INTERNATIONAL STANDARD

ISO/IEC 24791-3

First edition
2014-03-01

Information technology — Radio frequency identification (RFID) for item management — Software system infrastructure —

Part 3: Device management

Technologies de l'information — Identification de radiofréquence (RFID) pour la gestion d'élément — Infrastructure de systèmes logiciels —

Partie 3: Gestion de dispositif

Reference number
ISO/IEC 24791-3:2014(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 24791-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 31, *Automatic identification and data capture techniques*.

ISO/IEC 24791 consists of the following parts, under the general title *Information technology — Radio frequency identification (RFID) for item management — Software system infrastructure*:

— *Part 1: Architecture*

— *Part 2: Data management*

— *Part 3: Device management*

— *Part 5: Device interface*

Introduction

RFID air interface technology is based on non-contact electro-magnetic communication among interrogators and tags. RFID software systems are composed of RFID interrogators, intermediate software systems, and applications that provide control and coordination of air interface operation, tag information exchange, and health and performance management of system components. RFID technology is expected to increase effectiveness in many aspects of business by further advancing the capabilities of Automatic Identification and Data Capture (AIDC). To achieve this goal through the successful adoption of RFID technology into real business environments, RFID devices, software systems, and business applications must provide secure and interoperable services, interfaces, and technologies. This is the goal of the standards defined for RFID Software System Infrastructure (SSI), ISO/IEC 24791.

Information technology — Radio frequency identification (RFID) for item management — Software system infrastructure —

Part 3: Device management

1 Scope

This part of ISO/IEC 24791 defines interfaces for device management of RFID systems. Interfaces are defined that provide for discovery, configuration, initialization and monitoring of RFID systems within the Software System Infrastructure (SSI).

This standard only deals with devices that provide RFID related services. It does not distinguish the form factor of such RFID devices.

This part of ISO/IEC 24791 provides two distinct *interface sets*, one based on the EPCglobal Discovery, Configuration, and Initialization (DCI) standard and the IETF SNMP RFCs and the other based on the Organization for the Advancement of Structured Information Standards (OASIS) Device Profile for Web Services (DPWS) standard. The definition of the Device Profile for RFID will be referred to in this part as the RFID Device Management Profile, or RDMP.

Each interface option set provides interface definitions that provide ISO/IEC 24791-3 Client Endpoints and Services Endpoints with the mechanisms for:

- discovery of the RFID devices and services on a local or remote subnet
- a firmware upgrade service
- a management service that implements configuration related functions
- a monitoring service for reporting alerts, diagnostics, and performance information.

The two interface set definitions provided by this part of ISO/IEC 24791 allow for clients and services endpoints to implement and provide the services based on the specific characteristics of the RFID system to be implemented. Clause 2 defines the Conformance requirements for systems that implement components of one or both of the interface sets.

2 Conformance

This part of ISO/IEC 24791 provides two interface sets; the DCI and SNMP Interface Set and the RDMP interface Set. If a certain implementation conforms to the mandatory functions of at least one of the interface sets, that implementation is conformant to this part of ISO/IEC 24791.

2.1 DCI and SNMP Interface Set

This version of this International Standard divides the DCI capabilities into two *Conformance Groups:*

- Discovery, Configuration, and Initialization Conformance Group

This Conformance Group is defined in Clause 8.2.1. It specifies the protocols and operational procedures that are required for conforming Interrogator Implementations and Device Management Implementations, as defined in this part of ISO/IEC 24791 as well as in ISO/IEC 24791-1.

- Performance Monitoring and Diagnostics Conformance Group

  This Conformance Group is defined in Clause 8.2.2. It specifies the SNMP MIBs that may be implemented by Interrogator Implementations and Data Management Implementations as defined in this part of ISO/IEC 24791 as well as in ISO/IEC 24791-1. Conforming implementations claim conformance to the MODULE_COMPLIANCE statements in the SNMP MIBs appropriate for the particular implementation.

A conforming implementation must implement all of the requirements of each Conformance Group for its particular function in the SSI, but an implementation is not required to claim conformance to either group.

2.2 RDMP Interface Set

This version of the International standard specifies the following device management capabilities in RDMP

- Discovery of devices and hosted services in devices
- A Firmware Upgrade Service to initialize and manage firmware on devices
- A Management service to set and get device configuration and to perform specific device operations, such as reboot
- A monitoring service to monitor the health of a device using events and statistics

A conforming RDMP implementation shall implement DEVICE as defined in DPWS

A conforming RDMP implementation may implement the firmware update service (FUS). If it does implement FUS, it shall implement the mandatory requirements of the firmware update service

A conforming RDMP implementation may implement the management service (MS). If it does implement MS, it shall implement the mandatory requirements of the management service.

A conforming RDMP implementation may implement the monitoring service (MNS). If it does implement MNS, it shall implement the mandatory requirements of the monitoring service.

3 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 19762-1, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary — Part 1: General terms relating to AIDC*

ISO/IEC 19762-3, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary — Part 3: Radio frequency identification (RFID)*

ISO/IEC 24791-5, *Information technology — Radio frequency identification (RFID) for item management — Software system infrastructure — Part 5: Device interface*

Devices Profile for Web Services Version 1.1, OASIS Standard July 2009- http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.pdf.

Control and Provisioning of Wireless Access Points - Protocol Specification - http://www.rfc-editor.org/rfc/rfc5415.txt

EPCglobal, Reader Management Standard, http://www.epcglobalinc.org/standards/rm.

EPCglobal, Discovery, Configuration, & Initialisation Standard for Reader Operations, http://www.epcglobalinc.org/standards/dci.

Internet Engineering Task Force, RFC3418 - Simple Network Management Protocol (SNMP), http://www.faqs.org/rfcs/rfc3418.html

Internet Engineering Task Force, RFC 2011 – SNMPv2 Management Information Base for the Internet Protocol using SMIv2, http://www.faqs.org/rfcs/rfc2011.html

Internet Engineering Task Force, RFC 2863 – The Interfaces Group MIB, http://www.faqs.org/rfcs/rfc2863.html

XML Schema Part 2: Datatypes: http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19762-1, ISO/IEC 19762-3, and the following apply.

4.1
Component
identifiable part of a larger program that provides specific functionality

4.2
Device
RFID Interrogator implementation

4.3
Interface
functions or mechanisms that provide communications to or from a component

4.4
Data management
device functionality that includes or is a combination of reading, writing, collection, filtering, grouping, and event subscription and notification of RFID tag data to higher level applications and interfaces

4.5
Device management
functionality that includes or is a combination of monitoring and control of discovery, configuration, performance and diagnosis of one or more RFID interrogators

4.6
Endpoint
component that implements or exposes an interface to other components or uses the interface of another component

4.7
Implementation
software and hardware that provides the reduction to practice of particular functionality

4.8
Interrogator controller
software capability possibly embodied in a distinct physical device, within the Data Management implementation of the architecture in ISO/IEC 24791-1 and capable of exercising the data, control, and management of interrogators over the device interface defined in ISO/IEC 24791-5

4.9
CLIENT
network endpoint that sends MESSAGEs to and/or receives MESSAGEs from a SERVICE.

4.10
SERVICE
software system that exposes its capabilities by receiving and/or sending MESSAGEs on one or several network endpoints.

4.11
DEVICE
distinguished type of SERVICE that hosts other SERVICEs and sends and/or receives one or more specific types of MESSAGEs.

4.12
HOSTED SERVICE
distinguished type of SERVICE that is hosted by another SERVICE. The lifetime of the HOSTED SERVICE is a subset of the lifetime of its host. The HOSTED SERVICE is visible (not encapsulated) and is addressed separately from its host. Each HOSTED SERVICE has exactly one host. (The relationship is not transitive.)

5 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviated terms given in ISO/IEC 19762-1, ISO/IEC 19762-3, and the following apply.

AC Access Controller

DCI EPCglobal Discovery, Configuration, Initialization Standard

DPWS Devices Profile for Web Services Standard

IETF Internet Engineering Task Force

RFC Request For Comment

RM Reader Management

SNMP Simple Network Management Protocol

SSI Software System Infrastructure

UML Unified Modelling Language

FUS RDMP Firmware Update Service

MS RDMP Management Service

MNS RDMP Monitoring Service

6 Software System Infrastructure Architecture Overview

ISO/IEC 24791-1 defines the architecture for the Software System Infrastructure. The basic relationship among the interfaces and implementations of the Software System Infrastructure is depicted in Figure 1.

