H.812.3 Capability Exchange Capability

Version 2016

August 4, 2016

Table of Contents

Figures

Tables

0 Introduction

The Continua Design Guidelines (CDG) define a framework of underlying standards and criteria required to ensure the interoperability of devices used for applications monitoring personal health and wellness. It also contains additional design guidelines for interoperability that further clarify or reduce the options in underlying standards or specifications, or by adding a feature missing in an underlying standard or specification.

This document defines the additional design guidelines for the Capability Exchange Enabled Personal Health Gateway (PHG) and Health & Fitness Service Certified Capability Class (CCC). The purpose of the capability exchange is to reduce the amount of information that must be pre-configured on a device in order to obtain plug and play interoperability. Specifically, Capability Exchange enables an Personal Health Gateway (PHGs) to know what types of messages can be sent to the Health & Fitness Service, by identifying the defined Continua CCCs. Likewise; Capability Exchange provides a mechanism for the PHG to inform the Health & Fitness Service of its capabilities, to enable the Health & Fitness Service to tailor its communication with the PHG. Capability Exchange is mandatory for all Health & Fitness Services while it is optional for PHGs.

It is assumed that the PHG is pre-provisioned with a URL, or a set of URLs, denoting the service endpoint of one or more Health & Fitness Services. The capability exchange process takes place when the PHG first contacts a Health & Fitness Service. It may also take place intermittently, to update the information received in the first capability exchange. In most cases, the set of Continua CCCs implemented at a Health & Fitness Service changes slowly, if at all. Therefore, it is expected that the PHG can store the information about Services capabilities, and optionally, implement a policy for periodically updating that cache. A PHG might identify several Health & Fitness Services in this way, and communicate with one or more for different purposes.

The Health & Fitness Service describes the information about its supported CCCs in a file called “root file”. The root file is a special resource that describes the properties of CCCs and how PHG can start information exchange with those CCCs. The root file and other features of the exchange come from an HL7 standard called hData [HL7 HRF] [OMG HRT]. hData not only defines the root file format, but also defines the operations for exchanging root files, using HTTP using GET and POST operations, often referred to as “REST” (for representational state transfer).

Each Continua CCC (in addition to capability exchange) will use the root file to document information relevant to that capability, including the capability name, the information that can be exchanged under the capability and its format, and URLs for REST operations, if supported by that capability. Details are given in the documentation for the respective Continua CCCs.

This Guidelines document is part of the H.810 Interoperability design guidelines for personal health systems published by the Personal Connected Health Alliance. See [H.810] for more details.

0.1 Organization

This CCC guideline is organized in the following manner.
Clauses 0-5: Introduction and Terminology – These clauses provide overview information helpful in comprehending the remainder of the document.
Clause 6: Use Cases - This clause provides motivating examples.
Clause 7: Behavioral Model - This clause is an overview of sequences of interactions and summarizes typical iterations, constraints, and exceptions.
Clause 8: Implementation Guidance - This clause provides an informative description of the implementation of the Capabilities Exchange CCC
Clause Annex A Normative Guidelines - This clause specifies the normative requirements that must be followed by the Capability Exchange CCC.

0.2 CCC Guideline Releases and Versioning

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

0.3 What's New

To see what is new in this release of the design guidelines refer to Clause 0.3 of [H.810]

1 Scope

This document specifies design guidelines for the Capability Exchange Enabled PHG and Capability Exchange Enabled Services CCCs. The design guidelines specifies the testable requirements that must be implemented by the PHG in order to classify it as Capabilty Exchange Enabled PHG. The Capabilty Exchange Enabled PHG shall be able to retrieve root file from the Health & Fitness Service and be able to validate that the root file conform to the HL7 hData hRF document. In addition, the design guidelines specify testable requirements for a Health & Fitness Service which details how a Capabilty Exchange Enabled Health & Fitness Service shall respond to the requests from a Capabilty Exchange Enabled PHG and shall be able to validate that the root document conform to the HL7 hData hRF document

2 References

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

3 Definitions

This guidelines document uses terms defined in [H.810]

4 Abbreviations and Acronyms

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

5 Conventions

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

6 Use Cases

The use cases below are focused on the needs identified for capability exchange.

