optional text*	S
If a vital sign		
coding.	If an additional alternative coding system is used, this element **shall** occur second	
code	*LOINC code for corresponding vital sign*	R
system	"http://loinc.org"	R
display	*Some text*	O
valueQuantity.	**shall not** be present if a special value or an error:	
value	Compound-*-Nu-Observed-Value[*n*].*value* *decoded from the SFLOAT or FLOAT with the precision given by the respective Mder encoding*	R
units	*UCUM string for the unit code*	S
system	"urn:iso:std:iso:11073:10101"	R
code	*unitTermCode* = *Unit-Code* *if* Compound-Nu-Observed-Value { *unitTermCode* = Compound-Nu-Observed-Value[*n*].*unit* } *then code is set to (the unit code partition is always 4)* 4 * 2^{16} + *unitTermCode*	R
dataAbsentReason.	**shall not** be present if no error or no special value:	
coding.		
code	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

Special notes and cautions:

– The Observation.code is obtained from Type attribute alone

– If the Observation.component.dataAbsentReason is present, the Observation.component.valueQuantity **shall** be absent

– If the Observation.component.valueQuantity is present, the dataAbsentReason **shall** be absent

– One sub-entry of the compound may be in error and another not

– One sub-entry may be a vital sign and another not

– The order of any *code.coding element is MDC code first (required), LOINC second (if needed), and alternatives thereafter.

– See the section on the handling of special values and the handling of errors reported by the Measurement-Status attribute or measurement status from the complex observation attributes for a mapping from the error condition to the available data absent reason codes.

A.6.2.3 Enumeration OID Value

If the measurement value attribute is an Enum-Observed-Value-Simple-OID or an Enum-Observed-Value whose EnumVal element indicates that it is an OID, the measurement is a termCode.

A.6.2.3.1 Profile

The meta.profile entry **shall** contain "placeholder/phdCodedEnumerationObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.3.2 Encoding

The PHG **shall** encode the Enum measurement OID value as indicated in Table A-51.

Table A-51 – Enum Measurement OID Mapping

Observation Resource	Value	R,S,O, or Z
valueCodeableConcept.	**shall not** be present if error:	
coding.	If an alternative coding system is used, this element **shall** occur first	
code	*partition* = Type.*partition* *enumTermCode = Enum-Observed-Value-Simple-OID* *or* *enumTermCode* = Enum-Observed-Value-EnumVal.*enum-obj-id* *if Enum-Observed-Value-Partition* exists: { *partition = Enum-Observed-Value-Partition* } *then the code is set to:* *partition* * 2^{16} + *enumTermCode*	R
system	"urn:iso:std:iso:11073:10101"	R
display	*Reference identifier for code plus optional text*	S
dataAbsentReason.	**shall not** be present if no error:	
coding.		
code	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

Special notes and cautions:

– Do not confuse the Enum-Observed-Value.EnumVal.*metric-id* which describes the overall-type term code with the measurement Enum OID term code. The former would indicate something like 'meal context' and is handled in the section called Measurement Type. The latter indicates the measurement itself for example, 'breakfast', 'snack', 'lunch', 'dinner', etc.

– If the dataAbsentReason is present, the value[x] **shall** be absent

– If the valueCodeableConcept is present, the dataAbsentReason **shall** be absent

– See the section on the handling of errors reported by the *Measurement-Status* attribute for a mapping from the error condition to the available data absent reason codes.

A.6.2.4 Enumeration BITs Value

If the attribute is an *Enum-Observed-Value-Basic-Bit-Str* or an *Enum-Observed-Value-Simple-Bit-Str* or an Enum-Observed-Value whose EnumVal element indicates it is a BITs-string, the measurement is a 16-bit or 32-bit integer where each bit means something depending upon whether the bit is set or cleared. The meaning of each of these bits is defined in the specializations that use them.

A.6.2.4.1 Profile

The meta.profile entry **shall** contain "placeholder/phdBitsEnumerationObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.4.2 Encoding

There is no direct mapping of a BITs measurement into FHIR or CDAs. Therefore, the ASN.1 Vocabulary mapping is used to convert the bits to codes. Due to the possibility of multiple settings in a given measurement, each converted code that is to be reported in mapped to a component.

The mapping of each ASN.1 bit field to be reported consists of two steps, the first is to obtain the 32-bit nomenclature code using the procedure defined in the Measurement Type section. The bit position of the mapped setting is appended to the 32-bit nomenclature code separated by a dot (nomenclatureCode.bitPosition) which becomes the new code. This code is placed in the Observation.component.valueCodeableConcept element.

The vocabulary has been defined such that it can be created from protocol. The uploader **shall** create the converted 'ASN1code' as indicated in Table A-52.

Table A-52 – ASN.1 Bits Conversion Procedure

bitsValue = *Enum-Observed-Value-Basic-Bit-Str* or (16-bits) = *Enum-Observed-Value-Simple-Bit-Str* or (32-bits) = Enum-Observed-Value-EnumVal.*enum-bit-str (always 32 bit)*
nomenclatureCode = procedure from Measurement Type section
bitPosition = Mder Bit to be reported from 0 to 15/31
ASN1code = nomenclatureCode.bitPosition

Special notes and cautions:

– The Mder bit position is defined backwards, bit position 0 is the MOST significant bit of the 16- or 32-bit number.

– The ASN1code is NOT a decimal number but an alpha-numeric string.

Newer versions of the IEEE 11073-20601 specification define a bit mask which indicates which bit settings a given sensor device supports.

If there is no bit mask, the PHG

– **shall** map set, defined bits

– **may** map cleared bits,

– **shall not** map bits that are reserved or not defined by IEEE 11073-20601, a specialization, or a private encoding

The ASN.1 Enumeration BITs measurements **shall** be encoded as indicated in Table A-53 if there is no error.

Table A-53 – Enum Measurement ASN.1 BITs Mapping

Observation Resource	Value	R,S,O, or Z
value[x]	***shall not*** *be encoded*	X
dataAbsentReason	***shall not*** *be encoded (no appropriate reason exists)*	X
For each of the bit settings to be mapped:		
component.	These component elements **shall** occur before any additional component elements	
code.		
coding.		
code	*Set to the ASN1code computed from Table A-52*	R
system	"placeholder/fhir/IEEE.ASN1"	R
display	*ASN.1 name if known plus optional text*	O
valueCodeableConcept.		
coding.		
code	"y" *if set,* "n" *if cleared*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	*optional text*	O

If there is an error, the PHG **may** map the measurement. If the measurement is mapped, the ASN.1 Enumeration bits measurement **shall** be encoded as indicated in Table A-54.

Table A-54 – Enum Measurement ASN.1 BITs Mapping for Errors

Observation Resource	Value	R,S,O, or Z
value[x]	***shall not*** *be encoded*	R
dataAbsentReason.		
coding.		
code	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

Special notes and cautions:

– See the section on the handling of errors reported by the Measurement-Status attribute for a mapping from the error condition to the available data absent reason codes.

  If there is a bit mask (coming versions of 20601) the uploader has a means of determining which bits in the measurement are supported and defined. However, there is no way to distinguish between bits not defined in the specialization versus those defined but not supported by the sensor. Consequently, the same rules as shown in Table A-52 apply as for the case without a bit mask except

– The PHG **may** map a setting that is not supported by the device.

In this case, the bit setting is ignored and always mapped as "unsupported".

If the PHG maps an unsupported bit, setting the entry **shall** be encoded as indicated in Table A-55.

Table A-55 – Unsupported Enum Measurement ASN.1 BITs Mapping

Observation Resource	Value	R,S,O, or Z
value[x]	***shall not*** *be encoded*	R
dataAbsentReason	***shall not*** *be encoded (no appropriate reason exists)*	R
For each unsupported bit to be mapped:		
component.		
. code.		
. . coding.		
. . . *code*	*Set to the ASN1code computed from* Table A-52	R
. . . *system*	"placeholder/fhir/IEEE.ASN1"	R
. . . *display*	*ASN.1 name if known plus optional text*	O
. valueCodeableConcept	**shall not** be present	X
. dataAbsentReason.		
. . coding.		
. . . *code*	"unsupported"	R
. . . *system*	"http://hl7.org/fhir/data-absent-reason"	R
. . . *display*	*optional text*	O

A.6.2.5 Enumeration String Value

A.6.2.5.1 Profile

The meta.profile entry **shall** contain "placeholder/phdStringEnumerationObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.5.2 Encoding

If the attribute is an *Enum-Observed-Value-Simple-Str* or an Enum-Observed-Value whose EnumVal element indicates that it is a string, the measurement is human readable string. This attribute is used rarely in PHD devices but it is used in the cardiovascular specialization. The string maps to a valueString element.

The PHG **shall** map the Enumeration string value as indicated in Table A-56.

Table A-56 – Enum Measurement String Mapping

Observation Resource	Value	R,S,O, or Z
valueString.	**shall not** be present if error:	
. *value*	*value* = *Enum-Observed-Value-Simple-Str* or = Enum-Observed-Value-EnumVal.*enum-text-string*	
dataAbsentReason.	**shall not** be present if no error:	
. coding.		
. . *code*	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R

Observation Resource			Value	R,S,O, or Z
		system	"http://hl7.org/fhir/data-absent-reason"	R
		display	*Some text*	O

Special notes and cautions:

– If the dataAbsentReason is present, the value[x] element **shall** be absent

– See the section on the handling of errors reported by the Measurement-Status attribute for a mapping from the error condition to the available data absent reason codes.

A.6.2.6 RTSA sampled data Value

RTSA metrics distinguish themselves from the other two metric types in that it is not a single measurement but a sequence of measurements in time that occur at fixed intervals. The data is contained in the Simple-Sa-Observed-Value attribute but it is scaled. To decode the information, the information from the Sa-Specification, Scale-And-Range-Specification, and *Sample-Period* attributes is needed.

A.6.2.6.1 Profile

The meta.profile entry **shall** contain "placeholder/phdRtsaObservation".

NOTE – The actual profile URL is not yet specified.

A.6.2.6.2 Encoding

Periodic data is mapped to the SampledData data type in FHIR. The data element in this data type is also scaled. If $y[i]$ is the i^{th} entry of the actual unscaled data from the sensor, it is obtained from the SampledData type using the following relation:

$$y[i] = d[i] * s + b$$

where

s = SampledData.*scaleFactor*

b = SampledData.origin.*value*

$d[i]$ = SampledData.*data*$[i]$

In the case of the RTSA, the i^{th} entry of the actual unscaled data from the sensor is obtained from the Simple-Sa-Observed-Value attribute using the following relation:

$$y[i] = \frac{(A-B)x[i]}{I-J} + A - \frac{(A-B)I}{I-J}$$

where

A = Scale-and-Range-SpecificationX.*upper-absolute-value*

B = Scale-and-Range-SpecificationX.*lower-absolute-value*

I = Scale-and-Range-SpecificationX.*upper-scaled-value*

J = Scale-and-Range-SpecificationX.*lower-scaled-value*

Where X = 8, 16, or 32 and

$x[i]$ = Simple-Sa-Observed-Value.*data*$[i]$.