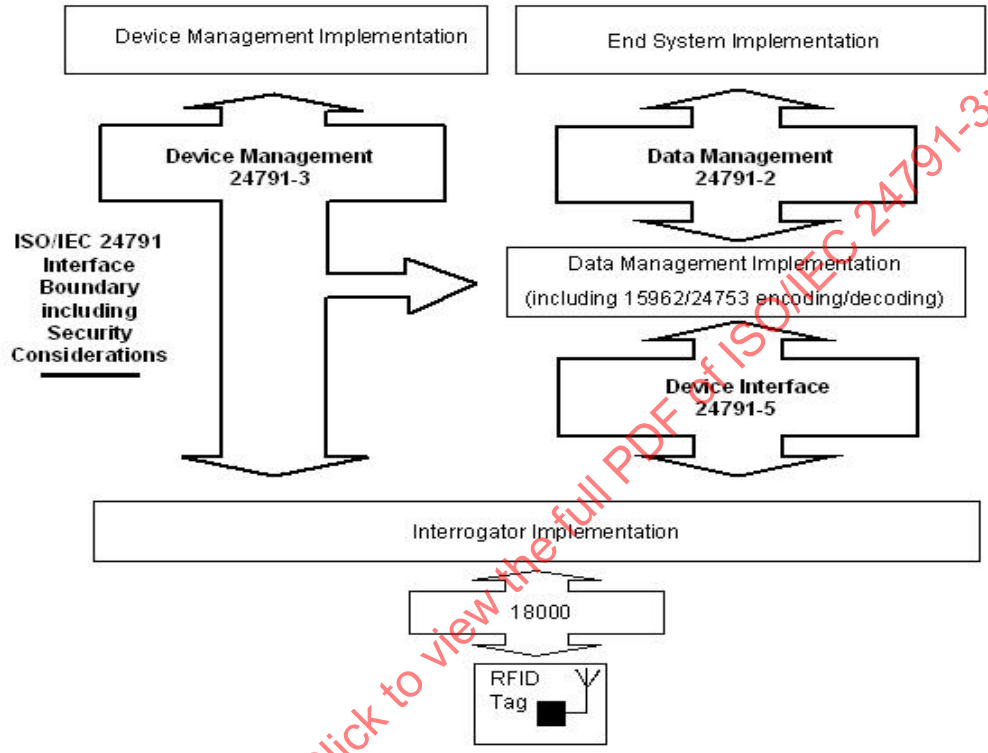

Figure 1 — Architecture Overview including Relationships to Other RFID Standards

The Parts of ISO/IEC 24791 that define Data Management, Device Interface, and Device Management each provide one or more interfaces which allow a client to communicate with a service-providing implementation, either within the same computing device or across a network. These client and service implementations are consistently referred to as Client Endpoints and Services Endpoints, respectively, and in general, the Client Endpoint accesses the capabilities provided by the Services Endpoint. It is the responsibility of the specific standard to define the formats, procedures, operations, and conformance requirements of each interface.

Device Management is concerned with providing discovery, configuration, initialization, performance monitoring, and diagnostics of Software System Infrastructure components and interrogators. As shown in Figure 1, Device Management defines *interfaces* that provide pairwise communications between Interrogator implementations, Data Management implementations, and Device Management Implementations.

In addition to defining interfaces for providing configuration and control of the implementations in the network, Device Management may also define requirements for basic initial operation of interrogators, particularly related to initialization in networked environments. This is necessary in order to achieve the SSI goal of providing scalable deployment and management of large numbers of interrogators in a system.

Although Figure 1 depicts the Device Management Implementation residing outside of the boundary of the SSI, the Device Management Implementation may be implemented within any device in a system. For example, it may reside within a standalone network management application or it may be just one component within a device that is also providing a Data Management Implementation. It may also be one component of an application that is also providing the End System Implementation. As with all other components of the SSI as defined in ISO/IEC 24791 Part 1, the platform on which the standard interfaces are implemented is not important; it is conformance to the interfaces and procedures defined in the ISO/IEC 24791 that is important. Examples of different deployment models of this part of ISO/IEC 24791 are provided in Annex A.

7 UML Modelling

Although Figure 1 provides a general overview of the relationship between the interfaces and implementations in the SSI, Unified Modeling Language (UML) is used for the figures in this document to graphically represent the organization and operation of the Device Management interfaces and implementations so that a precise and common understanding of the relationships among the components can be defined.

UML is a very rich language, but for simplicity only the Physical Diagram subset of the language is used to represent the architecture of the Software System Infrastructure. Physical diagrams, comprised of Component Diagrams and Deployment Diagrams, represent the relationships among the functions and the interfaces provided by the SSI architectural elements as well as how these functions might exist in standards compliant solutions, respectively. Refer to ISO/IEC 24791-1 for a more complete description how UML is used in the part standards of 24791.

8 Device Management

8.1 Architecture

Device Management defines the *interface(s)* that provide discovery, configuration, initialization, performance monitoring, and diagnostics of Software System Infrastructure components and interrogators. Device Management also defines a set of standardized operational procedures that must be executed by conforming devices, typically related to the initial operation of a device in a networked environment.

Specific Device Management interface capabilities are provided by a Device Management Services Endpoint. A Device Management Client Endpoint accesses the Services Endpoint in a component that provides the desired service(s). Figure 2 provides the representation of the Device Management interface in a component:

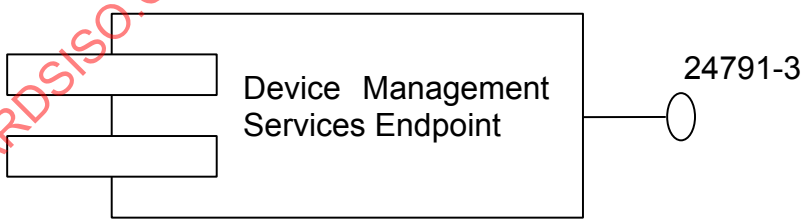

Figure 2 — Device Management Representation

The software programs that provide Device Management Client and Services Endpoints may reside within any of the Implementations that may exist in the SSI, as shown in Figure 1. This part of ISO/IEC 24791 does define requirements on the how the implementations are developed or packaged within computing or network platforms; requirements are only defined for the operation that is provided.

Device Management is distinct from the data and control interfaces provided by other parts of ISO/IEC 24791. It is possible that the implementation of the Device Management interface utilizes the same network interface

as the implementation of one of the data and/or control interfaces in the implementation. It is also possible that for a specific operation or interface, a component may be both a Client and Services Endpoint, essentially resulting in peer-to-peer operation or a negotiated Client/Server relationship. This does not change the architecture defined in this part of ISO/IEC 24791 or in ISO/IEC 24791-1.

The functions covered by Device Management may be grouped and defined as follows:

Discovery: the process of automatically finding components and devices in a system as well as dynamically identifying Service Endpoints and enabling connections between the components and services.

Configuration: the process of setting operational parameters for components that are loaded at system initialization and that change relatively infrequently, primarily through user interaction.

Initialization: the process of providing initial deployment of network and operating parameters for interrogators as well as installing, updating, maintaining software images at desired versions through a dynamic, potentially automated process.

Monitoring: the gathering of statistics and state data useful for determining the historic and current operational state of a component, in particular an interrogator or an SSI component that provides a Data Management implementation function, such as an interrogator controller within the Data Management implementation depicted in Figure 1.

Diagnostics: the mechanism to aid in the detection and isolation of faults or abnormal operation within a component of the Software System Infrastructure. Where the diagnostics involve the computing platform, they are applicable to an interrogator only. Diagnostic capabilities may be defined for other SSI software components, but diagnostic capabilities for general purpose computing platforms will not be defined.

The interfaces defined by this International Standard will provide extension mechanisms to allow implementations to expose management services beyond those specifically defined in this standard. This is consistent with standards-based approaches currently used in the management of telecommunication devices.

It is important to note that not all of the above capabilities are required to be deployed in all implementations of a Device Management Services Endpoint. For example, interrogators may implement and expose a different set of ISO/IEC 24791-3 capabilities from Data Management Implementations. Furthermore, different classes of interrogators may implement and expose different sets of ISO/IEC 24791-3 capabilities. Conformance requirements for implementations of the Device Management Services Endpoint are defined in Clause 2.

9 DCI and SNMP Interface Set

9.1 Discovery, Configuration, and Intialization Conformance Group

9.1.1 General

Conforming devices implement discovery, configuration, and initialization capabilities through the implementation of the protocols and procedures defined in this Conformance Group. This subclause of this International Standard references the EPCglobal Discovery, Configuration, and Initialization (DCI) for Reader Operations standard for the normative requirements for this SSI capability. The EPCglobal DCI standard, references the IETF CAPWAP (Configuration and Provisioning for Wireless Access Points) standard for the core network protocol, security, and communication operations and interfaces.

9.1.2 Interrogator Implementations

Interrogators that conform to this International Standard for discovery, configuration, and initialization capabilities shall implement all requirements, indicated with “shall”, for the *Reader* function as defined in the EPCglobal DCI standard. Conforming implementations may implement any requirements for the Reader function indicated with “may” in the EPCglobal DCI standard.

9.1.3 Device Management Implementations

Device Management Implementations that conform to this part of ISO/IEC 24791 shall implement all requirements, indicated with “shall” for the Access Controller (AC) function as specified in the EPCglobal DCI standard. Conforming implementations may implement any requirements indicated with “may” in the EPCglobal DCI standard.

It is not required that implementations of the Access Controller also implement the *RO Client* function, which is equivalent to the ISO/IEC 24791-5 Device Interface Client functionality, although it is possible and likely that the implementations will be co-resident in computing or network systems. Note that in such cases, the Device Management Implementation and Data Management Implementation from Figure 1 will coexist in the same device. This example is demonstrated in Annex A.

9.2 Performance Monitoring and Diagnostics Conformance Group

9.2.1 General

Performance monitoring and diagnostic information access within of SSI components is provided by Device Management Services Endpoints that expose SNMP MIBs within one or more of the implementations defined in ISO/IEC 24791-1 and illustrated in Figure 1 of this International Standard. SNMP clients (Client Endpoints in the SSI architecture) access the exposed device management information using the Simple Network Management Protocol (SNMP). Implementations claim conformance to one or more MODULE_COMPLIANCE statements within the specific SNMP MIBs normatively referenced in the following subclauses.

Conformance requirements for implementations that expose an SNMP MIB for performance monitoring and diagnostic information access according to this International Standard are defined in the following subclauses. It is not required that an implementation implement or claim conformance to both of the following subclauses if it claims conformance to one of them.

9.2.2 Interrogator Implementations

The EPCglobal Reader Management specification Version 1.0.1 defines an SNMP MIB for performance monitoring and diagnostic information access for Interrogator Implementations.

The MIB groups specified as MANDATORY-GROUPS in the SNMP MODULE-COMPLIANCE statement referenced in the EPCglobal Reader Management Version 1.0.1 specfication shall be implemented by interrogators that claim conformance to this subclause of this International Standard.

Implementation of non-SNMP bindings or transports described in the EPCglobal Reader Management standard is not required by this International Standard.