6.1 PHG Obtains Health & Fitness Service Information

Outpatient Adam Everyman is provided with health measurement devices that interact wirelessly with a smart phone application (the PHG). Adam's health practioner provides a URL in the form of a QR Code (for example) that can be scanned by the smart phone app during the configuration process, directing the PHG to the Disease Management Organization (DMO), a remote monitoring site. DMO remotely monitors patients at home and collects health information from health measurement devices installed at Adam's home. During configuration, the smart phone app contacts the URL and downloads an XML file (the "root file") containing information about DMO's services. By parsing the root file, the PHG determines the Continua Certified Capability Classes (CCCC) supported by the DMO. In this case, the DMO can receive observation uploads and questionnaires using RESTful HTTP and can participate in Authenticated Persistent Session.

6.2 Health & Fitness Service Receives PHG Information

Having discovered that the DMO can support Authenticated Persistent Sessions, the smart phone app now wants to inform the DMO that it also has the capability of supporting Authenticated Persistent Sessions. To do so, the PHG must first authenticate with the DMO. After authentication, the PHG may use an HTTP POST operation to send its root file (which is different than the DMO root file) to the DMO, using the designated URL supplied in the DMO's root file. The PHG's root file contains information on the PHG's capabilities, including the fact that the PHG can support Authenticated Persistent Sessions. If the PHG subsequently initiates an Authenticated Persistent Session with the Health & Fitness Service the Health & Fitness Service will use the information in the PHG's root file to send unsolicited commands to the PHG.

7 Behavioral Models

The following exchange mechanisms are specified for capability exchange service:

– PHG retrieves the Health & Fitness Service root file from the Health & Fitness Service
– The PHG sends its root file to the Health & Fitness Service

The following diagram illustrates transactions related to the capability exchange use cases described in Clause 0.

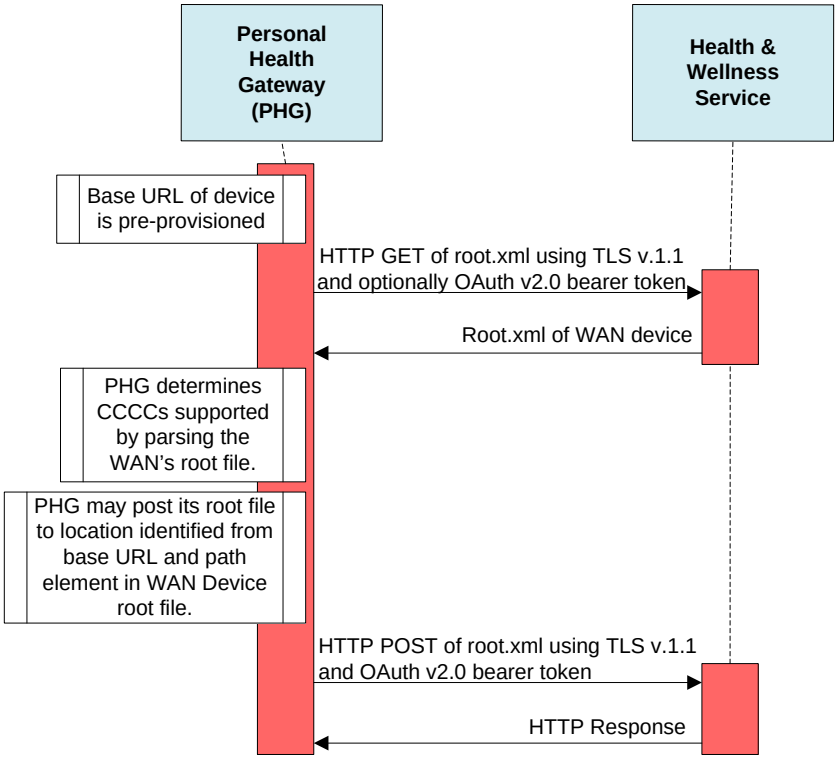

Figure 7-1 – Transactions between PHG and Health & Fitness Service related to root file exchange for determining the capabilities of Health & Fitness Service and PHG.

8 Implementation

8.1 Overview

A PHG supporting Capability Exchange obtains information from the Health & Fitness Service, and vice versa, in the form of a document called “root file”. The “root file”is so named because it exists at the top of the hData hierarchy [HL7 HRF]. The format of the root file is defined in the hData Record Format specification [HL7 HRF]. Health & Fitness Services (except SOAP based Health & Fitness Services) must have the capability to provide the root file in XML format, and can optionally provide it in JSON format. Similarly PHG (except SOAP based PHGs) must be able to process the root.xml file it receives from the Services Interface, and optionally, can also process the JSON equivalent.