Since $\frac{(A-B)x[i]}{I-J} + A - \frac{(A-B)I}{I-J} = d[i] * s + b$,

$d[i] = x[i]$ if

s = SampledData.*scaleFactor* = $\frac{(A - B)}{I - J}$

and

b = SampledData.origin.*value* = $A - \frac{(A-B)I}{I-J} = \frac{(BI-AJ)}{I-J}$

allowing one to map the Simple-Sa-Observed-Value.*data*[i] values directly to the SampledData.*data*[i] values.

Table A-57 indicates the parameters required to perform the mapping of the RTSA data to FHIR SampledData.

Table A-57 – RTSA to Sampled Data Mapping Parameters

significantBits = Sa-Specification.SampleType.*significant-bits*
X = Sa-Specification.SampleType.*sample-size* *where* X = 8, 16, or 32 bits per sample
saObsVal[i] = Simple-Sa-Observed-Value.*data*[i]
if (*significantBits* < *sampleSize*) the most significant (*sampleSize* - *significantBits*) bits of *saObsVal*[i] are zeroed and *saObsVal*[i] are treated as unsigned integers if *significantBits* equals 255 *saObsVal*[i] are treated as signed integers else *saObsVal*[i] are treated as unsigned integers
A = Scale-and-Range-SpecificationX.*lower-absolute-value* as an Mder FLOAT
B = Scale-and-Range-SpecificationX.*upper-absolute-value* as an Mder FLOAT
I = Scale-and-Range-SpecificationX.*lower-scaled-value* as a X-bit integer
J = Scale-and-Range-SpecificationX.*upper-scaled-value* as a X-bit integer
scaleFactor = $\frac{(A - B)}{I - J}$
offset = $A - \frac{(A-B)I}{I-J} = \frac{(BI-AJ)}{I-J}$

Using the information from Table A-57, if the PHG chooses to map the RTSA measurements, the PHG **shall** map the data as indicated in Table A-58.

Table A-58 – Periodic RTSA Measurement Mapping

Observation Resource		Value	R,S,O, or Z
valueSampledData.		**shall not** be present if error:	
	period	Set to *Sample-Period* / 8 (time between samples in milliseconds)	Z
	factor	Set to *scaleFactor* from Table A-57 as a decimal given the precision of the FLOAT	Z
	dimensions	Set to Sa-Specification.*array-size* (number of samples)	R

Observation Resource	Value	R,S,O, or Z
data[*i*]	Set to *saObsVal*[*i*] from Table A-57 converted to strings separated by a space unsigned if *significantBits* from Table A-57 is 255 signed otherwise	R
origin.		
value	Set to *offset* from Table A-57 as a decimal given the precision of the FLOAT	Z
unit	UCUM code string corresponding to *Unit-Code*	S
system	"urn:iso:std:iso:11073:10101"	R
code	Set to $4*2^{16}$ + *Unit-Code*	R
dataAbsentReason.	**shall not** be present if no error:	
coding.		
code	*One of the FHIR defined* http://hl7.org/fhir/2017Jan/valueset-observation-valueabsentreason.html *codes.*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

Special notes and cautions:

– The 'Z' requirement means that the FHIR element is required to be populated but the value for the valueSampledData.*period*, valueSampledData.*factor*, *and* valueSampledData.origin.*value* mapping shown in the table is not. There are many possible ways the data can be scaled. All that is necessary is that the choice of these values is such that $y[i] = \frac{(A-B)x[i]}{I-J} + A - \frac{(A-B)I}{I-J} = d[i] * s + b$. As an alternative scaling one could un-scale the Simple-Sa-Observed-Value.*data*[*i*] values to get $y[i]$, set that to SampleData.*data*[*i*], and set $s = 1.0$ and $b = 0$. However, it is recommended to use the suggested scaling as it is likely that the sensor developers chose these scalings to most efficiently transfer the sensor waveforms. Regardless of the scaling chosen, these elements **shall** be populated

A.6.3 The Time Stamp

The time stamp is obtained from one of the metric time stamp attributes or from the time of reception of the measurement if the sensor sends no time stamp with the measurement. The information in the coincident time stamp Observation is used to correct the time stamps such that they are on the same time line as the PHG if the PHG is better synchronized to NTP than the sensor. The coincident time stamp Observation is also used to convert sensor relative times to the PHG time line.

A.6.3.1 Coincident Time Stamp reference

If there is a time stamp reported by the sensor, there shall be an Observation.related.*target* element pointing to the appropriate Coincident Time Stamp Observation.

A.6.3.2 Absolute, Base Offset, or No Time Stamp

If there is an Absolute-Time-Stamp or Base-Offset-Time-Stamp attribute the time will need to be corrected if correction is needed as indicated by the coincident time stamp Observation resource. All absolute time stamps are assumed to be in the current time zone of the PHG. If there is no time stamp, the time stamp is the time of the reception of the measurement by the PHG.

The FHIR dateTime data type primitive uses the following format:

YYYY-MM-DDTHH:MM:SS[.sss]+/-ZZ:zz where

a) the hour HH is in 24 hour format and

b) the time zone **shall** be included.

The CDG adds the requirement that fractional seconds **shall** be included if the measurement has fractional seconds.

The PHG **shall** map the non-relative time stamps as indicated in Table A-59.

Table A-59 – Absolute, Base Offset, and PHG-Received Time Stamp Mapping

Observation Resource		Value	R,S,O, or Z
If there is no Measure-Active-Period attribute:			
effectiveDateTime		*if* Absolute-Time-Stamp: *finalTimeStamp* = Absolute-Time-Stamp *encoded as* YYYY-MM-DDTHH:MM:SS[.ss] +/-ZZ:zz *where* +/-ZZ:zz *is the offset from local time to UTC obtained from the PHG* *and [.ss]* ***shall*** *be excluded if the time stamp has no fractional seconds or the time resolution is >= to one second.* *if* Base-Offset-Time-Stamp: *finalTimeStamp* = Base-Offset-Time-Stamp *encoded as* YYYY-MM-DDTHH:MM:SS[.sss] +/-ZZ:zz *where* *[.sss]* ***shall*** *be excluded if the time resolution is >= to one second.* *if no time stamp*: *finalTimeStamp = PHG time of reception encoded as* YYYY-MM-DDTHH:MM:SS.sss +/-ZZ:zz *effectiveDateTime = finalTimeStamp*	R
If there is a Measure-Active-Period attribute:			
effectivePeriod.			
	start	*if* Absolute-Time-Stamp: *finalTimeStamp* = Absolute-Time-Stamp *encoded as* YYYY-MM-DDTHH:MM:SS[.ss] +/-ZZ:zz *where* +/-ZZ:zz *is the offset from local time to UTC obtained from the PHG* *and [.ss]* ***shall*** *be excluded if the time stamp has no fractional seconds or the time resolution is >= to one second.* *if* Base-Offset-Time-Stamp: *finalTimeStamp* = Base-Offset-Time-Stamp *encoded as* YYYY-MM-DDTHH:MM:SS[.sss] +/-ZZ:zz *where* *[.sss]* ***shall*** *be excluded if the time resolution is >= to one second.* *start = finalTimeStamp*	R
	end	*end = finalTimeStamp + Measure-Active-Period with precision indicated by the Mder FLOAT, encoded as a FHIR dateTime data type.*	R

Special notes and cautions:

– Bluetooth Low Energy PHDs do not report time stamps at a finer resolution than one second so fractional seconds are never reported.

– IEEE 11073-20601 PHDs using absolute time always report fractional seconds as it is part of the wire format, but the MdsTimeInfo.*time-resolution-abs-time* may indicate the actual resolution. If MdsTimeInfo.*time-resolution-abs-time* indicates unknown, fractional seconds are included.

– FHIR requires all dateTime primitives to include an offset to UTC but a PHD supporting absolute time has no knowledge of time zone offsets. To be FHIR compliant, the PHG has to add this information. The offset to UTC from the current time zone is used since it is the most-likely correct for most of the PHD measurements.

– The *Measure-Active-Period* reports the duration in units of seconds as an Mder FLOAT.

– When the measurement has a duration the Observation.effective[x] becomes Observation.effectivePeriod otherwise it is Observation.effectiveDateTime.

Example with no *Measure-Active-Period*:

```
"effectiveDateTime":"2015-07-10T18:48:12.681-04:00"
```

Example with *Measure-Active-Period*:

```
"effectivePeriod"{
    "start":"2015-07-10T18:48:12.681-04:00",
    "end":"2015-07-10T18:58:12.681-04:00",
}
```

A.6.3.3 Relative Time Stamp

If the time stamp is one of the relative time stamps the PHG obtains the *relativeTime* value from the attribute and scales it if needed to microseconds. The PHG needs to map this time to wall clock time before it can convert it to FHIR format and map it to the Observation resource. If there is a *Measurement-Active-Period* attribute the end time of the measurement needs to be computed. The algorithm shown in Table A-60 shows how to obtain the needed wall clock times.

Table A-60 – Relative Time to Wall Clock Measurement Time

currentRelativeTime = the PHD relative time from the coincident time stamp Observation
relativeTime = *Relative-Time-Stamp * 125 (scale to microseconds) or* *= HiRes-Time-Stamp*
currentPHGTime = the PHG current time from the coincident time stamp Observation
duration = 0 *if* Measurement-Active-Time attribute exists: *duration* = *Measurement-Active-Time* where the Mder FLOAT is converted to microseconds in the precision indicated by the Mder encoding.
wallclockMeasurementTime = (*relativeTime – currentRelativeTime*) + *currentPHGTime*
wallclockDurationTime = (*duration* + *relativeTime – currentRelativeTime*) + *currentPHGTime*

The PHG **shall** map the relative time stamps as indicated in Table A-61.

Table A-61 – Relative Time Stamp Mapping

Observation Resource		Value	R,S,O, or Z
If there is no *Measure-Active-Period* attribute:			
effectiveDateTime		*effectiveDateTime = wallclockMeasurementTime* from Table A-60	R
If there is a *Measure-Active-Period* attribute:			
effectivePeriod.			
	start	*start = wallclockMeasurementTime* from Table A-60	R
	end	*end = wallclockDurationTime* from Table A-60	R

Special notes and cautions:

– All relative times are assumed to be in the current time zone of the PHG thus the offset is always taken from the current offset of the PHG.

– Different FHIR data types are needed for the Observation.effective[x] when the measurement has a duration.

A.6.4 Measurement Status and Numerical Special Values

Errors are reported using the *Measurement-Status* attribute and the special values in numeric measurements. The complex Nu-Observed-Value and Enum-Observed-Value report their own measurement status in every measurement and those values override the *Measurement-Status* attribute.

The PHG **shall** report errors in the FHIR Observation resource using the Observation.dataAbsentReason and Observation.status element. The available dataAbsentReason codes for errors are listed in Table A-62.

Table A-62 – Measurement Status and Numeric Special Values

Code	Display	Definition
unknown	Unknown	The value is not known.
asked	Asked	The source human does not know the value.
temp	Temp	There is reason to expect (from the workflow) that the value may become known.
not-asked	Not Asked	The workflow did not lead to this value being known.
masked	Masked	The information is not available due to security, privacy or related reasons.
unsupported	Unsupported	The source system was not capable of supporting this element.
astext	As Text	The content of the data is represented in the resource narrative.
error	Error	Some system or workflow process error means that the information is not available.
NaN	Not a Number	NaN, standing for not a number, is a numeric data type value representing an undefined or un-representable value.
not-performed	Not Preformed	The value is not available because the observation procedure (test, etc.) was not performed.
PINF	Positive Infinity	Positive Infinity is used to represent numerical overflow or positive overflow errors and in some cases sensor overflows. This error code has been added after the 3.0.1 FHIR release for V2 compatibility.