In addition, the network-attached devices in which the interrogator implementations execute shall implement:

1. The MIB-II System Group, defined in the SNMPv2-MIB module in RFC 3418
2. The MIB-II IP Group, defined in the IP-MIB module in RFC 2011
3. The MIB-II Interfaces Group, defined in the IF-MIB in RFC 2863

9.2.3 Data Management Implementations providing Interrogator Controller Functionality

Annex B of this International Standard provides an SNMP MIB for performance monitoring and diagnostic information access of Data Management Implementations that implement a Device Interface (ISO/IEC 24791-5) Client Endpoint for the control and data access of interrogators. These implementations have been defined as *interrogator controllers*.

The MIB groups specified as MANDATORY-GROUPS in the SNMP MODULE-COMPLIANCE statement in Annex B of this International Standard shall be implemented by interrogator controller functions within Data Management Implementations that claim conformance to this subclause of this document. Note that other functions that may be implemented by a Data Management Implementation, such as the Data Management Services Endpoint, may provide performance monitoring and diagnostic information access by additions the the MIB in Annex B in a future version of this International Standard.

In addition, the network-attached devices in which the data management implementations execute shall implement:

1. The MIB-II System Group, defined in the SNMPv2-MIB module in RFC 3418
2. The MIB-II IP Group, defined in the IP-MIB module in RFC 2011
3. The MIB-II Interfaces Group, defined in the IF-MIB in RFC 2863

10 RDMP Interface Set

10.1 Non Normative Text

Non normative text is formatted as below in the RDMP Interface set.

```
This is non normative text
```

NOTE the Project Editor is aware that this clause 8.3.1 is both incorrect and redundant, now that the text has been reformatted to follow ISO drafting rules for non-normative Notes. During the next ballot, the Project Editor will submit a PE comment to remove this subclause 8.3.1 and renumber the subsequent subclauses.

10.2 XML Namespace

In addition to the namespaces defined in DPWS, this standard defines the following XML namespace.

http://standards.iso.org/iso/24791/-3/2013/01/rdmp

Table 1 lists XML namespaces that are used in this specification. The choice of any namespace prefix is arbitrary and not semantically significant.

Prefix	XML Namespace	Specification(s)
rdmp	http://standards.iso.org/iso/24791/-3/2013/01/rdmp	This specification
dpws	http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01	DPWS
soap	http://www.w3.org/2003/05/soap-envelope	See DPWS
wsa	http://www.w3.org/2005/08/addressing	See DPWS

Table 1 — XML Namespaces

10.3 Device Discovery

A conformant RDMP device shall implement DEVICE, as defined in DPWS.

A conformant RDMP device shall advertise the rdmp: ISO/IEC 24791-3 type in discovery messages.

NOTE DPWS uses WS-Discovery as the device discovery protocol. WS-Discovery defines a multicast discovery protocol. To be discovered by RDMP clients, RDMP devices should implement WS-Discovery as per the requirements of DPWS and as per the additional type requirement defined above. In summary, a RDMP device that implements DEVICE would include dpws:device and rdmp:ISO24791-3 in the Types section of the hello and probe match messages. A transport address may be sent by a device in the hello and probe match messages.

10.4 Device Metadata

10.4.1 General

This part of ISO/IEC 24791 does not specify requirements for exchanging device metadata in addition to those already specified in DPWS.

NOTE 1 DPWS defines a standard mechanism for retrieving device metadata from a device. Metadata would include information such as manufacturer name, model name, firmware version etc. This mechanism is documented in the Description Section in the DPWS spec. We describe it briefly here for illustration.

NOTE 2 An RDMP client interested in getting metadata about a RDMP device would send a SOAP envelope containing a WS-Transfer Get message to the transport address of the chosen device. The RDMP device would then send a WS-Transfer GetResponse message containing the device metadata.

NOTE 3 Please refer to the DPWS specification, section titled **Description** for more details and requirements.

10.4.2 Service Discovery

An RDMP conformant device may advertise services that are not specified in this standard in the dpws:Relationship/dpws:Host/dpws:Types

NOTE 1 In addition to device discovery, DPWS specifies mechanisms for discovery of HOSTED SERVICES on a device. Some examples of HOSTED SERVICES are a stock quote service, firmware update service, a print service, a calendar service etc. An RDMP client discovers a HOSTED service by parsing the Metadata section of the WS-Transfer GetResponse.

An example response that advertises a Printer Service is below:

```
<?xml version="1.0" encoding="utf-8"?>
<wsoap12:Envelope
      xmlns:wsoap12="http://www.w3.org/2003/05/soap-envelope"
      xmlns:mex="http://schemas.xmlsoap.org/ws/2004/09/mex"
      xmlns:wsa="http://www.w3.org/2005/08/addressing"
      xmlns:dpws="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01"
>
...
<mex:MetadataSection
Dialect="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/Relationship">
    <dpws:Relationship Type="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/host">
          <dpws:Hosted>
               <wsa:EndpointReference>
                    <wsa:Address>
      http://192.168.0.101:80/SamplePrintService0
                    </wsa:Address>
               </wsa:EndpointReference>
               <dpws:Types
                    xmlns:spt="http://example.com/wsdp/sample/print">
                    spt:PrinterServiceType
               </dpws:Types>
               <dpws:ServiceId>
```

```
                    http://example.com/sample/print/PrintService
                </dpws:ServiceId>
            </dpws:Hosted>
        </dpws:Relationship>
</mex:MetadataSection>
</wsoap12:Envelope>
```

An example for advertising ISOIEC24791-5 (LLRP) service by RDMP devices:-

```
<mex:MetadataSection
Dialect="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/Relationship">
    <dpws:Relationship Type="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/host">
            <dpws:Hosted>
                <wsa:EndpointReference>
                    <wsa:Address>                                    ISOIEC24791-
5://192.168.0.101:5084/                    </wsa:Address>
                </wsa:EndpointReference>
                <dpws:Types
                    ISOIEC24791-5
                </dpws:Types>
                <dpws:ServiceId>
                    http://example.com/sample/ISO24791-5/
                </dpws:ServiceId>
            </dpws:Hosted>
        </dpws:Relationship>
</mex:MetadataSection>
```

NOTE 2 Note that, the types sent in the dpws:Relationship/dpws:Host/dpws:Types element are the portTypes of services that are supported by this device. In contrast, the types element in WS-Discovery messages contains discovery-layer portTypes that are implemented by the device.

10.5 Firmware Update Service

10.5.1 General

This part of ISO/IEC 24791 describes a Firmware Update Service that is used to initialize and update firmware for RFID devices. The client is responsible for initiating a firmware update. The device downloads the firmware from the location presented by the client.

10.5.2 Firmware Update Service state machine

The Firmware Update Service on the device shall maintain the states as described in the firmware update state machine. Figure 3 illustrates the state machine for a firmware update service.

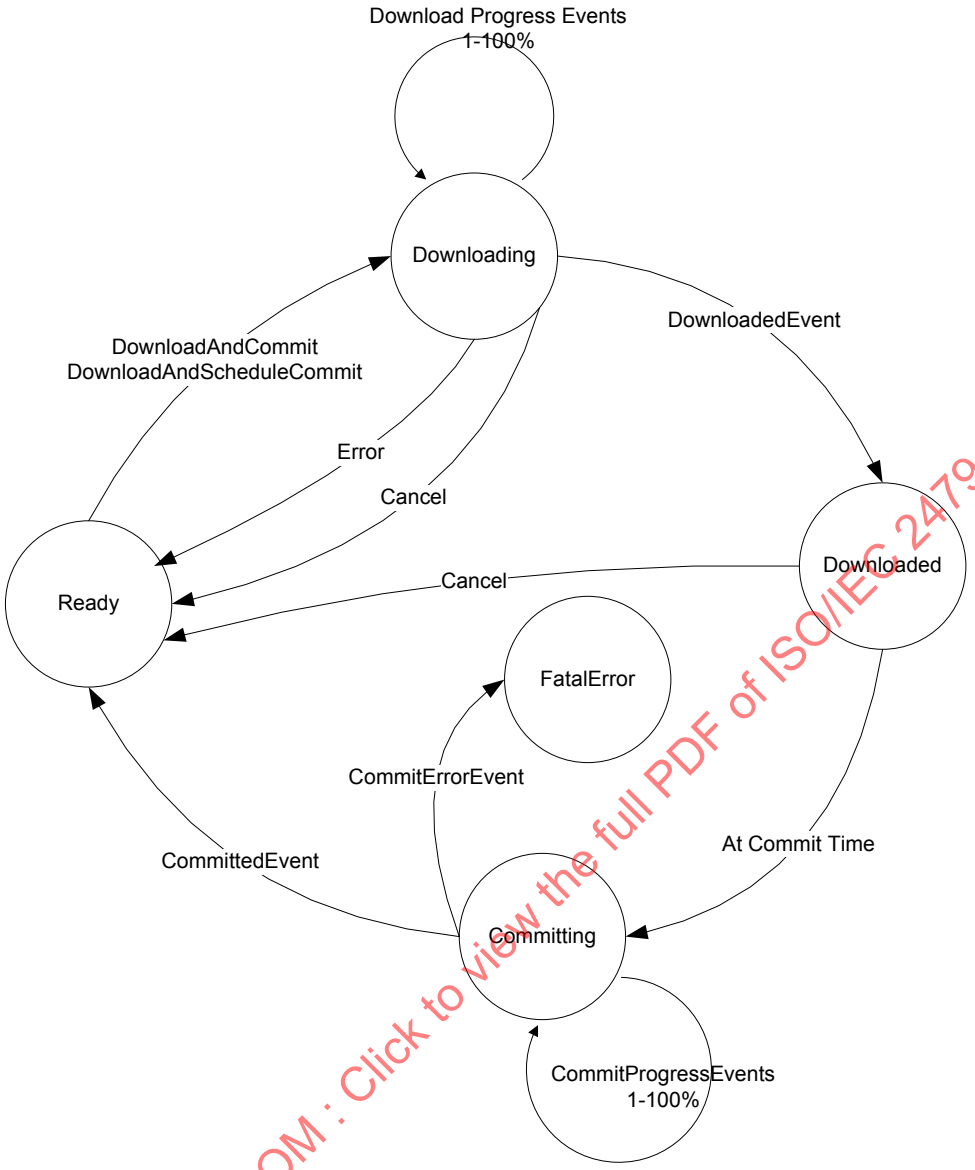

Figure 3 — Service State Machine

Firmware States

Ready - The ready state is the initial state. The firmware service state is ready if the service is not downloading or committing firmware currently.