The Health & Fitness Serice root file contains several different types of information useful to the PHG:

– A list of the Continua Capability Classes that the Health & Fitness Service supports,
– A list of the resource types that may be exchanged with the Health & Fitness Service in one or both directions,
– Information about the available representations of exchangeable resources,
– The location of the resources in terms of partial URLs,
– Any additional information required by a CCCC listed in the root file.

In the above descrition, the term “resource” is used in the REST sense: a logical entity that may have multiple representations.

Once the Health & Fitness Serice root file has been obtained, the PHG may optionally communicate information back to the Health & Fitness Service in the form of another root file. The root file sent from the PHG to the Health & Fitness Service represents the PHG’s capabilities, resource types, representations, and other parameters defined by specific CCCCs. This step of sending the PHG root file to the Services Interface requires authentication, so the Services Interface can positively identify the PHG that is the source of the root file. The authentication process is not discussed here. Because this step is optional, the PHG root file is not required.

Once capability information has been exchanged, the devices are able to invoke the appropriate protocols in an interoperable fashion. Capability Exchange reduces the amount of information that must be pre-configured on a device in order to obtain plug and play interoperability.

8.2 Root File Exchange

The root file is exchanged using the following REST mechanism:

- The PHG performs an HTTP GET using a TLS v1.1 secure channel, OAuth v2.0 authorization token of type bearer (use of OAuth is optional in the case where an PHG implements only SOAP based observation upload or consent enabled –PHG CCCs) and a pre-configured URL (the “base URL”) to obtain the root.xml file from the Health & Fitness Service. The PHG is expected to be able to parse the root file and determine the capabilities of the Health & Fitness Service.
- Optionally, PHG performs an HTTP POST of its root file to the Health & Fitness Service, using TLS v1.1 secure channel, OAuth v2.0 authorization token of type bearer (use of OAuth is optional in the case where a Health & Fitness Service implements only SOAP based observation upload or consent enabled –Services CCCs) and the relative URL

indicated by the Health & Fitness Serice root file. (The PHG is not assumed to support HTTP server capability, so an HTTP POST is used, rather than a Health & Fitness Service HTTP GET operation.)

More information about root files and REST methods are available in the hData specifications.

8.3 Contents of the Root File

The root file format is described in the HL7 hData Record Format Version 1 Specification [HL7 HRF]. The root files of the Services and PHG will conform to HRF version 1 and validate with the XSD provided with that specification. In this clause, we profile the elements of the root.xml file. Elements not specifically mentioned in this profile follow the element definitions in the HRF standard. The root file contains the following sub-elements under the top-level <root> element:

- version (xs:integer, 1..1) - The version of the hData Record Format used within the root file. The version number for root files complying with this version of the specification is 1.
- profile (0..*) - This element represents a CCCC supported by the Services or PHG application that owns the root file. Each CCCC is described by one <profile> element using the following sub-elements:
  - id (xs:string, 1..1) - The id is the formal name of the CCCC represented by the profile element. For Capability Exchange, the formal name is “CapabilityExchange”. For other device classes, the version-specific formal name will be given in the Continua documentation for that CCCC.
  - reference(xs:string, 1..1) - A reference to the Continua documentation for the CCCC represented by this profile element. The reference string is composed of the URL to the Continua Guidelines repository, along with a string that identifies the name of the document. For Capability exchange the reference string is “*H.812.3 Capability Exchange*”
- resourceType (1..*) - This element represents a resource type associated with one or more of the profiles listed in the root file. A specific resource type can be used in one or more CCCC. A resource type is represented by the following sub-elements:
  - id (xs:string,1..1) - This attribute contains the name for the resourceType. For capability exchange, the only resourceType is “root”. For other CCCCs, the resource id(s) are given in the CCCC documentation.
  - reference(xs:string, 1..1) – A version-specific reference to the semantic definition of the resource type. For the root resource type used by Capability Exchange, the reference is “http://www.hl7.org/implement/standards/product_brief.cfm?product_id=261”.
  - representation (0..*) - This element represents each serialization format of the resource available for “on the wire” communication.
    - mediaType (xs:string, 1..1) - Contains the media type of the resource. For Capability Exchange, the required media type is “application/xml”. An optional second representation is “application/json”.
    - validator(xs:string, 0..*) - An optional reference to a validator for this representation, such as an XML Schema Definition (XSD) or Schematron..
- section (1..*) - A section represents a “virtual file folder” where instances of a certain resource type are found. A section is identified by partial URL, relative to the base URL. Each CCCC may define one or more sections. For Capability Exchange, there is one required section in the Health & Fitness Serice root file.
    - path (xs:string, 1..1) - This text attribute is a path segment, used to construct the full path to the section. For capability exchange, the path is “roots”