Code	Display	Definition
NINF	Not a Number	Negative Infinity is used to represent numerical underflow or negative overflow errors and in some cases sensor underflows. This error code has been added after the 3.0.1 FHIR release for V2 compatibility.

Not all the conditions in the table are relevant to device measurements.

The errors reported in the Measurement-Status attribute or in the Enum-Observed-Value.*status*, Nu-Observed-Value.*status*, or Compound-Nu-Observed-Value.*status* elements are encoded as a 16-bit ASN1 BITs field as follows:

```
MeasurementStatus ::= BITS-16 {
     invalid(0),
     questionable(1),
     not-available(2),
     calibration-ongoing(3),
     test-data(4),
     demo-data(5),
     validated-data(8), -- relevant, e.g., in an archive
     early-indication(9),    -- early estimate of value
     msmt-ongoing(10)   -- indicates a new observation is just being taken
(episodic)
}
```

The Pulse Ox and Continuous Glucose Monitor specializations define two additional bit settings:

```
msmt-state-in-alarm(14) -- indicates that the observation is outside
threshold boundaries
```

and

```
msmt-state-al-inhibited(15)  -- indicates that the threshold indication is
disabled.
```

The Observation.*status* element has a 'preliminary' code which can be used to report the 'early-indication' option.

In addition to status errors numeric measurements may indicate special values such as NaN (not a number), Pinf (positive infinity), Ninf (negative infinity), among others.

The errors from numeric measurements or status reports are mapped to the FHIR Observation resource as shown in Table A-63.

Table A-63 – Error Encoding

Status field	Mapping
invalid	*.dataAbsentReason = error
questionable	value[x] reported unless special value
not-available	*.dataAbsentReason = unknown
calibration-ongoing	value[x] reported unless special value
test-data	value[x] reported unless special value
demo-data	value[x] reported unless special value
validated-data	value[x] reported unless special value
early-indication	Observation.status = preliminary value[x] reported unless special value
msmt-ongoing	value[x] reported unless special value

Status field	Mapping
Numeric Special Value	
NaN	*.dataAbsentReason = NaN
+INF	*.dataAbsentReason = PINF
-INF	*.dataAbsentReason = NINF
all else	*.dataAbsentReason = error

The PHG shall report status errors or special value errors as indicated in Table A-64.

Table A-64 – Measurement Status and Special Value Error Handling

Observation Resource	Value	R,S,O, or Z
status	"preliminary" *if indicated by the mapping in Table A-63 else* "final"	
value[x]	**shall not** be present if dataAbsentReason is present	
dataAbsentReason.	**shall not** be present if value[x] is present	
coding.		
code	*One of the mapped codes in Table A-63*	R
system	"http://hl7.org/fhir/data-absent-reason"	R
display	*Some text*	O

A.6.5 Source Handle Reference or Source Handle Reference List

When this attribute appears in a measurement it points to another metric object which is related in some way to the current measurement. A measurement containing a *Source-Handle-Reference* or Source-Handle-Reference-List attribute can only be received after the referenced measurement has been sent by the sensor.

The source handle references use the handle of the metric object as its pointer. The handle has no semantic meaning on its own. If several measurements have been received that use this handle, a source handle reference handle points to the most recently received measurement on that handle.

To maintain the semantic meaning of the source handle references in FHIR, one needs to point to the most recently sent or created Observation resource(s) containing the mapped metric measurements to which the current measurement is pointing. In FHIR URLs containing the logical id are used to reference one resource from another. In this case the URL would be the location on the server of the Observation being pointed to by the source handle references. If the Observation resource is being sent in a Bundle and the Observations being pointed to are also in the Bundle, the references would be pointing to the temporary logical ids.

The PHG **shall not** map these attributes if the measurement is in error; for example if the measurement itself is a NaN.

Otherwise the PHG **shall** map the Source-Handle-Reference or Source-Handle-Reference-List to its own Observation.related.*target* element as indicated in Table A-65.

Table A-65 – Source Handle Reference Mapping

Observation Resource	Value	R,S,O, or Z
For each handle contained in the attribute (1 for the Source-Handle-Reference, N for the list)		
related.		
target	*URL to Observation being referenced*	R

These entries are placed after the Coincident Timestamp Observation reference, if any, and before any application-included references, if any.

A.6.6 Component Elements

The remaining attributes in the Metric Object are encoded in component elements of the Observation. Component elements are sets of additional information which further describe the primary observation. Likewise, the remaining attributes represent additional information about the primary metric measurement.

All these support attributes are mapped into components in the same manner. The attribute code is mapped into the Observation.component.code and the attribute value is mapped into the Observation.component.value[x] element depending upon whether the attribute values are a Quantity, code, or string.

A.6.6.1 Supplemental Types

This attribute contains a list of partition:term code value pairs and is thus a CodeableConcept type of value in FHIR. Each entry in the list is mapped to its own component element.

The PHG **shall not** map this attribute if measurement is in error; for example if the measurement itself is a NaN.

Otherwise the PHG **shall** map the supplemental types attribute as indicated in Table A-66.

Table A-66 – Supplemental Types Mapping

Observation Resource	Value	R,S,O, or Z
For each of the supplemental types *n*:		
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	68193	R
system	"urn:iso:std:iso:11073:10101"	R
display	*"*MDC_ATTR_SUPPLEMENTAL_TYPES" *plus any optional text*	S
valueCodeableConcept.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	Supplemental-Types[*n*].*partition* * 2^{16} + Supplemental-Types[*n*].*termCode*	R
system	"urn:iso:std:iso:11073:10101"	R
display	*Reference id corresponding to the code plus optional text*	S

A.6.6.2 Measurement Active Period

This attribute value is incorporated into the time stamp.

A.6.6.3 Accuracy

The *Accuracy* attribute defines the maximum deviation of the actual observation from the sent observation. It applies only to numerics. The deviation is in the units of the observation. The accuracy is an absolute value and is encoded as an Mder FLOAT.

The PHG **shall not** map the *Accuracy* attribute value if it is a special value or no *.valueQuantity entry to which it refers exists.

Otherwise, the PHG **shall** map the *Accuracy* attribute value to an Observation.component element as indicated in Table A-67.

Table A-67 – Accuracy Attribute Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	67914	R
system	"urn:iso:std:iso:11073:10101"	R
display	*"*MDC_ATTR_NU_ACCUR_MSMT" *plus any optional text*	S
valueQuantity.	**shall not** be present if a special value or an error:	
value	*Accuracy* *decoded from the FLOAT with the precision given by the respective Mder encoding*	R
units	*UCUM string for the unit code*	S
system	"urn:iso:std:iso:11073:10101"	R
code	*Observation.valueQuantity.code or in the compound case the unit code associated with the compound value. See the special notes and cautions.*	R

Special notes and cautions:

– The unit code is to be in the same units as the measurement value. The IEEE 11073-20601 specification has a Compound-Nu-Observed-Value where each element of the compound can have different units but there are no restrictions on the use of the Accuracy attribute with a Compound-Nu-Observed-Value. It is assumed the Accuracy attribute cannot be present unless the units' value is the same for all Compound-Nu-Observed-Value components.

A.6.6.4 Relative Time Stamp

This attribute gives the relative time in units of 1/8 ms.

The PHG **shall** map this attribute to an Observation.component as indicated in Table A-68.

Table A-68 – Relative Time Stamp Attribute Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	67985	R
system	"urn:iso:std:iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_STAMP_REL" *plus any optional text*	S
valueQuantity.		
value	*Relative-Time-Stamp* * 125	R
units	"us" *UCUM code for microseconds*	R
system	"urn:iso:std:iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds).*	R

A.6.6.5 Hi Res Relative Time Stamp

This attribute gives the relative time in units of microseconds.

The PHG **shall** map this attribute to an Observation.component as indicated in Table A-69.

Table A-69 – HiRes Relative Time Stamp Attribute Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	68073	R
system	"urn:iso:std:iso:11073:10101"	R
display	*"*MDC_ATTR_TIME_STAMP_REL_HI_RES" *plus any optional text*	S
valueQuantity.		
value	*HiRes-Time-Stamp*	R
units	"us" *UCUM code for microseconds*	R
system	"urn:iso:std:iso:11073:10101"	R
code	264339 *(MDC 32-bit code for microseconds).*	R

A.6.6.6 Label String

The *Label String* attribute is an arbitrary human-readable text string describing more about the measurement type. This string **may** be placed in the Observation.text element. It would be advantageous if it were noted that the string originated from the PHD.

A.6.6.7 Unit Label String

The *Unit Label String* attribute is an arbitrary human-readable text string describing the unit code.

This string **may** be placed in the Observation.text element. It would be advantageous if it were noted that the string originated from the PHD.

A.6.6.8 Specialization Specific Metric Attributes

The specialization-specific Metric Attributes are encoded as Observation.component elements.

A.6.6.8.1 Pulse Ox

The pulse oximeter specialization defines three specialization-specific attributes for the purpose of establishing thresholds and alerts regarding those thresholds. The primary attribute is the Current-Limits attribute which specifies the low and high thresholds for a given measurement. Without this attribute, the Alert-Op-State and Alert-Op-Text-String attributes do not make sense. If these attributes are present without the Current-Limits attribute, their values **shall not** be mapped.

A.6.6.8.1.1 Alert-Op-State Attribute

This attribute indicates whether the thresholds defined in the Current-Limits attribute are on or off. This attribute should not be present if the Current-Limits attribute is not present. The attribute value is encoded as a 16-bit ASN1 BITs field. Table A-70 gives their ASN.1 vocabulary encodings.

Table A-70 – Alert-Op-State ASN.1 BITS Mapping

Code	ASN.1 name	Description
68746.0	lim-alert-off	When set, all alerts set in the current limits are off
68746.1	lim-low-off	When set, the low limit set in the current limits is off
68746.2	lim-high-off	When set, the high limit set in the current limits is off

Note that the settings are encoded in the negative, so a cleared bit means the alert is ON.

The PHG **shall not** map this attribute if the Current-Limits attribute is not present.

The PHG **shall not** map any undefined setting.

The PHG **shall** map each set value.

The PHG **may** map a cleared value.

For those bits that are mapped the PHG **shall** map each case to its own Observation.component element as indicated in Table A-71.

Table A-71 – Alert-Op-State Mapping

Observation Resource	Value	R,S,O, or Z
Value		
For each bit to be mapped		
code.		
coding.		
code	*The code from Table A-70 (will be 68746.x*)	R
system	"placeholder/fhir/IEEE.ASN1" *(placeholder)*	R
display	*ASN.1 name from Table A-70 plus optional text*	S
valueCodeableConcept.		

Observation Resource	Value	R,S,O, or Z
coding.		
code	"y" *if set,* "n" *if cleared*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	*optional text*	O

A.6.6.8.1.2 Current-Limits Attribute

This attribute has the lower followed by upper threshold limits for the said observation as an Mder FLOAT. This attribute describes a range and its values are mapped to an Observation.component.valueRange element.