Downloading – The service is currently downloading firmware to the device.

Downloaded: Firmware was successfully downloaded to the device. The firmware may be immediately committed or scheduled for a commit based on the client request.

Committing - Firmware is currently being applied to the target.

Fatal Error - There was an error during commit of firmware and the RF module is now unusable.

10.5.3 FUS Operations

10.5.3.1 OperationNotPermittedInCurrentState

Any of the operations defined in FUS may return the **OperationNotPermittedInCurrentState** fault. This fault is sent when the FUS is not in one of the allowed states for this operation.

Table 2 lists the details for an operation fail in current state.

Field	Value
[Code]	soap:Receiver
[Subcode]	rdmp:OperationNotPermittedInCurrentState
[Reason]	The service is not in one of the allowed states for this operation.
[Detail]	*Current state of the service. Allowed states for this operation.*

Table 2 — Fault Details for Operation Fail

10.5.3.1.1 DownloadandCommit

10.5.3.1.2 General

This method shall be supported by the FUS.

If this operation is invoked in a state other than Ready, the FUS shall return OperationNotPermittedinCurrentState error. Once the FUS responds successfully to this operation, it shall move into the Downloading state.

NOTE It is best practice to invoke CheckFirmwareApplicability before invoking DownloadandCommit method

10.5.3.1.3 Request Elements [Method Parameters for DownloadandCommit]

10.5.3.1.4 FirmwareLocation

The URL from where the firmware is available for download.

The FUS shall support downloading firmware using HTTP. The FUS may support downloading firmware using HTTPS.

If the URL has an unsupported scheme, the FUS shall return an UnsupportedFirmwareLocationScheme fault. If the URL is invalid for other reasons, the FUS may return the InvalidRequestElement fault. The [Reason] element may have additional text to explain why the URL is invalid. Table 3 lists the details for the Firnware location.

Field	Value
[Code]	soap:Sender
[Subcode]	rdmp:UnsupportedFirmwareLocationScheme
[Reason]	The supplied FirmwareLocation has an unsupported URI scheme.
[Detail]	*<SupportedUriSchemes>* *<!-- One or more supported URL schemes-->* *<SupportedUriScheme>HTTP</SupportedUriScheme>* *<SupportedUriScheme>HTTPS</SupportedUriScheme>* *...* *</SupportedUriSchemes>*

Table 3 — Details for FirmwareLocation

Faults defined in this method may be generated by the FUS just by examining the FirmwareLocation URL syntactically. FUS provides another mechanism to address other errors such as the FirmwareLocation being unreachable. Since the firmware download is attempted in the Downloading state, if download fails, the FUS shall move into the Ready state. The error may be recorded by the FUS and reported in a GetStatus operation.

NOTE 1 The device may use HTTP Range headers to control the rate at which the download happens. The heuristics used to determine the size of the range and timing of (partial) HTTP(S) GET requests is left to the device.

NOTE 2 In case of error prone connections, range headers may be used to resume a download. It is the device's responsibility to determine the resume location in the firmware update file.

When the FUS does a partial HTTP GET, it shall use “bytes" as the Range Unit.

NOTE 3 As best practice, this standard recommends the following:- The HTTP server that serves firmware download requests should support partial HTTP GET using the Range header. It should support “bytes” as a Range Unit. It is the responsibility of the HTTP Server serving the firmware download request to specify and implement any policies with respect to the valid duration of the URL.

10.5.3.2 Post Download

Once download is completed successfully, the FUS may raise a DownloadedEvent.

Once download is completed successfully, the FUS shall move into the Downloaded state.

The FUS is recommended to validate the firmware binary prior to committing. The mechanism used for validation is out of scope for this document. If the binary is found to be invalid, the FUS may record the error and report this in a GetStatus operation.

The FUS is expected to commit the firmware immediately once the download is complete.

NOTE The HTTP Server for FUS may be implemented in the same host as the RDMP client or hosted on a remote server.

10.5.3.3 DownloadAndScheduleCommit

10.5.3.3.1 General

This method may be supported by the FUS.

If this operation is invoked in a state other than Ready, the FUS shall return OperationNotPermittedinCurrentState error. Once the FUS responds successfully to this operation, it shall move into the Downloading state.

10.5.3.3.2 Request Elements [Input Parameters]

10.5.3.3.2.1 FirmwareLocation

Please see the DownloadandCommit section for description of requirements.

10.5.3.3.2.2 AtTime

This is the date & time at which to commit the loaded firmware. If the AtTime is in the past, the FUS shall return an InvalidRequestElement error. Additional text may be added to the [Reason] element of the SOAP fault.

10.5.3.3.2.3 PostDownload

Once download is completed successfully, the FUS may raise a DownloadedEvent.

Once download is completed successfully, the FUS shall move into the Downloaded state.

The FUS is recommended to validate the firmware binary prior to committing. The mechanism used for validation is out of scope for this document. If the binary is found to be invalid, the FUS may record the error and report this in a GetStatus operation.

10.5.3.4 CheckFirmwareApplicability

10.5.3.4.1 General

This method requests the device firmware update service to checks if specified firmware is applicable for the device. This method shall be supported by the FUS.

10.5.3.4.2 Request Elements [Method Parameters]

10.5.3.4.2.1 FirmwareVersionToCompare

The FirmwareVersion is the version of the firmware that the client wishes to compare with the committed firmware version on the FUS.

This part of ISO/IEC 24791 does not mandate any specific requirements for the FirmwareVersion string. Implementations are recommended to construct a string that enables the device to determine

a) If the firmware being supplied is applicable to the FUS

b) If the firmware being supplied is newer or older

10.5.3.4.3 Response Elements

The method returns one of the following

Newer – firmware is applicable to FUS and is more recent than the currently installed version

Older - firmware is applicable to FUS and is less recent than the currently installed version

SameVersion - firmware is applicable to FUS and is the same version as the currently installed version

NotApplicable - firmware is not applicable for the device or the supplied string is invalid or the FUS is unable to parse the string

10.5.3.5 GetStatus

10.5.3.5.1 General

This method returns current state of firmware update service on the device.This method shall be supported by the FUS.

10.5.3.5.2 Response Elements

The response shall contain the following field

State – one of Ready, Downloading, Downloaded, Committing, or FatalError

The response may contain the following fields:

1. % downloaded - If state is Downloading, the FUS may return the % of firmware downloaded.
2. % committed - If state is committing, the FUS may return the % committed.
3. Error: The FUS may persist an error string from a previous operation or error situation.
4. TimeofError: the time at which the last error occurred.
5. Description – A localizable string. It applies to any state.

NOTE Let us say that while committing firmware there is an error and the service goes into the Ready state. The error may be stored in the service to be retrieved later by the client. The error string may also be used to diagnose errors encountered by the FUS.

10.5.3.6 Cancel

10.5.3.6.1 General

This method may be supported by the FUS.

10.5.3.6.2 Cancel Description

If this operation is invoked in the Ready state, it may be ignored. If the FUS is in the Downloading or Downloaded states, it shall move to the Ready state. In the Committing state, the FUS may reject the Cancel operation by returning the CancelNotPermittedNow error. Table 4 lists the details of the cancel method.

Field	Value
[Code]	soap:Receiver
[Subcode]	rdmp: CancelNotPermittedNow
[Reason]	Cancel operation not permitted at this point.
[Detail]	None

Table 4 — Details of Cancel Method

10.5.3.7 Events from FUS

10.5.3.7.1 General

The FUS shall implement eventing as defined in the DPWS requirements for eventing. Please see section titled **Eventing** in the DPWS specification.

10.5.3.7.2 DownloadProgressEvent

10.5.3.7.2.1 General

DownloadProgressEvents may be generated only in the Downloading state.

10.5.3.7.2.2 PercentCompleted

Percentage of download completed may be provided with the DownloadProgressEvent by the service. If provided, this shall be an integer between 0 and 100.

10.5.3.7.3 DownloadedEvent

DownloadedEvent may be generated when the service transitions from the Downloading state to the Downloaded state.

10.5.3.7.4 CommitProgressEvent

10.5.3.7.4.1 General

CommitProgressEvents may be generated only in the Committing state.

10.5.3.7.4.2 PercentCompleted

Percentage of update completed may be provided with the CommitProgressEvent by the service. If provided, this shall be an integer between 0 and 100

10.5.3.7.4.3 Description

This is a descriptive localizable string about the status of the commit.

10.5.3.7.5 CommittedEvent

CommittedEvent may be generated when the service transitions from the Committing state to the Ready state.

10.5.3.7.6 CommitErrorEvent

10.5.3.7.6.1 General

If there is an error during firmware commit, it may be reported using the CommitErrorEvent.

10.5.3.7.6.2 Description

This is a localizable string containing a description of what went wrong during the firmware commit.

10.5.3.7.6.3 Time

This is the time at which this particular error took place.

10.5.3.7.7 DownloadErrorEvent

If there is an error during firmware download, it may be reported using the DownloadErrorEvent.

10.6 Management Service (MS)

10.6.1 General

This section describes a Management Service (MS). The management service provides the following functionality:-

a. Configuring a device with standard properties. In this document, we include the ability to configure ISO/IEC 24791-5 properties on the device.

b. Implementing common operations like device reboot and reset to factory settings.