- profileID(xs:string, 0..*) - The <id> of the CCCC defining this section. The value of this element MUST be equal to the id attribute of a <profile> element.
- resourcePrefix(xs:boolean, 0..1) - This element is omitted.
- resourceTypeID (xs:string, 0..1) - The value of this element MUST be equal to the id attribute of a <resourceType> element. Only resources whose type matches the resourceTypeID element can appear in the section. If no resourceTypeID is given, the section may not contain resources, only other sections.
- metadataSupport(xs:boolean, 0..1) - This element is omitted.
  - section (section, 0..*) – The subsections belonging to the current section, if any.

In addition to these elements, each CCCC may define extensions to the root file. The required extension elements, if any, will be present when the corresponding CCC is declared in the <profile> element.

8.4 Optional JSON Version of Root File

The Services Interface may optionally support a JSON version of the root.xml file. If the PHG requests “application/json” in the HTTP accept header, and the Services Interface supports the JSON, the Services Interface should return the JSON version of the root file.
The JSON version of the root file contains the same information as the XML version. The transform from XML to JSON and the JSON root file format is discussed in the HL7 Version 3 Specification: hData Record Format, Release 1 [HL7 HRF].

Annex A Normative Guidelines

Table A-1 – Normative Guidelines for Health & Fitness Service

Name	Description	Comments
CapX-HFS-Root-Standard	Root file of the Health & Fitness Service **shall** comply to the HL7 Version 3 Specification: hData Record Format, Release 1 [HL7 hRF].	
CapX-HFS-Root-Security	Health & Fitness Service **shall** support TLS v1.1 as defined in document H.812. All hData based Health & Fitness Services **shall** support OAuth authorization token of type bearer as defined in document H.812.	A Health & Fitness Service that implements only SOAP based observation upload or consent enabled -HFS CCCs is not required to support the Capability Exchange-HFS CCC.
CapX-HFS-Root-Profile	The root file of the Health & Fitness Service **shall** contain a profile element for each CCC it supports.	
CapX-HFS-Root-XML-Version	The Health & Fitness Service **shall** support an XML version of its root file.	
CapX-HFS-Root-JSON-Version	The Health & Fitness Service **may** support a JSON version of its root file.	Note that the HL7 hRF sepecification document does not specifiy schema for validating JSON formatted root file.
CapX-HFS-Root-Validation	The XML root file of the Health & Fitness Service **shall** validate against the hData Version 1 root.xsd	
CapX-HFS-Root-CCC-Conformance	A Health & Fitness Service listing a particular CCC in its root file **shall** conform to the normative guidelines for that CCC.	
CapX-HFS-Root-Version	The version number in the Health & Fitness Serice root file conforming to this specification **shall** be 1	
CapX-HFS-Root-Profile-Element	The Health & Fitness Serice root file **shall** contain a profile element with the id “CapabilityExchange” and reference “http://handle.itu.int/11.1002/3000/hData/CX/2015/01/H.812.3.pdf”	