The PHG **shall** map the attribute to an Observation.component element as indicated in Table A-72.

Table A-72 – Current-Limits Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	67892	R
system	"urn:iso:std:iso:11073:10101"	R
display	"MDC_ATTR_LIMIT_CURR" *plus optional text*	S
valueRange.		
low.		
value	Current-Limits.*lower coded to a FHIR decimal with the precision of the Mder FLOAT*	R
unit	*UCUM code of the Unit-Code attribute*	S
system	"urn:iso:std:iso:11073:10101"	R
code	$4*2^{16}$ + *Unit-Code*	R
high.		
value	Current-Limits.*upper coded to a FHIR decimal with the precision of the Mder FLOAT*	R
unit	*UCUM code of the Unit-Code attribute*	S
system	"urn:iso:std:iso:11073:10101"	R
code	$4*2^{16}$ + *Unit-Code*	R

A.6.6.8.1.3 Alert-Op-Text-String Attribute

The Alert-Op-Text-String attribute has arbitrary text strings representing the lower followed by upper threshold limits in the Current-Limits attribute. It is informational only and not machine readable. This attribute should not be present if the Current-Limit attribute is not present.

The PHG **shall not** map this attribute as component if the Current-Limits attribute is not present.

The PHG **may** also choose to add this information to the overall text element description for the Observation resource.

Given the above, the PHG **may** map this attribute as a component.

If the PHG maps this attribute it **shall** be mapped to a single Observation.component element as indicated in Table A-73.

Table A-73 – Alert-Op-Text Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	68104	R
system	"urn:iso:std:iso:11073:10101"	R
display	"MDC_ATTR_AL_OP_TEXT_STRING" *plus optional text*	S
valueString	*The information from the* Alert-Op-Text.*lower_text and* Alert-Op-Text.*upper_text presented in an application dependent manner.*	R

A.6.6.8.2 Medication Monitor

A.6.6.8.2.1 Context-Key Attribute

The *Context-Key* attribute is an EUI-64 context identifier that by itself means nothing but it can be used to identify a context specification for the given medication monitor, for example to determine what the medication is and what the feedback questions and answers are. The attribute value is mapped to a component.valueCodeableConcept element as a 16-digit hexadecimal code.

The PHG **shall** map the attribute as indicated in Table A-74.

Table A-74 – Context-Key Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	68216	R
system	"urn:iso:std:iso:11073:10101"	R
display	"MDC_ATTR_CONTEXT_KEY" *plus optional text*	S
valueCodeableConcept.		
coding.		
code	*Context-Key as a 16-digit Hexadecimal string*	R
system	"urn:oid:1.2.840.10004.1.1.1.0.0.1.0.0.1.2680"	R
display	*optional text*	O

A.6.6.8.3 Continuous Glucose Monitor

The continuous glucose monitor has two additional specialization-specific attributes and defines two extra bit settings in the *Measurement-Status* attribute. The Measurement-Confidence-95 attribute specifies a range over which the manufacturer is 95% sure that the actual value lies within the range specified by *lower <= measured <= higher*. The two extra bits settings in the *Measurement-Status* attribute and the Threshold-Notification-String attribute are used report information about any threshold. The Threshold-Notification-String is a human readable text string and is not generically processable electronically.

A.6.6.8.3.1 Measurement-Confidence-95

The Measurement-Confidence-95 attribute contains a lower bound followed by an upper bound. The values are encoded as an Mder SFLOAT.

The PHG **shall** encode the attribute into an Observation.component as indicated in Table A-75.

Table A-75 – Measurement-Confidence-95 Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	526988	R
system	"urn:iso:std:iso:11073:10101"	R
display	"MDC_ATTR_MSMT_CONFIDENCE_95" *plus optional text*	S
valueRange.		
low.		
value	Measurement-Confidence-95.*lower_bound coded to a FHIR decimal with the precision of the Mder SFLOAT*	R
unit	*UCUM code of the Unit-Code attribute*	S
system	"urn:iso:std:iso:11073:10101"	R
code	$4*2^{16}$ + *Unit-Code*	R
high.		
value	Measurement-Confidence-95.*upper_bound coded to a FHIR decimal with the precision of the Mder SFLOAT*	R
unit	*UCUM code of the Unit-Code attribute*	S
system	"urn:iso:std:iso:11073:10101"	R
code	$4*2^{16}$ + *Unit-Code*	R

A.6.6.8.3.2 Threshold-Notification-Text-String

The *Threshold-Notification-Text-String* attribute has as a human readable string related to the any of the possible current threshold notifications. It is not for machine interpretation.

The PHG **may** map this attribute as a component element.

The PHG **may** place the contents in the text element of the Observation.

If the PHG chooses to map this attribute as a component element, it **shall** map this attribute to an Observation.component as indicated in Table A-76.

Table A-76 – Threshold-Notification-Text-String Mapping

Observation Resource	Value	R,S,O, or Z
component.		
code.		
coding.	If an alternative coding system is used, this element **shall** occur first	
code	68232	R
system	"urn:iso:std:iso:11073:10101"	R
display	"MDC_ATTR_THRES_NOTIF_TEXT_STRING" *plus optional text*	S
valueString	*Threshold-Notification-Text-String*	R

A.6.6.8.3.3 Measurement-Status alert bits

The alert bits use Mder bits 14 and 15 of the Measurement-Status. Table A-77 gives their ASN1 vocabulary encodings.

Table A-77 – Measurement-Status Alert Bits

Code	ASN.1 name	Description
133447.14	msmt-state-in-alarm	When set, indicates that the measurement is outside threshold boundaries
133447.15	msmt-state-al-inhibited	When set, indicates that the threshold indication is disabled.

The PHG **shall** map each set value.

The PHG **may** map each cleared value.

If the PHG maps a case, it **shall** map each case to its own Observation.component element as indicated in Table A-78.

Table A-78 – Measurement Status Alert Mapping

Observation Resource	Value	R,S,O, or Z
Value		
For each bit to be mapped		
code.		
coding.		
code	*The code from Table A-77 (either 133447.14 or 133447.15)*	R
system	"placeholder/fhir/IEEE.ASN1" *(placeholder)*	R
display	*ASN.1 name from Table A-77 plus optional text*	S
valueCodeableConcept.		
coding.		
code	"y" *if set,* "n" *if cleared*	R
system	"http://hl7.org/fhir/v2/0136"	R
display	*optional text*	O

A.6.7 The Observation Identifier

The Observation.identifier.value provides the search string for the conditional create. It is constructed from the Patient.identifier, DeviceComponent.identifier, Observation.code, value, and unmodified time stamp from the sensor. If there is a Supplemental-Types attribute, that list of codes is needed as well. The use of the unmodified sensor time stamp is important because the PHG may modify the time stamp differently if the PHG is re-synchronized and its timeline changes.

The identifier is to provide a string that uniquely identifies a given measurement for a given patient from a given sensor device.

The Observation.identifier.value **shall** be set (the dashes are part of the identifier) as follows:

– Patient.identifier.*value*-Patient.identifier.*system*-

DeviceComponent.identifier.*value*-
Observation.code.coding.*code*-
Observation.value[x].*value*-
reported_sensor_timestamp-
Observation.component*(supplemental_types)*.code.coding.*code (32-bit values separated by dashes)*

where

– Patient is the Patient resource referenced by this Observation resource,

– DeviceComponent is the top-level DeviceComponent resource referenced directly or indirectly by the Observation resource,

– *reported_sensor_timestamp* is absent if the sensor does not report a time stamp,

– *reported_sensor_timestamp* is the time stamp as reported by the sensor as follows:

  - If the sensor reports absolute time this string will be encoded as an HL7 DTM YYYYMMDDTHHMMSS.ss.

    Example 1: The sensor reports absolute time as an 8-byte Binary Coded Decimal (BCD) Mder OCTET STRING:

    0x20 0x07 0x02 0x01 0x12 0x05 0x20 0x86 which is the date and then the time.

    reported_sensor_timestamp = 20070201120520.86

    Example 2: The BTLE sensor reports absolute time as a 7-byte string with no hundredths and it is not BCD or Mder. The year is the first two bytes in little endian thus 0x7E0 = 2016:

    0xE0, 0x07, 0x05, 0x17, 0x11, 0x34, 0x11

    reported_sensor_timestamp = 20160523T175217.00

  - If the sensor reports base-offset time this string will be encoded as seconds.fractional_seconds.offset

    Example: The sensor reports base offset time as an 8-byte string. The first four bytes are the seconds since 1900/01/01 00:00:00 (not the Unix Epoch!), the next two bytes the fractional seconds in units of $1/65536^{th}$ of a second and the last two bytes are the offset in minutes. Only the offset is signed.

    Example:

    8-byte Mder OCTET STRING: 0xD4 0x67 0x40 0x38 0x13 0x14 0xFE 0xD4

    seconds are 0xD4674038 = 3563536440

    fractional seconds 0x1314 = 4884

offset -300

reported_sensor_timestamp = 3563536440.4884.-300

if offset is positive: *reported_sensor_timestamp* = 3563536440.4884.+300

- If the sensor reports relative time the string will be the reported value in units of $1/8^{th}$ seconds as a decimal unsigned integer with no leading zeros. On the wire this value is an Mder sequence of 4 bytes with the MSB to the left.
- If the sensor reports high resolution relative time, the string will be the number of microseconds as an unsigned integer with no leading zeros. On the wire this value is an Mder sequence of 8 bytes with the MSB to the left.

– Observation.code.coding.*code* is the required 32-bit MDC code entry,

– Observation.value[x].*value* depends upon the value type as follows:

- If Observation.value[] exists *value* is one of:
  - Observation.valueQuantity.*value*
  - Observation.valueCodeableConcept.coding.*code*
  - Observation.*valueString*
  - Observation.valueSampledData.*period*-Observation.valueSampledData.*dimensions*-Observation.valueSampledData.*data*[0]
- If Observation.dataAbsentReason exists
  - value = Observation.dataAbsentReason.coding.code
- If neither Observation.value[] or Observation.dataAbsentReason exists one has a compound measurement or a mapped ASN1 BITs measurement
  - If compound the Observation.component.valueQuantity.*value* from each compound entry is used
    - *value* = entry1-entry2- …entryN
  - If an ASN1 BITs string the 16-bit or 32-bit value as an unsigned integer is used
    - *value* = bits value received from the sensor device enumeration

Note that this string is not designed for convenient interpretation.

A.6.8 FHIR-Specific Entries

This section describes the FHIR-specific elements required by these guidelines. Some of these are required by FHIR and have fixed values, such as the resourceType which is "Observation".

– The Observation.*status* is required by FHIR. It **shall** be set to 'final' unless a measurement status value has been received indicating 'early indication' in which case the Observation.*status* is set to 'preliminary'.

– The Observation.*subject* **shall** point to the Patient resource identifying the patient the measurement was taken on.

– The Observation.*device* **shall** point to the top-level DeviceComponent resource that identifies the PHD that took the measurement if the System-Type-Spec-List contains a single entry. If there are multiple entries in the System-Type-Spec-List, the Observation.*device* **shall** point to the DeviceComponent containing the specialization or sub profile from which the measurement came. If the PHG does not know which specialization or sub-profile the

measurement came from, the Observation.*device* element **shall** point to the top-level DeviceComponent.