The MS specifies a set of configuration properties that are commonly understood among RFID device implementations. The MS also allows for custom configuration properties to be retrieved and applied on devices. The MS also provides a mechanism to advertise metadata for configuration properties.

Each configuration property has a unique identifier and a value associated with the identifier.

10.6.2 Property Identifier

A property identifier has two parts:-

1. **Group URI** - (example: http://standards.iso.org/24791-3/rdmp/Configuration/RF). The Group URI acts as a namespace.

2. **Property Name** - (example: "Power Level")

A Property Name shall be of type token as defined in XML Schema Part 2.

Token: token represents tokenized strings. The value space of token is the set of strings that do not contain the carriage return (#xD), line feed (#xA) nor tab (#x9) characters, that have no leading or trailing spaces (#x20) and that have no internal sequences of two or more spaces. The lexical space of token is the set of strings that do not contain the carriage return (#xD), line feed (#xA) nor tab (#x9) characters, that have no leading or trailing spaces (#x20) and that have no internal sequences of two or more spaces.

A property identifier used by MS shall NOT have null value for either the group URI or the Property Name

10.6.3 Property Value Metadata

Metadata of the property value is made up of the following:-

- **Description** – Friendly description of the property. It is of type string. A MS service shall include description in the metadata.
- **Writeable** – A property is writeable if it may be modified on the device. It is of type Boolean. MS may include this field in the metadata. If this is not specified, the client may assume the property to be writeable.
- **Persisted** – A property is persisted if the device stores the property in permanent storage. . It is specified as a Boolean. MS may include this in the metadata. If this is not specified, the client may assume the property to be persisted.
- **RestartRequired** - Does device requires a restart for the change in the property to take effect – This is specified as a Boolean. MS may include this in the metadata. If this is not specified, the client may assume that restart is not required.
- **Type** – the type of the property, indicating if the property is of the type string, int, boolean, or some other complex type. The MS shall include the type in the metadata. The type shall be a XML QName, as defined in the XML Schema as defined in XML Schema Part 2: Datatypes.
- **DefaultValue** – The value that the device uses if the client has not modified. It may be included. If included, it shall conform to the Type specified in the metadata.

NOTE The type metadata is critical for clients to interpret and process configuration data sent by the device.

10.6.4 Device Property Profile

A property profile is a collection of properties.

NOTE Using a collection of properties is useful in many scenarios. Management software may want to bring the device to a base configuration, when it is first discovered. When a device changes role, role specific configuration may need to be applied to the device. Using a property profile enables device and clients to store a collection of properties together.

10.6.5 Sources and Source Types

Sources in RDMP may refer to the device or the antennas or physical ports connected to the device. The standard source types are antenna, GPI, GPO and Barcode.

10.6.6 MS Operations

10.6.6.1 GetSources

10.6.6.1.1 General

This method is invoked by the client to get the list of sources on the device and the types of the sources. This method shall be supported by MS.

10.6.6.1.2 Request Elements

None

10.6.6.1.3 Response Elements

This method returns a collection of the following tuples. Each tuple shall have

SourceName – of type token, as defined in XML Schema

Type – Source Type

10.6.6.2 GetAllPropertyMetadata

10.6.6.2.1 General

This method shall be supported by MS. This is the mechanism for a client to obtain a list of supported properties and property metadata.

NOTE GetAllPropertyMetadata is the key method for a client to discover device and source related configuration. It is recommended that the Management Service provide comprehensive metadata information about all the properties configurable via RDMP. Metadata enables the management software to correctly display configuration (example: a property may be shown disabled in the user interface if it is not writeable). It also enables management software to warn the user to reboot the device if the property change requires a restart to take effect.

10.6.6.2.2 Request Elements

SourceName – this is of type string. This field is optional.

If the SourceName parameter is not specified, this method shall return the metadata associated with all the properties of the device.

If the SourceName parameter is specified, this method shall return the metadata associated with all the properties of the specified source.

10.6.6.2.3 Response Elements

The method returns a collection of property identifiers and property value metadata.

10.6.6.3 GetPropertyMetadata

10.6.6.3.1 General

This method may be supported by MS.

10.6.6.3.2 Request Elements

This method takes Property Identifier and a source name. The SourceName is an optional parameter.

10.6.6.3.3 Response Elements

If the SourceName parameter is not specified, this method shall return the property metadata associated with the specified property of the device.

If the SourceName parameter is specified, this method shall return the property metadata associated with the specified property of the source.

10.6.6.4 GetAllPropertyValues

10.6.6.4.1 General

This method may be supported by MS.

10.6.6.4.2 Request Elements

SourceName – an optional parameter

10.6.6.4.3 Response Elements

If the SourceName parameter is not specified, this method shall return a property profile of the device.

If the SourceName parameter is specified, this method shall return a property profile of the source.

10.6.6.5 GetPropertyValue

10.6.6.5.1 General

This method may be supported by the MS.

10.6.6.5.2 Request Elements

This method takes an optional SourceName and a PropertyIdentifier as input.

10.6.6.5.3 Response Elements

This method returns a PropertyValue.

If the SourceName parameter is not specified, this method shall return the property value associated with the specified property of the device.

If the SourceName parameter is specified, this method shall return the property value associated with the specified property of the source.

10.6.6.5.4 Errors

If the property is not supported by the MS, it shall return rdmp:PropertyNotSupported fault as defined below. Table 5 lists the details of an error.

Field	Value
[Code]	soap:Sender
[Subcode]	rdmp:PropertyNotsupported
[Reason]	The requested property is not supported.
[Detail]	*PropertyIdentifier*

Table 5 — Error Details

10.6.6.6 SetPropertyProfile

10.6.6.6.1 General

This method shall be supported by MS.

10.6.6.6.2 Request Elements

This method takes a PropertyProfile as input and an optional SourceName parameter.

If the SourceName parameter is not specified, this method shall set the property profile on the device.

If the SourceName parameter is specified, this method shall set the property profile on the source.

10.6.6.6.3 Response Elements

None

10.6.6.6.4 Errors

If the property is not writeable, the MS shall return rdmp:PropertyNotWriteable fault as defined below. Table 6 lists the details of an error.

Field	Value
[Code]	soap:Sender
[Subcode]	rdmp:PropertyNotWriteable
[Reason]	The requested property is not writeable.
[Detail]	*PropertyIdentifier*

Table 6 — Error Details

If the MS service was unable to apply all properties, the MS shall return rdmp:UnableToApplyAllProperties fault as defined below. MS may include more details about the error in the Reason. The Detail may contain the PropertyProfile of the actual Properties MS as shown in Table 7 was unable to apply.

Field	Value
[Code]	soap:Sender
[Subcode]	rdmp:UnableToApplyAllProperties
[Reason]	*The reason for being unable to apply the property.*
[Detail]	*PropertyProfile*

Table 7 — Error Details

10.6.6.7 GetPropertyValuesByGroup

10.6.6.7.1 General

This method may be implemented by MS.

10.6.6.7.2 Request Elements

This method takes an xs:anyURI as GroupURI and an optional parameter SourceName.

10.6.6.7.3 Response Elements

If the SourceName parameter is not specified, this method shall return a property profile for the specified property group of the device.

If the SourceName parameter is specified, this method shall return a property profile for the specified property group of the source.

10.6.7 Standard Properties

The following group names and properties as shown in Table 8 shall be supported by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/configuration/General

Property Name	Type	R/W (Read Only or Read-Write)	Applicable Targets (Antenna, Device)	Notes
Name	token	R/W	A,D	Friendly Name of the source/device
ReaderRegulatoryRegion	token	R	D	Legal regulatory region of the device

Table 8 —General Configuration Properties

The following RF properties as shown in Table 9 may be supported by the by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/configuration/RF

Property Name	Type	R/W	Applicable Targets	Notes
AirProtocolsSupported	tokens	R	A, D	Protocols (tag types) supported by this device/source
PowerLevel	Float	W	A	A % that designates the antenna's power setting for reading tags.
OperationEnvironment	token	R	A,D	Reader operation mode. Example values: single, multiple and dense

Table 9 —RF Configuration Properties

The following group names and properties as shown in Table 10 may be supported by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/configuration/ISO24791-5

NOTE RDMP clients are allowed to configure certain properties and capabilities of 24791-5 conformant devices. The reason we want to allow this is to enable management software to set default configuration values that are not specific to a LLRP client session. One cannot expect LLRP clients to set all default values every time they connect. In terms of capabilities, the ability to get this information by RDMP clients will help enterprise management software provide data for reporting applications. It is to be noted that when a LLRP client sets a value during a session and the RDMP client sets the same value, the new value does not take effect until a reboot is performed.

The following 24791-5 properties (capabilities and configuration) may be supported by the Management Service.

Property Name	**Type**	**R/W**	**Applicable Targets**	**Notes**
LLRPStatus	LLRPStatus	R	D	
GeneralDeviceCapabilities	GeneralDeviceCapabilities	R	D	
RegulatoryCapabilities	RegulatoryCapabilities	R	D	
Antenna Properties	AntennaProperties	R/W	A	
Antenna Configuration	AntennaConfiguration	R/W	A	
KeepaliveSpec	KeepaliveSpec	R/W	D	
EventsAndReports	EventsAndReports	R/W	D	
Identification	Identification	R	D	
LLRPServers	anyURI	R/W	D	The URI specified used this property should have the URI scheme ISOIEC24791-5. Example: **ISOIEC24791-5://192.168.0.101:5555/**

Table 10 —ISO 24791-5 Properties

The following RDMP properties as shown in Table 11 may be supported by the by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/Configuration/General

Property Name	Type	R/W	Applicable Targets	Notes
Location	String	W	A,D	Location of the source
Description	String	W	A,D	
TimeServer	String	W	D	Time server used by the device for time syncs
Role	String	W	D	Role of the device
DeviceEPC	String	R	D	Manufacturer assigned EPC URI
TimeServers	tokens	W	D	Time servers used by the device for time syncs
DHCPServers	Tokens	W	D	DHCP server currently used by the device for DHCP.
Contact	String	W	D	The textual identification of the contact person for this device, together with information on how to contact this person.