Name	Description	Comments
CapX-HFS-Root-ResourceType-Element	The Health & Fitness Serice root file **shall** contain a resourceType with id "root" and reference "http://www.hl7.org/implement/standards/product_brief.cfm?product_id=261"	
CapX-HFS-Root-MediaType-XML	The Health & Fitness Serice root file **shall** have a representation element under the "root" resourceType with mediaType "application/xml".	
CapX-HFS-Root-MediaType-JSON	The Health & Fitness Serice root file **may** have a representation element under the "root" resourceType with mediaType "application/json"	
CapX-HFS-Root-Section-Element-Inclusions	The Health & Fitness Serice root file **shall** have a section element with the path "roots", the profileID "CapabilityExchange", the resourceTypeID of "root", and **shall not** specify the resourcePrefix or metadataSupport elements.	
CapX-HFS-Root-Section-Element-Exclusions	The Health & Fitness Serice root file section element with the path "roots" **shall not** specify the resourcePrefix or metadataSupport elements.	
CapX-HFS-REST-Standard	The Services Interface responses to HTTP method calls **shall** comply to OMG hData REST Binding for RLUS [OMG HRT]	
CapX-HFS-REST-GET-XML-Response	By default, the Health & Fitness Service **shall** respond to a root file GET request (i.e., an HTTP GET on [baseURL]/root) by returning the XML version of the Health & Fitness Services's root file.	
CapX-HFS-REST-GET-JSON-Response	A Health & Fitness Service having an "application/json" representation element under the "root" resource type in its root file **shall** return the JSON version of its root file in response to a PHG's GET request that specifies "application/json" in the HTTP accept header. If the Health & Fitness Service doesn't have JSON version then it **shall** return HTTP status code 501 Not Implemented.	
CapX-HFS-REST-POST-Response	The Health & Fitness Service **shall** accept a HTTP POST at the URL [baseURL]/roots only if the sending PHG has a valid authorization token of type bearer as defined in document H.812	

Name	Description	Comments
CapX-HFS-REST-POST-Unauthenticated-Sender	If any content is posted to the Health & Fitness Service by an unauthorized sender, then the Health & Fitness Service **shall** respond with a HTTP 401 Unauthorized error.	
CapX-HFS-REST-POST-XML-Validation	When an XML file is POSTed to the URL [baseURL]/roots, the Health & Fitness Service **shall** validate the file against the hData Version 1 root.xsd and return HTTP 201 if the file is validated, and in case of validation failure, return HTTP 422 Unprocessable Entity.	
CapX-HFS-REST-POST-JSON-Validation	When a JSON file is POSTed to the URL [baseURL]/roots, the Health & Fitness Service **shall** return HTTP 422 Unprocessable Entity if the JSON does not conform to the hData root file specification and otherwise return HTTP 201.	Note that the HL7 hRF sepecification document does not specifiy schema for validating JSON formatted root file.
CapX-HFS-REST-POST-Response	In response to a successful POST of the PHG root file to [baseURL]/roots, the Health & Fitness Service **shall** return the unique URL of the newly-created root resource.	

Table A-2 – Normative Guidelines for PHG Device

Name	Description	Comments
CapX-PHG-REST-XML-Request	Given the URL of a Health & Fitness Service complying with Capability Exchange ("baseURL"), the PHG device **may** obtain the root file of the Health & Fitness Service using an HTTP GET operation.	
CapX-PHG-REST-Services-Root-security	PHG **shall** obtain root file of the Health & Fitness Service using TLS v1.1 secure channel as defined in document H.812. All hData based PHGs **shall** support OAuth authorization token of type bearer as defined in document H.812.	A PHG that implements only SOAP based observation upload or consent enabled -PHG CCCs is not required to support the Capability Exchange-PHG CCC.
CapX-PHG-REST-XML-Request	The PHG device **shall** be able to request the Health & Fitness Serice root file in XML format by specifying "application/xml" in the HTTP accept header.	
CapX-PHG-REST-JSON-Request	The PHG device **may** request the Health & Fitness Serice root file in JSON format by specifying "application/JSON" in the HTTP accept header	Note that the HL7 hRF sepecification document does not specifiy schema for validating JSON formatted root file.
CapX-PHG-Root-POST	PHG **may** do an HTTP POST of its root file at the URL [baseURL]/roots using TLS v1.1 secure channel as defined in document H.812 and a valid authorization token of the type bearer as defined in document H.812. The authorization token **shall** be obtained according to the guidelines described in the document H.812.	
CapX-PHG-Root-Standards	Root file of the PHG **shall** comply to the HL7 Version 3 Specification: hData Record Format, Release 1 [HL7 hRF]	
CapX-PHG-Root-Profile	The root file of the Health & Fitness Service **shall** contain a profile element for each CCC it supports.	

Name	Description	Comments
CapX-HFS-Root-CCC-Conformance	A PHG listing a particular CCC in its root file **shall** conform to the normative guidelines for that CCC.	For example profile and section info in the root file for a CCC are defined by that specific CCC.