– The Observation.related.*target* **shall** point to the coincident time stamp Observation associated with this measurement if the measurement had a sensor-provided time stamp.

  - If there are multiple Observation.related.*target* entries, the entries shall be ordered as follows:
    - Coincident Timestamp Observation reference if any
    - Source-Handle-Reference Observation references if any
    - Application references if any

A.6.9 Special Situations

A.6.9.1 Time Discontinuities: Date-Time-Adjustments

The reported timeline for the measurement **shall** remain unbroken. If there is a discontinuity in the timeline, a coincident time stamp Observation **shall** be reported by the PHG that reflects the adjustment. Continua compliant sensors using [20601] indicate the adjustment in time by the presence of a Date-Time-Adjustment attribute in scan event reports (live data) or a PM-Segment Date-Time-Adjustment in stored data. In either case, the PHG creates a coincident time stamp Observation for the observation data affected by the adjustment.

A positive adjustment value of *t* seconds indicates that the current time *at the time of the adjustment* has been advanced by *t* seconds.

Live Measurements

If the adjustment value is received as a Date-Time-Adjustment in a scan event report, the adjustment is happening at that time. The Observation.valueDateTime of the current coincident time stamp reported by the PHG **shall** be adjusted by the adjustment value to create the Observation.valueDateTime for the new coincident time stamp Observation. All subsequent measurements received in scan event reports **shall** point to the new coincident time stamp Observation. The new coincident time stamp Observation becomes the current coincident time stamp Observation.

For example, if the current coincident time stamp Observation has an Observation.valueDateTime of 2017-01-14T17:30:41-05:00 and the Observation.effectiveDateTime is 2017-01-14T18:30:41-05:00 and the Date-Time-Adjustment is one hour, the new coincident time stamp Observation would have the Observation.valueDateTime set to 2017-01-14T18:30:41-05:00. The Observation. effectiveDateTime does not change.

Stored Measurements

If the adjustment value is received in a PM-Segment Date-Time-Adjustment attribute, the sensor's clock is *already* on the new timeline. Thus, the coincident time stamp Observation created by the PHG using the current time of the sensor includes the adjustment value. The data in the PM Segment, however, is on the old time line and does not include the adjustment value. In this case the PHG **shall** generate a coincident time stamp Observation that represents the sensor's current time without the adjustment value, as this reflects the sensors clock at the time the measurements were made. The PHG **shall** subtract the adjustment value from the existing coincident time stamp Observation. valueDateTime and those measurements will refer to the modified value in the new coincident time stamp Observation.

All measurements from the PM segment affected by the PM-Segment Date-Time-Adjustment attribute **shall** point to this modified coincident time stamp Observation.

For example, if the Observation.valueDateTime value in the existing coincident time stamp Observation is 2014-05-10T09:22:32-05:00 and the date-time-adjustment value in the PM-Segment Date-Time-Adjustment attribute is 72 minutes, that indicates the current clock is 72 minutes ahead of what it was prior to the change. Thus, one subtracts the 72 minutes from the current Observation.valueDateTime to get the time the clock would have had if no time change were made, making the Observation.valueDateTime 2014-05-10T09:21:20-05:00.

A.6.9.2 Observations with and without Timestamps

ISO/IEEE 11073-20601 allows a sensor to send observations with and without timestamps in the same association. An example would be a pulse ox sending specialization configuration 0x190 observations ('streamed' SpO_2 and Pulse Rate values with no time stamp) and occasional SPOT observations from specialization configuration 0x191 with time stamps in an extended configuration. Observations with timestamps **shall** point to a coincident time stamp Observation. Observations without timestamps **shall not** point to a coincident time stamp.

Appendix I

FHIR Background

(This appendix does not form an integral part of this document.)

This appendix gives some background on FHIR and a discussion of some of the issues one faces when mapping from IEEE 11073-20601 to FHIR. It is assumed that the reader is familiar with the basics of the CDG, PCD-01, and IEEE 11073-20601. From the point of view of this discussion, the new feature is FHIR.

I.1 FHIR Message Payload

Before the introduction of the FHIR standard, HL7 had two major approaches for the representation of health care data; V2 (version 2) messages and V3 (version 3) documents. V2 messages consist of segments where each segment represents some general concept like a patient, an order, an observation, etc., and each segment consists of fields that describe aspects of the segment. The fields consist of V2 data types which can consist of primitives (like an integer, string, url, etc.) or additional data types. These data types form the building blocks of all V2 messages. V2 messages are usually expressed in EDI format (the familiar OR-bars used in PCD-01) but may also be expressed in XML. V3 documents consist of headers and sections where the headers contain all the non-clinical content and the sections contain the clinical content. The sections contain entries that represent clinical concepts like observations, allergies, medications, procedures, etc. and are named accordingly. The headers also contain sub-elements that have no formal group designation like 'entries' of a section but are identified only by their name, such as the custodian, author, recipient, recordTarget, etc. The header sub-elements and section entries then consist of elements that describe the components of the given entry. V3 elements eventually reduce to V3 data types. V3 headers, header sub-elements, sections, and entries are all identified by a unique OID called a template. V3 documents are always expressed as XML.

The common feature of V2 messages and V3 documents is that they describe a complete health care event. The distinguishing feature of the FHIR standard is that it breaks down health care events into basic units called resources that are linked together by references. It would take several resources to semantically represent the content of a V2 message or V3 document.

Defining these resources and their content is the major challenge in the development of the FHIR standard. Given the complexity of healthcare, it is impossible to create resources that satisfy all the use cases 100% of the time and still maintain the simplicity desired. In that end FHIR uses the 80% rule; the elements defined in the resources are those which are common across 80% of the use cases. Trying to satisfy every use case 100% of the time would likely require at least one element addition for each use case, significantly increasing the number of elements in a resource. Thus, to handle the remaining 20%, FHIR provides extensions. Extensions are much like private attributes in IEEE 11073-20601; they only mean something to the definer of the extension. On the other hand, FHIR defines a separate 'profiles' standard which allows one to exchange the information necessary for other parties to interpret these extensions. The extensions with their descriptions can be registered so they are accessible to the community.

For those familiar with V3 CDA documents its clear to see that many of the FHIR resources, both in name and content, come from the CDA entry templates. A list of the currently defined resources (STU3 3.0.1) is available at http://hl7.org/fhir/resourcelist.html.

If one examines the list of FHIR resources one will see there are about 110 of them. That number may grow. Fortunately, only three of these are needed for the CDG mapping. They are the Patient, DeviceComponent, and Observation resources. For the mapping of sensor and PHG information and the measurements generated by sensors, only the DeviceComponent and Observation resources are

needed. PHD sensor devices do not provide patient information by protocol sufficient to populate a Patient resource. Patient information for the Patient resource needs to be obtained out-of-band (by some other means than the IEEE 11073-20601 or BTLE exchange protocols).

I.1.1 FHIR Fundamentals

A FHIR resource is expressed as JSON or XML. Each resource is designated by a 'resourceType' (JSON) or 'name' (XML) such as 'Observation' and a structure definition. A structure definition consists of a sequence of elements that occur in a specified order. Each element has a *name*, a *cardinality* (how many times it may occur within the structure) and a *FHIR Data Type*. Data Types are pre-specified groups of elements which themselves have names, cardinalities, and Data Types. The C programming language analog would be a struct. Eventually all Data Types resolve into simple or primitive Data Types just as all C-structs eventually resolve to C-language primitives like signed and unsigned ints and floats. FHIR currently defines 16 primitive Data Types. Only primitive Data Types can take a value.

Data Types also have names. Primitive Data Type names are in all lower case like 'code' or 'integer' whereas complex Data Type names use upper case letters like 'Identifier' or 'CodeableConcept' or 'Coding'. Do not confuse resource element names with Data Type names. *It is the element names that appear on the wire as the JSON or XML tags and not the Data Type names*. The names of elements and Data Types are often the same or similar, for example the Observation resource has an element with the name 'identifier' which is of Data Type Identifier. There is also an element with the name 'code' which one might assume is of the Data Type 'code' but it is not; it is of Data Type CodeableConcept. The STU3 3.0.1 defined Data Types are defined here http://hl7.org/fhir/datatypes.html#2.26.0.

I.1.1.1 Notation

For convenience, this document denotes elements of a resource using a dotted notation on the resource value. For example, the identifier element of the Observation resource is denoted by 'Observation.identifier'. Since the identifier is of Data Type 'Identifier', the Observation.identifier element itself has the six child elements of the Data Type 'Identifier'. The 'type' element of the Identifier is then denoted by Observation.identifier.type. Since the type element is of Data Type CodeableConcept, the Observation.identifier.type itself has the two child elements of the Data Type CodeableConcept. The 'coding' element of the Data Type CodeableConcept is then denoted by Observation.identifier.type.coding. The coding element is of Data Type Coding which has five elements all of which are primitives. Thus, all the possible elements of the Observation identifier resolved to primitives are as follows:

Observation.identifier.use

Observation.identifier.type.coding.system

Observation.identifier.type.coding.version

Observation.identifier.type.coding.code

Observation.identifier.type.coding.display

Observation.identifier.type.coding.userSelected

Observation.identifier.type.text

Observation.identifier.system

Observation.identifier.value

Observation.identifier.period.start

Observation.identifier.period.end

Observation.identifier.assigner.reference

Observation.identifier.assigner.display

For clarity, the structure elements and their values are often represented in tables where the primitives of the resource structure are shown in italics. Only the primitives can take on values. An example is shown in Table I-1.

Example table representation of resource elements

Table I-1 – Example table representation of resource elements

Observation structure	Value
identifier.	
use	"official"
type.	
coding.	
system	"http://hl7.org/fhir/v2/0136"
version	"1"
code	"y"
display	*Some text*
userSelected	false
text	*Some more text*
system	"urn.iso.std.iso:11073:10101"
value	67666
period.	
start	"2017-02-13T05:42:58.657-05:00"
end	"2017-02-13T05:45:58.657-05:00"
assigner.	
reference	"Patient/45693"
display	*Some more text*

On the wire, the identifier element shown in Table I-1 would appear as JSON as indicated in Figure I-1.

```
"resource":{
  "resourceType":"Observation",
      Other elements
  "identifier":[
    {
      "use":"official",
      "type":{
        "coding":[
          {
            "system":"http://hl7.org/fhir/v2/0136",
            "version":"1",
            "code":"y",
            "display":"I do not know what this is",
            "userSelected":"false"
          }
        ]
        "text":"an identifier code",
```

```
      }
      "system":"urn:iso:std:iso:11073:10101",
      "value":"67666",
      "assigner":{
         "reference":"Patient/45693",
          "display":"who assigned this"
      }
    }
  ]
      Other elements
}
```

Figure I-1 – JSON representation for the resource elements in Table I-1

The array notation is present since the identifier and coding elements have 0 to many cardinalities.

I.1.1.2 The Base DomainResource and Resource

Every FHIR resource inherits either the elements of the DomainResource, which inherits the elements of the Resource, or it inherits the elements of the Resource alone. It is easy to confuse the overloaded use of the word 'resource'. The DomainResource and Resource never exist by themselves but are always part of some generic resource.