Table 11 —General Properties

The following RDMP properties as shown in Table 12 may be supported by the by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/Configuration/RF

Property Name	Type	R/W	Applicable Targets	Notes
Frequency	Double	W	A	Frequency used by the source in MHz
EffectiveRange	Float	R	A	Range of antenna in metres
AntennaSequence	tokens	W	D	Sequence of antennas for reading. The order is important.

Table 12 —RF Properties

The following RDMP properties as shown in Table 13 may be supported by the by the Management Service.

GroupName URI: http://standards.iso.org/24791-3/rdmp/Configuration/Notification

Property Name	Type	R/W	Applicable Targets	Notes
EventMode	Boolean	W	D	Boolean to indicate if the device can send notifications to the host

Table 13 —Notification Properties

10.6.8 Other Management Operations

10.6.8.1 Reboot

10.6.8.1.1 General

By invoking this method, a client initiates reboot of a device. This method shall be supported by MS.

10.6.8.1.2 Request Elements

None

10.6.8.1.3 Response Elements

None

10.6.8.2 ResetToFactorySettings

10.6.8.2.1 General

Using this method, the client instructs the device to reset any settings such as configuration to factory specified defaults. This method shall be supported.

10.6.8.2.2 Request Elements

None

10.6.8.2.3 Response Elements

None

10.6.8.3 ResetToFactorySettingsExceptNetwork

10.6.8.3.1 General

Using this method, the client instructs the device to reset any settings such as configuration to factory specified defaults EXCEPT the network settings.

This method may be supported.

10.6.8.3.2 Request Elements

None

10.6.8.3.3 Response Elements

None

10.7 Operation Error Reporting

10.7.1 General

Mechanisms and artifacts defined in this section are used by operations in all the services defined in this document to communicate errors to the client.

SOAP faults are the defined mechanism to communicate errors. Fault messages are correlated as replies using the [relationship] property as defined in Web Services Addressing 1.0 – Core [*WS-Addressing Core*] recommendation.

10.7.2 Common Operation Error Codes

10.7.2.1 General

Operations defined in the specification may return any of the faults defined in [Common Operation Faults]. In addition, operations may define additional SOAP faults.

10.7.2.2 Action Not Supported

When an optional operation of any service defined in this specification is invoked by a client but is not supported by a service it shall generate a wsa:ActionNotSupported fault. The fault is defined in [Web Services Addressing 1.0 – SOAP Binding].

10.7.2.3 InvalidRequestElement

This fault is sent when a Client sends an invalid request element as part of an operation. Table 14 lists the details of the error message.

Field	Value
[Code]	soap:Sender
[Subcode]	rdmp: InvalidRequestElement
[Reason]	At least one request elements is invalid
[Detail]	*Name of the invalid request element.*

Table 14 —InvalidRequestElement Description

10.7.2.4 InternalError

This fault is sent when the service encounters an unexpected condition such as out of memory that prevented it from fulfilling the request. Additional text may be appended to the [Reason] element to provide more information about the internal error. Table 15 lists the details of the error message.

Field	Value
[Code]	soap:Receiver
[Subcode]	rdmp: InternalError
[Reason]	The service had an unexpected error
[Detail]	*None*

Table 15 —InternalError Description

10.8 Monitoring Service (MNS)

10.8.1 General

The monitoring service (MNS) is used to monitor the health of the device. The monitoring service publishes events and statistics to enable RDMP monitoring clients to actively monitor the device health.

10.8.2 Monitoring Event Structure

Each monitoring event has the following structure.

Fields: Each event contains these standard fields

- **TimeofOccurrence**: Date and time that the event was generated. This is of type xs:datetime. The TimeofOccurrence field shall be implemented by MNS.
- **Level**: Critical, Error, Warning or Informational. It is of type string that allows only the above values. Level indicates severity of the event. Level shall be implemented by the MNS.
- **Description:** a localizable string. Describes the event in friendly terms to the user. This may be implemented.

An event may have additional fields. Additional fields are described in specific events below.

10.8.3 Events

The following events may listed in Table 16 be supported and sent by MNS.

EventName	Additional Fields	Type	Description
BatteryEvent	% of battery remaining – this field shall be included for the Battery Event	Integer	The service shall send a battery event at Informational level upto 15% of battery left. Between 14- 10% remaining, it shall send a warning level event. Between 9 - 5%, it shall send an error level event. Less than 5% it shall raise a Critical level event. Within the same battery level, the frequency of

			sending events is not defined in this standard.
MemoryEvent	% of free memory remaining on device - this field shall be included for the Memory Event	Integer	The service shall send a memory event at Informational level upto 15% of memory left. Between 14- 10% remaining, it shall send a warning level event. Between 9 - 5%, it shall send a error level event. Less than 5% it shall send a Critical level event. Within the same memory level, the frequency of sending events is not defined in this standard
SourceStateEvent	SourceName – this field shall be included	String	Indicates the state of the source. Level shall be Error when down, shall be Informational when Up
	State – values are Up or Down. This field shall be included	enum	
SourceNoiseLevelEvent	SourceName - this field shall be included	String	
	Noiselevel - This field shall be included	integer	
PowerSupplyEvent (applies to devices that support battery backup)	State – values are Up/Down. This field shall be included	enum	Down event raised when power supply to device is cut. Up event when power is restored. Level – shall be Warning when down, shall be Informational when Up
PrinterEvent A printer event, in addition to the standard fields contains an error code and an error string. The possible error codes are :-			
PrinterEvent	ErrorCode		
	rdmp:CutterFault rdmp:GapNotFound rdmp:Jam rdmp:OutOfMedia rdmp:NoCurrentTemplate rdmp:RibbonFault	QName	
TagOperationalEvent A Tag Operation event, in addition to the standard fields may contain an error code a TagInfo structure and an error string. The possible error codes are:-			
TagOperationalEvents	ErrorCode	Type	

	rdmp:TagLocked	QName	Locked when accessing the tag
	rdmp:TagLockFailed		
	rdmp:InvalidPasscode		
	rdmp:PartiallyLocked		
	rdmp:ReadOnly		Trying to write to a region that is read-only
	rdmp:CorruptTag		
	rdmp:WriteFailed		
	rdmp:KillFailed		
	rdmp:FailedLock		
	rdmp:FailedErase		
	rdmp:FailedMemoryRead		
	rdmp:FailedRead		
HardwareErrorEvent	This event consists of an error code in addition to the standard fields. Error code is vendor dependent. Error code shall be a QName.		

Table 16 —Event Details

10.8.4 Statistics

10.8.4.1 General

The MNS supports reporting of the following statistics information.

10.8.4.2 GetStatisticsMetadata

10.8.4.2.1 General

This method shall be supported by the MNS

10.8.4.2.2 Request Elements

SourceName – this is of type string. This field is optional.

If the SourceName parameter is not specified, this method shall return the metadata associated with all the statistics of the device.

If the SourceName parameter is specified, this method shall return the metadata associated with all the statistics of the specified source.

10.8.4.2.3 Response Elements

This method shall return a collection of StatisticsIdentifier and StatisticsMetadata.

10.8.4.3 GetStatistics

10.8.4.3.1 General

This method shall be supported by the MNS.

10.8.4.3.2 Request Elements

This method takes an optional SourceName as input.

10.8.4.3.3 Response Elements

This method shall return a StatisticsProfile, which is a collection of StatisticsIdentifier and StatisticsValue.

The GroupName URI for the identifiers used in this service shall be: http://standards.iso.org/24791-3/rdmp/Monitoring/Statistics

Vendors extending statistics shall NOT use this group name for custom statistics.

10.8.4.4 ResetStatistics

10.8.4.4.1 General

This method may be supported by the MNS

10.8.4.4.2 Request Elements

This method takes an optional SourceName as input.

10.8.4.4.3 Response Elements

None

10.8.4.5 Standard Statistics

Table 17 lists the standard statistics details.

Name	Property Value Type	Remarks
BatteryPowerRemaining	Positive integer in the range of 0-100	Percentage of battery power remaining. Not resettable
BatteryTimeRemaining	Positive integer in the range of 0-100	Remaining time remaining in minutes. Not resettable
DeviceUptimeTicks	Positive double	Time in ticks for which the device has been up. Not resettable.
FailedKillCount	Positive integer	
FailedLockCount	Positive	

	integer	
FailedReadCount	Positive integer	
FailedWriteCount	Positive integer	
KillCount	Positive integer	
LockCount	Positive integer	
ReadCount	Positive integer	
WriteCount	Positive integer	
MemoryReadCount	Positive integer	
EraseCount	Positive integer	
FailedEraseCount	Positive integer	
PrintedCount	Positive integer	
FailedPrintedCount	Positive integer	
TimeSinceLastTagNotification	Integer, in minutes	Enables troubleshooting of why no tag reads for a duration.

Table 17 —Standard Statistics

10.9 Security

Parties involved in message exchanges defined in this specification fall into the following categories

1. CLIENT: To elaborate this category covers both clients that invoke operations on the device and clients that act as an Event Sink i.e. receive events from the device.

2. DEVICE

3. Firmware Server

Security between CLIENT and DEVICE is covered in the Security section of [DPWS]. Firmware is downloaded from the Firmware Server as explained in FUS. This communication may be secured by using a secure transport such as HTTPS.

NOTE Security encompasses

a) Authentication of device and client: Authentication is the process by which the identity of the sender is determined by the recipient. A client may need to authenticate the devices it can communicate with or vice versa.

b) Integrity and confidentiality of messages exchanged between device and client.