Appendix I Root File Inclusions for Capability Exchange

I.1 Required root file inclusions for Health & Fitness Service

```
<profile>
   <id> CapabilityExchange</id>

<reference>http://handle.itu.int/11.1002/3000/hData/CX/2015/01/H.812.3.pdf</refe
rence>
</profile>

<section>
  <path>roots</path>
  <profileID> CapabilityExchange</profileID>
  <resourceTypeID>root</resourceTypeID>
</section>

<resourceType>
  <id>root</id>
  <reference>
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=261
  </reference>
  <representation>
     <mediaType>application/xml</mediaType>
  </representation>
  <representation>  <!-- optional -->
     <mediaType>application/json</mediaType>
  </representation>
</resourceType>
```

I.2 Schema for the root.xml

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
targetNamespace="http://hl7.org/schemas/hdata/2013/08/hrf"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:hrf="http://hl7.org/schemas/hdata/2013/08/hrf">
  <xs:element type="xs:string" name="id"/>
  <xs:element type="xs:float" name="version"/>
  <xs:element type="xs:dateTime" name="created"/>
  <xs:element type="xs:dateTime" name="lastModified"/>
  <xs:element type="xs:string" name="name"/>
  <xs:element type="xs:anyURI" name="uri"/>
  <xs:element type="xs:string" name="email"/>
  <xs:element type="xs:string" name="reference"/>
  <xs:element type="xs:string" name="path"/>
  <xs:element type="xs:string" name="profileID"/>
  <xs:element type="xs:boolean" name="resourcePrefix"/>
  <xs:element type="xs:string" name="resourceTypeID"/>
  <xs:element type="xs:boolean" name="metadataSupport"/>
  <xs:element type="xs:string" name="mediaType"/>
  <xs:element type="xs:string" name="validator"/>

  <xs:group name="extensionElement">
```

```
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:any namespace="##local" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:group>

  <xs:element name="author">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:name"/>
        <xs:element ref="hrf:uri" minOccurs="0"/>
        <xs:element ref="hrf:email" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="profile">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:id"/>
        <xs:element ref="hrf:reference"/>
        <xs:group ref="hrf:extensionElement" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="section">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:path"/>
        <xs:element ref="hrf:profileID" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="hrf:resourcePrefix" minOccurs="0"/>
        <xs:element ref="hrf:resourceTypeID" minOccurs="0"/>
        <xs:element ref="hrf:metadataSupport" minOccurs="0"/>
        <xs:group ref="hrf:extensionElement" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="hrf:section" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="representation">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:mediaType"/>
        <xs:element ref="hrf:validator" minOccurs="0" maxOccurs="unbounded"/>
        <xs:group ref="hrf:extensionElement" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="resourceType">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:id"/>
        <xs:element ref="hrf:reference"/>
        <xs:element ref="hrf:representation" minOccurs="0" maxOccurs="unbounded"/>
        <xs:group ref="hrf:extensionElement" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
```