I.1.1.3 The id element

The base Resource contains a logical 'id' element which uniquely identifies the resource *on a given server*. It is a restricted alpha-numeric string of no more than 64 characters. The logical 'id' is an important element as it is used in references. When a resource references another resource, the reference uses the logical id. The logical id is used extensively in FHIR as it is the means to link resources together to describe a health care event. For example, a health care event such as a blood pressure measurement taken by a patient in the home would consist of an Observation resource containing the measurement with a reference element pointing to the DeviceComponent resource describing the blood pressure device and a reference element pointing to the Patient resource representing the patient. In FHIR that pointer, if present and pointing to a resource, is required to be the logical id.

I.1.1.4 The identifier element

The Patient, DeviceComponent, and Observation resources also contain an 'identifier' element of data type 'Identifier'. It is used extensively in the CDG mapping. It is easy to confuse the 'identifier' with the logical 'id' since they both identify something. The 'identifier' differs from the logical id in that the identifier has a semantic meaning; it uniquely identifies something about what the resource represents and has meaning outside the context of the resource. The logical id is an opaque value that has no semantic meaning. An example of an Identifier in a Patient resource might be a US social security number, a Scandinavian person number, or an XDSb patient identifier and institutional authorization number. If a resource containing an identifier were to move from one server to another, the logical id would likely change, whereas the identifier would not.

I.1.1.5 Required elements

Examining the elements of the FHIR resources reveals that there are few elements that are required (have a minimum cardinality of one). This limited number of requirements allows use cases, such as this mapping, to specify requirements as they see fit. The only element that is required in every resource is the logical id. However, one may note that in the structure definition of the Resource which contains the logical id element, the minimum cardinality is 0. The reason for the 0 is that in some cases the logical id is added by the FHIR server, for example when the client uploads a single resource using a create operation (HTTP POST). In that case the client does not include the logical id element. However, when residing on a FHIR server, all resources will contain a logical id.

I.1.1.6 Unused elements

Most resource elements in the FHIR specification have a minimum cardinality of 0. The 0 minimum indicates they do not have to be present. If they are not used, they are not transmitted. Placeholders for unused elements are not needed.

I.1.1.7 Choice elements: element[x]

In examining the FHIR resource structure definitions there are some elements designated by element-name[x] followed by list of element names. What that means is, depending upon the situation, the actual element will be one and only one of the elements in the list. An example pertinent to the CDG mapping is the value[x] element of the Observation resource. When the observation is a quantity such as a weight, the valueQuantity element is used which is of Data Type Quantity. If the observation is a code (which might come from an Enumeration metric) the valueCodableConcept element is used which is of Data Type CodeableConcept.

I.1.2 Data Types

There are numerous data types used in the FHIR specification. Only a few of them are used in this mapping. This section examines the more complex data types that are frequently used.

I.1.2.1 CodeableConcept

Codes are ubiquitous in data representations as they provide a means for machine processing. Codes are often numbers or alphanumeric strings. In all cases where codes are used there must exist a dictionary which explains what the codes mean. In something as complex as health care, one can be sure that there are several sets of codes and associated dictionaries. The CodeableConcept is a data type that has two elements, a free text element for human consumption and a coding element, which is of Data Type Coding containing primitive Data Types for the code itself, the code system (dictionary), code version, and other pieces of metadata describing the code hierarchy.

It should be noted that the Coding Data Type has a cardinality of 0 to many. What this 'many' allows is the expression of the *same* concept in more than one coding system. A common problem when it comes to codes is that different realms (countries) often have different requirements for coding systems. The CDG mapping requires the use of the 11073 10101 (MDC) coding system *since that is what the device provides* but the application can add translations to other coding systems such as LOINC or SNOMED CT as required by the realm.

What one cannot do when the Coding Data Type has a cardinality of 0 to many is to include codes representing multiple concepts. A case in 11073-20601 where it becomes tempting to do just that is the handling of the ASN.1 BITs codes. Note that neither FHIR or CDAs has a way to handle ASN.1 BITs, so a special mapping of the ASN.1 BITs to codes has been developed. This mapping defines a unique code for each bit setting. ASN.1 BITs measurements may have more than one bit set and in that case it will be mapped to more than one code. It is tempting to list the codes for each one of the bit settings in a single CodeableConcept. That is not allowed and is semantically inconsistent since each bit setting is something different. It would violate the meaning associated with the CodeableConcept which expresses a single concept. Thus if one has to map an ASN.1 BITs measurement that has five settings of interest, one would need five CodeableConcepts.

I.1.2.2 Quantity

This complex data type is used to express values that are physical quantities with units. All 11073-20601 numeric metric measurements map to Quantities. The Quantity is a complex Data Type consisting of five primitives; the value (which would be a number such as 75.6), the unit which is a human readable string, a code which is the unit as a code, and system which is the coding system. There is no 1 to many cardinality in this case which means there is no means to express the units in more than one coding system. There is also a comparator element which is a code for stating that the

reported value is less than, greater than, etc. than the actual value. This information is not available by the IEEE 11073-20601 protocol and therefore is not used in the CDG mapping.

I.1.2.3 SampledData

This complex Data Type is used to represent Quantity data that is periodic. If the data is periodic and the period is known, one knows the units and time stamp of each sample given the time stamp and units of the first sample. The periodicity allows one to specify the quantity and time stamp information once and the respective values can then be derived for each following sample. This approach is much more efficient than creating an entire Observation resource for each data sample. Since the number of samples may be large, for example ECG waveforms, the SampledData Data Type also provides scaling information in addition to the initial quantity description. Scaling can reduce the size of the data samples when the samples vary by a small amount about a given value that may be large.

The most confusing element in the SampledData Data Type is the origin element of Data Type SimpleQuantity. First, the SimpleQuantity is identical in structure to the Quantity Data Type except it does not have the comparator element. Otherwise, the meaning of the origin element in the specification is misleading. The origin.value is what one adds to the rescaled data value to get the original measured value. A better description of the origin.value element would be an offset. Thus to create the data values from the sensor data one would use the following:

```
valueSampledData.data[i] = (sensorData[i] – valueSampledData.origin.value) /
valueSampledData.factor
```

On the receiving end one would obtain the original sensor data by

```
sensorData[i] = valueSampledData.data[i] * valueSampledData.factor +
valueSampledData.origin.value
```

Clearly if the scale factor is 1 and the 'offset' origin value is zero, then the values in the data array are the measured sensorData values.

I.1.3 FHIR Data Model vs FHIR Transaction Protocol

A constant source of confusion is that FHIR consists of two separate parts. One concerns the exchanges of which there are three; "RESTFul FHIR", "FHIR Messaging", and "FHIR Documents". The second concerns the FHIR data model (resources). The "RESTful FHIR" API is the exchange of interest to CDG which consists of the create, update, conditional create, read, delete, etc., interactions. The interactions are based upon, but do not exactly follow REST. These interactions describe how resources are exchanged between a RESTful FHIR client and RESTful FHIR server. The second part is the FHIR data model. The FHIR data model describes how data is represented in FHIR format. These are the resources and their structure definitions. *There is no requirement that the FHIR data model can only be used in the context of one of the FHIR exchange protocols*. The FHIR upload to an H&FS server that is **not** a RESTful FHIR server is an example of using the FHIR data model but not the FHIR transaction protocol. In this case the upload is identical to the PCD-01 upload using hData except that the information content of the PCD-01 message is replaced by a collection of FHIR resources; the V2 segments of PCD-01 have been replaced by FHIR resources. The transaction in this case has no knowledge of, or cares about, the data payload.

The mapping detailed in these guidelines is only concerned with specifying the use of the FHIR data model for representing IEEE 11073-20601 data.

I.1.4 IEEE 11073-20601 to FHIR Mapping

In this discussion, the mapping of patient data is excluded. Sensor devices currently provide no practical patient information and the population of the FHIR Patient resource is discussed separately.

I.1.4.1 Data Representation

The goal of the CDG mapping is to represent the IEEE 11073-20601 data in terms of FHIR resources without loss of information. The IEEE 11073-20601 specification is complicated and it is not possible to discuss the protocol in detail here. It is assumed the basics of IEEE 11073-20601 are understood.

IEEE 11073-20601 represents device and measurement data as objects containing attributes. The set of objects and their concomitant attributes is called the Device Information Model or DIM. Measurements are represented by Metric objects and sensor device information is represented by the Medical Device System (MDS) object. At the highest level the IEEE 11073-20601 object is equivalent to a FHIR resource and the IEEE 11073-20601 Object attributes are equivalent to FHIR Resource elements. The three types of IEEE 11073-20601 Metric Objects (Numeric, Real Time Sample Array (RTSA), and Enumeration) map to FHIR Observation resources and the IEEE 11073-20601 MDS object maps to the FHIR DeviceComponent resources.

IEEE 11073-20601 attributes are ASN.1 structures. These ASN.1 structures are analogous to the FHIR Data Types. ASN.1 structures, like the C programming language structs, contain elements which are either additional ASN.1 structures or ASN.1 primitives like INT-U8s, which are 8-bit unsigned integers, or SFLOATs, which are 16-bit float representations. Though not important for the mapping, ASN.1 structures are self-defining Type-Length-Value (TLV) structs allowing parse-and-ignore. If the recipient knows the type (T), the structure value (V) can be parsed, if it does not, the length (L) field tells the parser how far to go to get to the next ASN.1 element which the parser may understand.

Like most machine-readable data models IEEE 11073-20601 uses codes to represent concepts. In the IEEE 11073-20601 case these are the 11073 10101 nomenclature codes. The human reader then needs a dictionary to convert these codes into comprehendible language. The nomenclature codes are used to specify measurement types, in some cases measurements, and units. They are also used to identify attributes, APDU types, Objects, and other internal IEEE 11073-20601 protocol processes. The 32-bit nomenclature codes are broken into *partitions* (the most-significant 16 bits) and *term codes* (the least significant 16 bits). The partitions identify logical groupings for the term codes like those for disease management, dimensions (where all the units are), health and fitness, independent living, among others. For the aid of the human reader 11073 10101 also defines a standard reference identifier for each nomenclature code which are human readable, for example, MDC_TEMP_BODY which corresponds to the nomenclature code, 150364, that represent a body temperature. Note that 150364 in HEX is 0x0002 4B5C which means the code has partition 2 and term code 0x4B5C. Viewing these codes in HEX is a convenient way of determining the partition and term code. All reference identifiers have the prefix MDC. For this reason, as well as its brevity, the nomenclature codes are referred to as MDC codes in this document. The reference ids do not appear on the wire in the protocol exchange.

I.1.4.2 Selection of resource elements by Protocol

An important aspect of this specification is that *only those values obtained by protocol are considered for mapping*. For example, the FHIR Observation resource 'interpretation', 'specimen', and 'encounter' elements cannot be populated with data delivered from the sensor device. Population of these elements would require out of band information and that is out of scope for this mapping. If a FHIR resource element cannot be populated with information obtained from the sensor via the CDG PHD interface, it is not required to be populated in this specification.