Briefly, transport Layer Security may be used to provide mutual authentication of client and device as well as the establishment of a secure channel over which messages are exchanged. Integrity and Confidentiality are achieved by using the secure channel. The policy assertions carried in the messages exchanged during discovery may contain the client security requirements as well as the security protocols supported by client and device for authentication and establishment of a secure channel. Since Discovery precedes setting up of a secure channel, the integrity of discovery messages is protected using message-level signatures, while the integrity of other messages is protected using a secure channel

10.10 Extensibility

10.10.1 General

Messages defined by this specification may include constructs to be extensible by conforming implementations. Clients or devices that do not understand a custom extension to a type may choose to ignore it.

The XML/XSD notation for types shall be:-

```
<xs:complexType name="PropertyMetadata">
       <xs:sequence>
           <xs:element name="Type" type="xs:QName" minOccurs="1" maxOccurs="1"
nillable="false"/>
           <xs:element name="Description" …/>
           …
           <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
       </xs:sequence>
       <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

The following example shows an extension to PropertyMetadata that conforms to the above schema. The extension adds a new element called lowerBound.

```
<PropertyMetadata xmlns="http://schemas.xmlsoap.org/ws/2008/rdmp"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <Type>xs:int</Type>
  <Description>some description</Description>
   …
  <lowerBound xmlns="http://www.example.org/rdmpextensions">5</lowerBound>
</PropertyMetadata>
```

10.10.2 Extending Monitoring events

RDMP implementations of the Monitoring service may choose to implement custom events outside of the standard specification. This part of ISO/IEC 24791 provides a way to describe a custom event in order for any client to understand and interpret the event.

The XML/XSD notation for extension events shall be:

```
<xs:complexType name="VendorDefinedEvent">
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
       <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

A similar notation shall be followed for extending statistics for the monitoring service

NOTE 1 Management Service and Monitoring Service can be extended to support custom types as values for properties and statistics. The following example shows how the management service can be extended to support a custom type for properties. Similar changes can be done to monitoring service schema to support custom types for Statistics.In the Management Service schema, within the PropertyValue complex Type, add the text that is in bold:

```
<xs:complexType name="PropertyValue">
    <xs:choice>
      <xs:element name="String" type="xs:string" />
      <xs:element name="Integer" type="xs:int" />
      <xs:element name="TagDataSelector" type="tns:TagDataSelector" />
      <xs:element name="QName" type="xs:QName" />
      <xs:element name="Boolean" type="xs:boolean"/>
```

```
        <xs:element name="Float" type="xs:float"/>
        <xs:element name="Double" type="xs:double"/>
        <xs:element name="Uri" type="xs:anyURI"/>
        <xs:element name="DateTime" type="xs:dateTime"/>
    <xs:element name="NewComplexType" type="myns:NewComplexType"/>
      </xs:choice>
  </xs:complexType>
```

Where “myns” is the prefix defined for the namespace containing the “NewComplexType” xml type.

NOTE 2 RDMP services can be extended to support custom defined events. The following example shows how to add a custom event to Firmware Service.

To add a new event for FUS called FirmwareUpdate_CustomEvent, do the following:

1. In the FirmwareUpdate wsdl message definition section, add this:

```
<wsdl:message name="FirmwareUpdate_CustomEvent_OutputMessage">
    <wsdl:part name="parameters" element="myns:CustomEvent"/>
</wsdl:message>
```

2. In the port binding section of the FirmwareUpdate wsdl, add the following:

```
<wsdl:operation name="CustomEvent">
<wsdl:output wsaw:Action="<Custom Event Namespace>/CustomEvent"
  message="tns:FirmwareUpdate_ CustomEvent _OutputMessage"/>
</wsdl:operation>
```

3. In the soap binding section of the FirmwareUpdate wsdl, add the following:

```
<wsdl:operation name=" CustomEvent ">
            <soap12:operation soapAction="http://customnanespace/CustomEvent"
style="document"/>
     <wsdl:output>
        <soap12:body use="literal"/>
     </wsdl:output>
</wsdl:operation>
```

Where myns is the prefix of the namespace containing the xml schema definition of “CustomEvent”.

Annex A
(informative)

Implementation Examples

A.1 General

There are many possible organizations of the implementations of the software components referenced in this International Standard, including how the components are distributed among different physical computing platforms. This Annex provides informative examples of the different organizations utilizing UML deployment diagrams.

The figures in this Annex do not show any of the data or control interfaces of ISO/IEC 24791 that may be implemented on the devices because the implementation-specific logic provides any interaction between the interfaces, making them functionally independent. In certain deployment scenarios, the instances of the Device Interface or Data Management Services Endpoint could receive commands or requests from the Device Management Implementation logic in response to commands or requests from the Device Management Services Endpoint. These requests would be for configuration, provisioning, performance monitoring, and/or diagnostics, and *not* for tag or sensor data as defined in this International Standard.

A.2 DCI and SNMP Interface Set Implementation Examples

Figure A1 illustrates a node that implements the Device Management Services Endpoint in an interrogator. In this example Services Endpoint would conform to Clause 8.2.1.2 by implementing discovery, configuration, and initialization capabilities as defined in this International Standard, which references IETF RFC 5415 (CAPWAP) and EPCglobal DCI. Also illustrated in this example, the Services Endpoint would conform to Clause 8.2.1.3 by implementing monitoring and diagnostics capabilities as defined in this International Standard which references the EPCglobal Reader Management SNMP MIB and the IETF MIBII MIBs.

The separation between the Services Endpoint and the Implementation in the Figure A.1 illustrates how the standards-based interfaces are logically distinct from the implementation that provides the management data or achieves the specified device management result.

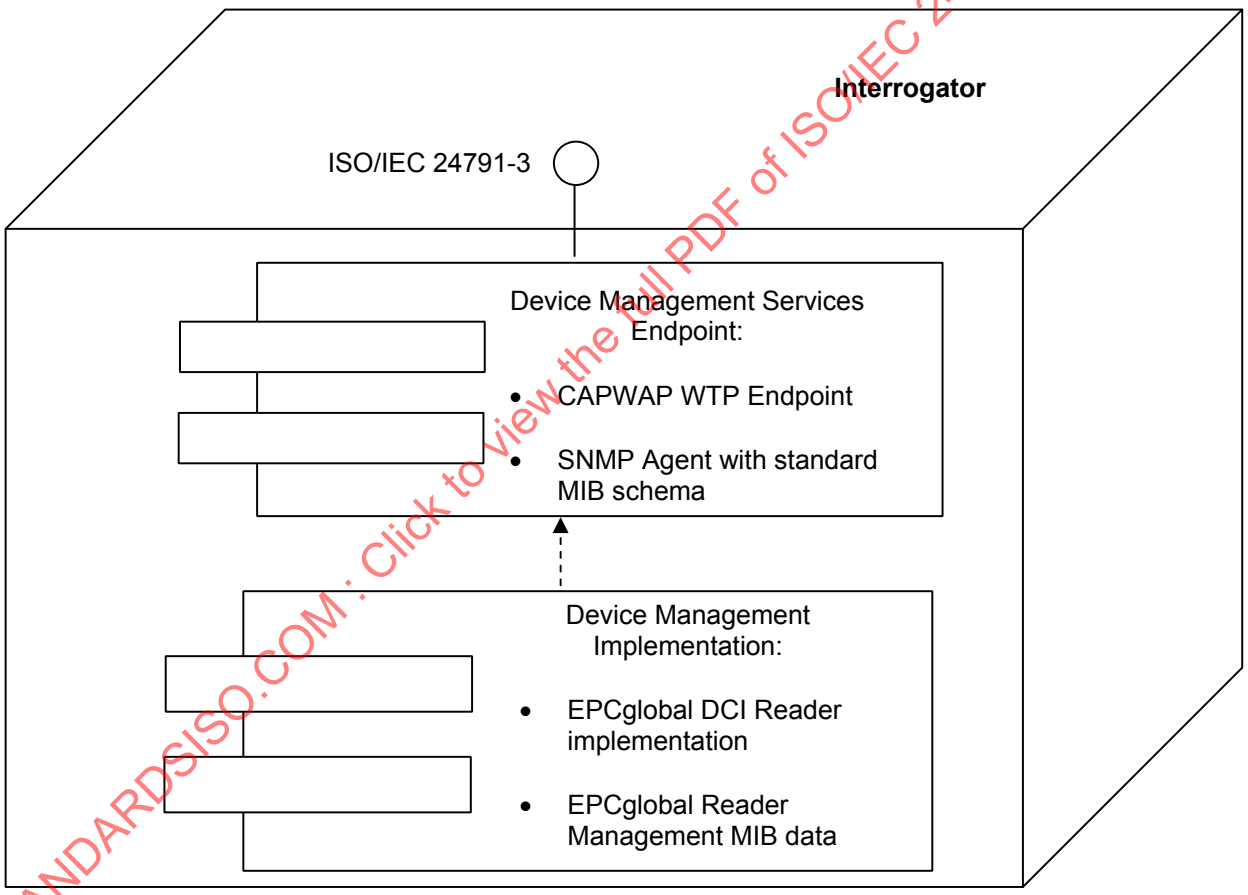

Figure A.1 — Node Diagram with DCI and SNMP Interface Set Implementation on Interrogator

Figure A.2 illustrates nodes in a system that implements the Device Management Endpoints on an interrogator controller as well as on interrogators. The interrogator controller is one example of a Data Management Implementation as shown in Figure 1 of this document. Other examples of deployment models providing conforming implementations are possible.

In this example, the interrogators provide the same services as in the example of Figure A.1.

The Services Endpoint on the interrogator would conform to Clause 8.2.1.2 by implementing discovery, configuration, and initialization capabilities as defined in this International Standard, which references IETF RFC 5415 (CAPWAP) and the EPCglobal DCI Specification. The DCI Access Controller provides a Services Endpoint to the interrogators, which, as defined in the specific standards, provide peer-level services back to the interrogator controller.