```
    </xs:complexType>
  </xs:element>

  <xs:element name="root">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hrf:id"/>
        <xs:element ref="hrf:version"/>
        <xs:element ref="hrf:created"/>
        <xs:element ref="hrf:lastModified"/>
        <xs:element ref="hrf:profile" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="hrf:section" maxOccurs="unbounded"/>
        <xs:element ref="hrf:resourceType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:group ref="hrf:extensionElement" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
    <xs:key name="PKResourceType">
      <xs:selector xpath="hrf:resourceType/hrf:id"/>
      <xs:field xpath="."/>
    </xs:key>
    <xs:keyref name="FKSectionToResourceType" refer="hrf:PKResourceType">
      <xs:selector xpath="hrf:section/hrf:resourceTypeID"/>
      <xs:field xpath="."/>
    </xs:keyref>
    <xs:key name="PKProfile">
      <xs:selector xpath="hrf:profile/hrf:id"/>
      <xs:field xpath="."/>
    </xs:key>
    <xs:keyref name="FKSectionToProfile" refer="hrf:PKProfile">
      <xs:selector xpath="hrf:section/hrf:profileID"/>
      <xs:field xpath="."/>
    </xs:keyref>
  </xs:element>
</xs:schema>
```

I.3 Required root file inclusions for PHG

There are NO required root file inclusions, however a PHG device listing a particular CCC in its root file (as a profile element) **shall** conform to the normative guidelines for that CCC.

Appendix II hData

hData is a lightweight, web-based specification for exchanging electronic health data. Created in 2009 by US non-profit MITRE Corporation and evolved in cooperation with leaders in the health care industry, hData is the first RESTful standard for health data exchange. The hData specifications have been approved by Health Layer 7 (HL7) and the Object Management Group (OMG).

hData uses REST (Representational State Transfer) over HTTP in a way that separates content, transport, and security. REST is a design pattern that is simple, scalable, and widely adopted. hData is used in all Continua Device Classes, either as the only mechanism, or as an alternative to SOAP based exchange.

Resources are a central concept in REST and in hData. A resource can be practically any piece of information: data about a patient, a device, a prescription, a plan of care, an imaging study, a problem or condition, or a complete medical document such as a Consolidated CDA [HL7 CDA IHE HSC] . For purposes of information exchange, resources can have multiple representations, such as XML or JSON.

Sections represent a virtual arrangement of resources in hData. Sections are analogous to directories in a hierarchical file system, and are defined by paths consisting of one or more forward-slash delimited sub-levels. Each section is associated with a specific type of resource (called *resourceTypes* in hData). For example, resources that represent a person's allergies may be found in a section called *allergy*. The allergy section may contain zero or more instances of an allergy resource. The arrangement of sections forms a tree structure, called the hData Hierarchy (HDH).

URLs uniquely identify each resource. The URL of a resource is a combination of a base URL, a section path, and a resource ID, as follows:

resource URL = (baseURL)/(sectionPath)/(resourceID)

The baseURL is the location of the hData service endpoint, and consists of the protocol (in this case, HTTP or HTTPS), a host identifier (IP address or domain name), and optionally a port. The resourceID is defined arbitrarily by the resource owner, subject to the constraint that the resource URL is unique.

Root Files are provided by hData service endpoints to advertise the resource types (extensions) and the section paths (sections) provided by that service. The format of the root file is described in [HL7 hRF]. The root file is accessed by a HTTP GET operation on the following URL:

root file URL = (baseURL)/root

Content Profiles are the means to achieve interoperability between hData service endpoints. Content profiles are implementation guides that describe an application of hData that promote information interoperability. If every hData service endpoint were to arbitrarily define its own resource types and hData hierarchy, the result would be an ecosystem with no predictability or consistency, with naming conflicts and incompatible resource schemas. To counteract this potentially chaotic situation, hData calls for the creation of Content Profiles which provide standard section naming and resource schemas for a specific business need or technical capability. For example, pharmacy domain experts have provided a Content Profile for Medication Statements [b-HL7 V3IG MSSP] . Conforming to this Content Profile assures interoperability of medication statements among providers and consumers of this type of information.

Each Certified CapabilityClass (CCC) defines one or more resource types and associated section paths in an hData Content Profile. If an hData service endpoint supports more than one CCC, then its root file will effectively be a union of those extensions and section paths. To create a root file from multiple HCPs, the implementer should copy and combine the information in the example root files from each HCP to create a single root file, creating a combined list of profiles, sections, and resourceTypes. The result is a HDH that combines multiple CCCs.

REST Operations, summarized in Table II-1, are at the heart of hData. There are three types of operations: resource operations, section operations, and base operations, corresponding to the target being a resource (baseURL/sectionPath/resourceID), a section (baseURL/sectionPath), or the base (baseURL), respectively. hData represents a REST binding of a Retrieve, Location, and Updating Service (RLUS). For details, including information on required and optional behaviors and parameters, and return arguments, see [OMG/hData RESTful Trans] the hData RESTful Transport Specification.

Table II-1 – Types of operations

Operation	Operation Description	HTTP Implementation	Requirement
Read	Get the current version of the resource	GET (*resourceURL)*	Required
Version Read	Get a specific version of the resource	GET (*resourceURL*)/history/(*versionId*)	Optional
Update	Update an existing resource	PUT *(resourceURL)*	Optional
Delete	Delete a resource	DELETE *(resourceURL)*	Optional
List	Gets a list of subsections and resources in the section as an ATOM feed	GET (*baseURL or sectionURL*)	Required
Create	Create a new resource or subsection in a section	POST (*baseURL or sectionURL*)	Optional
Batch Create/ Update	Create or update multiple resources in a section	POST (*baseURL or sectionURL*) using Atom feed	Optional
Search	Gets a list of section resources matching the query parameters	GET *(baseURL or sectionURL)/* ?search*(queryString)*	Optional
Validate	Validate a proposed creation action, prior to commit	POST (*sectionURL*)/validate	Optional
Capability Read	Gets root file for capability exchange	GET (*baseURL*)/root	Required
Metadata	Gets service metadata; returns security mechanisms available and a list of supported hData content profiles	GET (*baseURL*)/metadata, optionally OPTIONS (*baseURL*)	Required without prior authentication or authorization
Update Metadata	Replaces metadata on document	POST (*resourceURL)*	Optional

The hData Interoperability framework is depicted in Figure II-1.

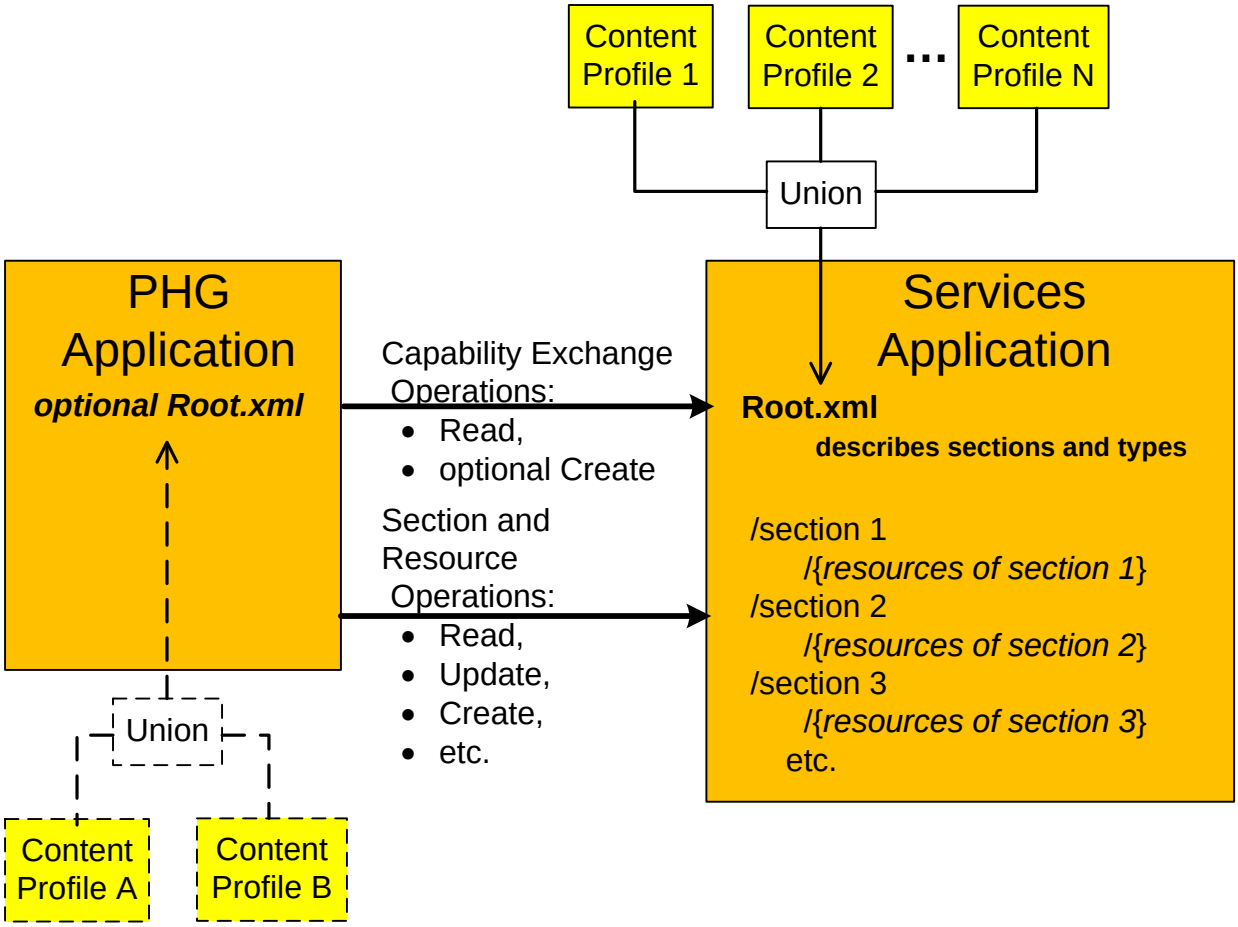

Figure II-1 – hData Interoperability framework

(Note: The PHG Application and Services Application root.xml files are not the same)