I.1.4.3 Generic Mapping

This mapping is also designed to be generic and as future proof as reasonable. To accomplish this goal, *the mapping is based upon the 11073 attribute without requiring any knowledge of what the*

attribute value means in a given context. For example, non-time stamp observational metric attributes are always mapped to value[x] elements;

– the nu-observed-value attributes map to valueQuantity elements,

– sa-observed-value attributes map to valueSampledData elements,

– enumeration oids and ASN.1 BITs map to valueCodeableConcept elements, and

– enumeration strings map to valueString elements.

The mapping is based solely upon the attribute type. Note that this approach is not the only possible mapping one could do. If one understands the meaning of the attribute value there are more possibilities. For example, an enumeration OID attribute value, which is an MDC term code, might specify a code representing a body site location or a testing method. One could then legitimately map that value to the 'bodySite' or 'method' element in the Observation instead of the valueCodeableConcept. The problem with that approach is if the code were introduced in a new specialization that was completed after the PHG was already operational out in the field and had to operate with a new PHD using this specialization, the PHG would not have this detailed knowledge and could not perform that mapping. By mapping simple oid attributes to valueCodeableConcept elements in all cases, a PHG would not need to understand the meaning of the code and future codes can be handled transparently. Recipients of the data that understand the code would be able to correctly interpret the meaning whether it is in the valueCodeableConcept element or method element. In the former, the code value itself identifies the measurement as a 'body site' or 'method' type.

I.1.4.4 Measurement Mapping

Both 11073 metric objects and FHIR Observation resources provide a model to represent a measurement. To map one to the other, it is advantageous to note that almost all sensor measurements are described by the following generic terms:

– type: a means of stating what the measurement is. In both HL7 and 11073 data models, the 'type' is specified by a code

– value: a set of one or more numbers or a code that is the measurement.

– units: may not be needed but if needed typically a code

– time stamp

– duration: may not be needed but if needed a period of time

– additional optional descriptions like the accuracy, status, statistical nature, etc.

Mapping the IEEE 11073-20601 metric objects to FHIR Observation resources is nothing more than obtaining the above information from the IEEE 11073-20601 Metric Objects and placing it in the semantic equivalent in the FHIR Observation resources.

I.1.4.4.1 Measurement representation in a FHIR Observation

On the FHIR side, the elements that describe a sensor measurement are straightforward and can be summarized as below:

– The type is given by the Observation.code

– The value and units are given by Observation.value[x] which is

  - valueQuantity if the value is a quantity that has units
  - valueCodeableConcept if the value is a code
  - valueSampledData if the value is periodic sequence of quantities

- valueString if the value is a human readable string (rare but it does happen)

– The time stamp is given by Observation.effective[x] which is
  - effectiveDateTime if the time stamp is an instance in time
  - effectivePeriod if the measurement has a duration

– Measurement errors are given by Observation.dataAbsentReason

A table representation for the case when the measurement is a single quantity is shown in Table I-2.

Table I-2 – Example table resource elements representation: measurement of a single quantity

Observation structure	Measurement Concept
code.	*What the measurement is*
coding	
system	"urn:iso:std:iso:11073:10101" *(MDC code system)*
code	type
display	
effectiveDateTime	time stamp
valueQuantity.	*The value of the measurement; in this case a quantity*
value	value (numeric)
units	*UCUM string for the units*
system	"urn:iso:std:iso:11073:10101" *(MDC code system)*
code	units *(as a code)*

If the measurement value requires a set of quantities to describe it, such as an acceleration which has x, y, and z components or the blood pressure which has systolic, diastolic, and MAP components, a more complex representation is needed. In IEEE 11073-20601, such measurements are represented by compound observational attributes. FHIR uses the Observation.component element in these situations. An Observation.component provides a means to further describe the 'primary' observation. A component is essentially an Observation reduced to contain only the code, value[x], and dataAbsentReason elements. It is an 'observation' that helps describe an Observation. A blood pressure measurement in FHIR would then consist of an Observation.code which states that this is a non-invasive blood pressure, the Observation.effectiveDateTime would give the time stamp, and three Observation.component elements for the systolic, diastolic, and MAP pressures, respectively. Each of the three Observation.components would have an Observation.component.code and Observation.component.valueQuantity, where the valueQuantity contains the units.

In table form, the compound would appear as in Table I-3 for a Blood Pressure example.

Table I-3 – Example table resource elements representation: blood pressure

Observation structure	Value
code.	
coding	
system	"urn:iso:std:iso:11073:10101"
code	150020
display	MDC_PRESS_BLD_NONINV: non-invasive blood pressure
effectiveDateTime	"2017-02-13T05:42:58.657-05:00"
component.	

Observation structure	Value
code.	
coding.	
system	"urn:iso:std:iso:11073:10101"
code	150021
display	MDC_PRESS_BLD_NONINV_SYS: systolic blood pressure
valueQuantity.	
value	105
units	"mmHg"
system	"urn:iso:std:iso:11073:10101"
code	266016
component.	
code.	
coding.	
system	"urn:iso:std:iso:11073:10101"
code	150022
display	MDC_PRESS_BLD_NONINV_DIA: diastolic blood pressure
valueQuantity.	
value	70
units	"mmHg"
system	"urn:iso:std:iso:11073:10101"
code	266016
component.	
code.	
coding.	
system	"urn:iso:std:iso:11073:10101"
code	150023
display	MDC_PRESS_BLD_NONINV_MEAN: mean blood pressure
valueQuantity.	
value	81.7
units	"mmHg"
system	"urn:iso:std:iso:11073:10101"
code	266016

The Observation.component is also used to describe other additional features of an observation, such a IEEE 11073-20601 Supplemental-Types attribute. If one is familiar with PCD-01, any time one has an OBX segment with child OBX segments such as a facet, in FHIR one would have an Observation.component containing the content of that OBX child. There would be an Observation.component for every OBX child.

I.1.4.4.2 Measurement representation in IEEE 11073-20601 Metric Objects

On the IEEE 11073-20601 side, extracting the essential measurement quantities from the Metric Object attributes for the simplest situation is straightforward; the measurement type is given by the

Type attribute, the time stamp by the Absolute- or Base-Offset-Time-Stamp attribute, a quantified value by the Basic-Nu- or Simple-Nu-Observed-Value attribute and Unit-Code attribute, and a coded value by the Enum-Observed-Value-Simple-Oid attribute. In this idealized case, the metric Type attribute maps to the Observation.code element, the metric absolute or base offset time stamp attribute maps to the Observation.effectiveDateTime element, the Basic-Nu- or Simple-Nu-Observed-Value and Unit-Code attribute map to the Observation.valueQuantity while the Enum-Observed-Value-Simple-Oid attribute maps to the Observation.valueCodeableConcept. This is summarized in Table I-4.

Table I-4 – Mapping of simple IEEE 11073-20601 Metric Objects into FHIR observations

FHIR Observation structure	Simplest IEEE 11073-20601
code	From the metric Type attribute
effectiveDateTime	From the metric Time stamp attribute
One of:	
valueQuantity	From the *-Nu-Observed-Value and Unit-Code attributes
valueCodeableConcept	From the Enum-Observed-Value-* attributes
valueString	From the Enum-Observed-Value-Simple-Str attribute

However, the Metric Object can get complicated and this simple one-to-one correspondence no longer works. If the Metric Object also contains a Metric-Id attribute, the Type attribute alone is not sufficient to describe the type of measurement. According to IEEE 11073-20601, the Metric-Id attribute provides a more informative description about the measurement type than the Type attribute itself; for example, the Type attribute may state that the measurement is a body temperature, whereas the Metric-Id attribute states that it is an oral body temperature. The Type attribute has a partition and term code component which are combined to get the 32-bit MDC code. When there is a Metric-Id attribute, the Metric-Id attribute value replaces the term code component of the Type attribute and the Type's partition and Metric-Id's nomenclature code are used to compute the 32-bit MDC code. This more descriptive value is then mapped to the Observation.code element.

If the Metric Object contains a Metric-Id attribute, it may also contain a Metric-Id-Partition attribute. If so, the partition component of the Type attribute is replaced by this value. The MDC 32-bit code is then generated from the Metric-Id-Partition value and the Metric-Id term code value. This 32-bit code value is then mapped to the Observation.code element.

Further, there are three complex observational attributes which contain their own term code sub-structure for describing the measurement and these always take precedence. Consequently, to obtain the type of measurement as a 32-bit MDC code from an IEEE 11073-20601 metric, one needs to use the following kind of algorithm:

– Obtain the partition and term code components of the Type attribute

– If there is a Metric-Id attribute or Nu-Observed Value or Enum-Observed-Value attribute replace the term code component of the Type attribute with the term code obtained from these attributes. Note that the term code from the Nu-Observed Value or Enum-Observed-Value attribute takes precedence over any contribution from the Metric-Id attribute.

    - If in the previous situation there is also Metric-Id-Partition attribute, replace the partition component of the Type attribute with the Metric-Id-partition value

– Compute the 32-bit MDC code from the obtained partition and term code values and map to the Observation.code element.

Obtaining the unit code and measurement status has a similar twist. In most cases the Unit-Code and Measurement-Status attributes suffice, but if there is a Nu-Observed-Value attribute, the unit code is contained in this attribute instead of the Unit-Code attribute, and if there is a Nu-Observed Value or Enum-Observed-Value attribute, the measurement status is obtained from these attributes instead of the Measurement-Status attribute.

The point is that the mapping is not always one-to-one and a generic mapping that supports all the IEEE 11073-20601 modelling options must consider these options.

I.1.4.5 Protocol-only Attributes

There are also some metric attributes which are unrelated to the measurement and are only used in the protocol itself for decoding. The attribute value map and handle are examples of attributes that fall into that category. Once they are used to decode the string of bytes received from the device into the various measurement elements, they are no longer needed and are of no interest to the downstream receiver of the measurement. There are also IEEE 11073-20601 Objects that fall into the protocol-use only category. The PM Store, PM Segment, and Scanner objects are examples of such objects. These objects are only used to decipher the packets of bytes (APDUs) coming from the sensor device.

I.1.4.6 Sensor Device Properties Mapping

The sensor device properties are items like the serial number, firmware version, model number, manufacturer name, system id, regulation and certification status, etc. This *static* information comes from the IEEE 11073-20601 MDS object. The static information is mapped to the DeviceComponent resource and it is straightforward except for the System-Type-Spec-List when it contains more than one specialization.

To understand the mapping when more than one specialization is defined one needs to look at how classic IEEE 11073-10201 was simplified to IEEE 11073-20601. In the classic model, there is also a Virtual Medical Device (VMD) object in addition to an MDS. The VMD represents a specialization and there can be several of them. In addition, a VMD has its own Production Specification and System Model among other attributes and could represent sub-units from different manufacturers. For simplified medical units where there is only a single 'classic' MDS and VMD the two are combined into a simple MDS and IEEE 11073-20601 is born. To cover the case of a simple device supporting more than one specialization the System-Type-Spec-List attribute is introduced which does not exist in IEEE 11073-10201.

To map PHDs to FHIR in a manner that can harmonize with Point of Care Devices (PoCD) using IEEE 11073-10201, multiple specializations in the System-Type-Spec-List are mapped to multiple DeviceComponents as if each specialization were, instead, a VMD as it would be in IEEE 11073-10201. In other words, revert the IEEE 11073-20601 MDS and System-Type-Spec-List to the IEEE 11073-10201 MDS and VMDs. However, in the PHD mapping, only the 'top level' DeviceComponent resource has the productionSpecification and identifier elements populated. The child DeviceComponent resources require only the type and parent elements to be populated. In PoCD cases, there would likely be several more of the elements populated in the child DeviceComponents consistent with the IEEE 11073-10201 VMD attributes.

I.1.4.7 Dynamic Device Properties Mapping

It should be noted that there is dynamic information which comes from the MDS object such as the current time, battery level, and power status. These types of attributes, if mapped, are represented in Observation resources except for the current time. The current time is handled by the Coincident Time Stamp Observation. Updates of the current time evented through a scan event report are ignored unless associated with a date-time-adjustment. A date-time-adjustment requires the creation of a new Coincident Time Stamp Observation resource.

I.1.4.8 PHG Properties Mapping

The PHG properties are mapped to a DeviceComponent resource in the same manner as the sensor properties. The difference is that PHGs do not have an MDS object. What is done instead is to pretend that the PHG does have an MDS object and its attributes are the properties to be reported. There is one subtle difference; a PHG supports at least two interfaces, the PHD and the H&FS. The PHG needs to report the set of Continua Certified Classes on both interfaces whereas the sensor only needs to report those from the PHD. There is no analogue to the System-Type-Spec-List for a PHG property and there is only one PHG-DeviceComponent.

I.2 FHIR Upload Basics

Once the mapping of the IEEE 11073-20601 data to FHIR resources has been done, these resources need to be delivered to the server. There are two types of servers considered for upload operations in these guidelines and the difference is important.

The first is the "RESTful FHIR" http://hl7.org/fhir/conformance-rules.html server. The RESTful FHIR server is a repository that persists uploaded resources and follows the RESTful FHIR API for such servers. FHIR also defines a "FHIR messaging" and "FHIR documents" compliance models but these are out of scope for the CDG. In the CDG, this kind of server is referred to as a *FHIR Observation Server*. A peer PHG is a *FHIR Observation Client*.

The second is a server that does not know anything about previously uploaded data. The CDG Health and Fitness server for SOAP and hData uploads of PCD-01 messages is an example of a server that is not required to know anything about previous uploads. In this case, the PCD-01 payload of the hData upload is replaced by the equivalent set of information in the form a collection of FHIR resources. Collections of FHIR resources are placed in a special resource called a FHIR Bundle. This upload is only using the FHIR data model, and not the FHIR transaction protocols. In the CDG this kind of server is referred to as a *FHIR Observation Reporting Server*. A peer PHG is a *FHIR Observation Reporting Client*.

When uploads are done to a *FHIR Observation Server*, the PHG may upload individual resources and generally do not upload a resource to a given server that has already been uploaded. Patient and DeviceComponent resources are examples of resources that are typically uploaded once.

When uploads are done to a *FHIR Observation Reporting Server*, all information with respect to the taken measurements needs to be uploaded in a single package, even if some of that information is identical to information that has already been uploaded, for example the patient and device information. In this case uploads of FHIR resources must be in a FHIR Bundle resource that contains the Patient, DeviceComponent, and Observation resources. A FHIR Bundle is a special container resource that allows one to include multiple resources into a single package that can be delivered in a single payload. A Bundle has 0 to many entry elements where each entry element has a resource element containing the resource. The semantic content of this FHIR Bundle with patient, device and observation information is analogous to the content of the PCD-01 message in SOAP and hData uploads. The *FHIR Observation Reporting Server* does not understand FHIR interactions and all that is really happening is that the FHIR data model is being used instead of PCD-01 V2 data model. The *FHIR Observation Reporting Server* will only go as far as to check that the syntax of the uploaded Bundle is legal. However, this specification will require that a transaction Bundle be used with defined interactions such that it would be acceptable to, and consistent with, a *FHIR Observation Server* though perhaps not as efficient. In this manner, the *FHIR Observation Reporting Server* could forward this Bundle to a *FHIR Observation Server* without manipulating the package.

In the CDG, this special type of Bundle generated by a *FHIR Observation Reporting Client* is referred to as a *complete* Bundle. Bundles may also be used by a *FHIR Observation Client* and *FHIR Observation Server* but in that case, there is no need that they be "complete". For example, the Patient and DeviceComponent resources may already have been uploaded so the Bundle may only contain Observation resources. Bundling may be desired by the FHIR Observation Client to limit the

number of transactions and to simplify the handling of measurements containing source handle references.

I.2.1 Interactions

FHIR interactions to a *FHIR Observation Server* are based upon RESTFul FHIR. For uploads, there are three options of interest to CDG uploads, create, update, and conditional create. Their differences are important.

I.2.1.1 Create

The create transaction uses an HTTP POST. When a resource is created, the client is telling the server that this is a new resource and it does not currently exist on the server. The create is then asking the server to accept this resource and store it. When a resource is created in this manner, the client does not specify the logical id of the resource. Instead, the logical id is generated by the server when the server creates the resource in its repository. The value of this logical id is returned to the client in the response so it can reference the uploaded resource in subsequent uploads if needed. Note that if one uses a create operation twice for the same resource, a second copy of the resource will be generated on the server, differing only in their logical id.

If the create transaction is specified for a resource in a transaction Bundle, a logical id may be needed because other resources in the Bundle reference the resource to be created. In this case, the client specifies a special type of logical id called a temporary id. All references in the Bundle to this resource use that temporary id. The server then generates its own logical id any replaces all the temporary ids in the Bundle with its generated logical id.

I.2.1.2 Update

The update transaction uses an HTTP PUT. When a resource is updated, the client is telling the server that this resource may exist and if it does, update it with the new information. The existing resource is replaced with the uploaded resource and the server may update the metadata associated with the resource to indicate the update. If the same resource is updated twice, the resource itself will not change but the versionId and lastUpdated elements of the metadata might be updated; servers are not required to maintain such information though updating the versionId is a "should" requirement in the FHIR specification. If the resource does not exist, the server creates the resource but unlike the create transaction, *the logical id is taken from the uploaded resource*. Thus, an important difference between a create and update transaction is that in the update transaction, the client specifies the logical id. FHIR provides this type of operation to allow uploaders to avoid the overhead of handling responses to get a server-specified logical id but the client is responsible for providing a unique logical id.

The update operation in a transaction Bundle is identical to that of an individual resource upload.

I.2.1.3 Conditional Create

The conditional create is not a standard RESTful transaction. It uses an HTTP POST analogous to the create transaction with an extra HTTP header "If-None-Exist" with the argument being that of a search transaction. What this special operation does is to effectively perform a GET operation for the all resources of the uploaded type that satisfy the search parameters but without returning the results to the client. If the server finds no resources that satisfy the search criteria, the uploaded resource is created as in the create transaction using an HTTP POST with a 201 return code. If one and only one resource is found, the uploaded resource is ignored as it already exists and the server returns 200. If more than one resource is found, a 412 error is returned indicating the search parameters were not selective enough.

The conditional create is used for duplicate prevention and assuring that the resource on the server is left exactly as it was. The client needs to specify a search criteria that is selective enough to find only

the uploaded resource should it exist. In this manner if it does exist, a new duplicate resource is not created AND the existing resource is left exactly as is. When a conditional create is done, the logical id is not specified by the client unless in a Bundle where a temporary id is used and the server creates the logical id if the resource is not found, otherwise the logical id of the existing resource is used if it is found.

The conditional update is similar to the conditional create except that if the resource is found, the conditional update updates the resource which could change it or the metadata.

I.2.1.4 The Transaction Bundle

When a Bundle is uploaded, the operation is always a POST. However, how the Bundle is treated by the server depends upon the Bundle type. THE CDG is only interested in the *transaction* Bundle. A transaction Bundle indicates to the FHIR server that each resource entry shall be treated as if uploaded individually by the individually specified interactions. The server then breaks up the Bundle into its component resources and handles them according to the transaction specified by the entry. These interactions could be creates, updates, conditional updates, or conditional creates. However, there is one significant difference between this Bundle breakup and individual uploads. In the Bundle there could be resources that reference other resources in the Bundle that have not yet been uploaded thus the resource to be referenced has no logical id unless it is to be 'updated'. In this case, a temporary id is used. The temporary id is designated by prefixing the logical id entry with "urn:uuid:" or "urn:oid". The logical id should be a GUID or OID, respectively, and it must be unique within the Bundle. Many servers may accept non-GUIDs or OIDs but one should not rely on it. When a resource within the Bundle needs to reference another resource within the Bundle, this temporary logical id is used. Upon reception, the server creates and or obtains the logical id for the resource and replaces the temporary id and systematically changes all references to this temporary id within the Bundle to this logical id. Note the statement 'creates or obtains' with respect to the logical id. It could be that a conditional create transaction with a temporary id results in finding that the resource already exists. In that case, the logical id of the existing resource is used to update the temporary ids.

The use of the Bundle is convenient for the client when Observation resources are required to reference other Observation resources. For PHD sensors this referencing is needed when handling source handle reference attributes and coincident time stamps. Using a Bundle, the client does not need to wait for or save any returned logical ids in case a reference to that uploaded resource is needed when mapping a newly received measurement. The data flow from the sensor may also be faster than individual resource uploads can be processed which would require internal queuing of the resources by the client.

It is important to understand that if there is any error in the processing of a Bundle, the entire transaction fails. Thus, if a single resource transaction were specified as a conditional create and the search parameters were not selective enough, the failure of this resource transaction would cause the entire Bundle to fail and none of the resources in the Bundle will be placed on the server.

I.2.1.5 CDG FHIR Uploads In Practice

What type of transactions make most sense for the various CDG DeviceComponent and Observation resource uploads? How are these uploads affected by the PHG? To examine these questions, one must examine how these resources are related to one another.

Observations may point to a Coincident Time Stamp Observation and always point to a top-level or child DeviceComponent. The top-level DeviceComponent always points to the PHG-DeviceComponent.

This hierarchy means that if the same stored measurement data is uploaded from the same sensor device by the same patient on two different PHGs running the same software, the top-level DeviceComponent in the two cases will be different since the top-level DeviceComponent.parent

element will point to different PHG DeviceComponents. That means any child DeviceComponents will also be different since they point to their respective top-level DeviceComponents. The Observations, even for the same measurement, will also be different because the Observation.device will point to different child or top-level DeviceComponents.

It may seem incorrect that an identical stored measurement taken by the same patient on the same device when uploaded by two different PHGs create two different Observation resources. However, the two different PHGs may not be perfectly synchronized and that difference in synchronization will result in different measurement time stamps and different Coincident Time Stamp Observations. Generating a separate set of resources for each PHG is the only way to maintain time integrity and auditability.

Given the above, each PHG is more or less responsible for the integrity of the resources it generates without having to be concerned with the fact another PHG may have generated a similar set of resources.

Bibliography

For a list of non-normative references and publications that contain further background information, see [ITU-T H.810]. Additionally, this document refers to the following document.

[b-IEEE GL-EUI-64] IEEE GL-EUI-64 (2017-08-03), *Guidelines for Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI), and Company ID (CID).*
http://standards.ieee.org/develop/regauth/tut/eui.pdf (Visited 2018-02-21)