Also illustrated in this example, the Services Endpoint would conform to Clause 8.2.1.3 by implementing monitoring and diagnostics capabilities as defined in this International Standard and the referenced IETF MIBII MIBs.

Figure A.2 shows the Node Diagram with ISO/IEC 24791-3 Endpoints on Interrogator Controller and Interrogators In this deployment diagram (Figure A.2), there is no Device Management Client endpoint on the interrogator controller to request monitoring and diagnostic Device Management services from the interrogators. While this is possible, in this deployment example, a management application would access the performance monitoring and diagnostic data directly using the SNMP MIBs referenced in Annex B of this part of ISO/IEC
24791.

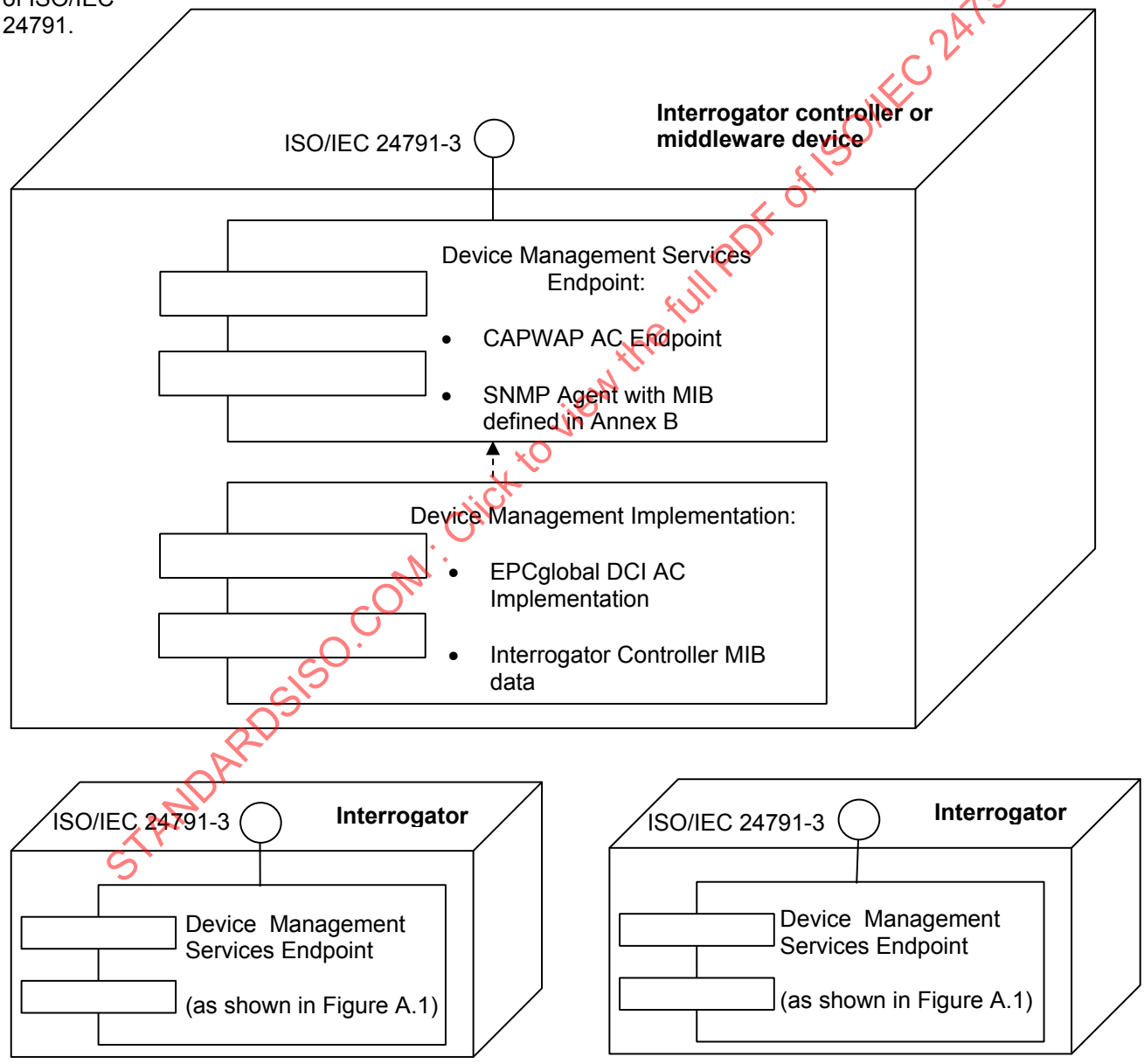

Figure A.2 — Node Diagram with ISO/IEC 24791-3 Endpoints on Interrogator Controller and Interrogators

Annex B
(normative)

SSI Device Management MIB

```
ISO-24791-3-MIB DEFINITIONS ::= BEGIN
--
-- ISO/IEC 24791-3 MIB for Software System Infrastructure Device Management
--

IMPORTS
   Counter32, IpAddress
      FROM SNMPv2-SMI
   DisplayString, TimeStamp
      FROM SNMPv2-TC
   MODULE-IDENTITY
      FROM SNMPv2-SMI
   iso
     FROM SNMPv2-SMI;

rfidSsiDeviceManagementMIB MODULE-IDENTITY
   LAST-UPDATED "201212122100Z"
   ORGANIZATION "ISO"
   CONTACT-INFO
      "Convener, ISO WG4/SG1"
   DESCRIPTION
      "This module defines a MIB that represents that manageable entities
       described in the RFID Software System Infrastructure International
       Standard ISO/IEC 24791, specific Part 3, Device Management"
   ::= { iso(1) standard(0) iso24791(24791) part3(3) }


rfidSsiDeviceManagementObjects      OBJECT IDENTIFIER   ::= { rfidSsiDeviceManagementMIB 0 }
rfidSsiDeviceManagementConformance  OBJECT IDENTIFIER   ::= { rfidSsiDeviceManagementMIB 1 }


rfidSsiDeviceManagementDescription OBJECT-TYPE
   SYNTAX  DisplayString (SIZE (0..255))
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "A textual description of this RFID Network."
   ::= { rfidSsiDeviceManagementObjects 1 }

rfidSsiDeviceManagementLocation OBJECT-TYPE
   SYNTAX  DisplayString (SIZE (0..255))
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "A textual identification of the location of this RFID Network.  This may
      be a GLN or other structured identifier represented in the URI format."
   ::= { rfidSsiDeviceManagementObjects 2 }


--
-- RFID Interrogator Controller (RIC) infrastructure representation MIB in
-- the SSI Device Management MIB tree.  The RFID Interrogator Controller
-- is a function that requests ISO/IEC 24791-5 Device Interface services
```

```
-- to one or more interrogators.
--
-- It is expected that more than one RIC will be deployed in an RFID Network to
-- provide for redundancy, load balancing, and scaling.  This is represented
-- by the ricControllerTable that may contain more than one RIC.
--

ricMIB OBJECT IDENTIFIER ::= { rfidSsiDeviceManagementObjects 10 }

ricNotifications OBJECT IDENTIFIER ::= { ricMIB 0 }
ricObjects       OBJECT IDENTIFIER ::= { ricMIB 1 }


ricControllerTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF RicControllerEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A table of information about the RICs in the RFID Network."
    ::= { ricObjects 1 }


ricControllerEntry OBJECT-TYPE
    SYNTAX  RicControllerEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "An entry containing information about a particular RIC."
    INDEX   { ricControllerName }
    ::= { ricControllerTable 1 }

RicControllerEntry ::=
    SEQUENCE {
        ricControllerName
            DisplayString,
        ricControllerIpAddr
            IpAddress,
        ricControllerDescription
            DisplayString
    }

ricControllerName OBJECT-TYPE
    SYNTAX  DisplayString
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A administrative name or IP hostname for this RIC. Because this is used as the
   index for this table, it is mandatory, but it is not necessary that the value is
   a proper IP hostname to be resolved.  If it is not, another mechanism must be
   used to provide a valid value for ricControllerIpAddr"
    ::= { ricControllerEntry 1 }

ricControllerIpAddr OBJECT-TYPE
    SYNTAX  IpAddress
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The main IP address of this RIC that will be used for
        communication from outside the RFID Network, possibly resolved from
   ricControllerName. The interrogators in the RFID Network will connect to this address,
        but it is also possible that the readers will connect to another
        address on the RIC.  The IF-MIB on the RIC should be used
        for access to the full set of interfaces."
    ::= { ricControllerEntry 2 }

ricControllerDescription OBJECT-TYPE
```

```
    SYNTAX  DisplayString
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A textual description for this RIC."
    ::= { ricControllerEntry 3 }



--
--  Notifications for the ricMIB
--

ricStartup NOTIFICATION-TYPE
    STATUS      current
    DESCRIPTION
   "A notification for an RFID Network Controller startup event"
    ::= { ricNotifications 1 }


ricShutdown NOTIFICATION-TYPE
    STATUS      current
    DESCRIPTION
   "A notification for an RFID Network Controller shutdown event "
    ::= { ricNotifications 2 }

--
-- RFID RIC Interrogator representation MIB in the SSI Device Management MIB
--
-- This MIB provides the representation of the interrogator from the
-- the viewpoint of an RFID Interrogator Controller.  The MIB for an
-- interrogator within SSI Device Management is provided by the EPCglobal
-- Reader Management MIB as described in ISO/IEC 24791-3.
--

ricInterrogatorMIB OBJECT IDENTIFIER ::= { rfidSsiDeviceManagementObjects 11 }

ricInterrogatorNotifications OBJECT IDENTIFIER ::= { ricInterrogatorMIB 0 }
ricInterrogatorObjects    OBJECT IDENTIFIER    ::= { ricInterrogatorMIB 1 }


ricInterrogatorTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF RicInterrogatorEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A table of information about the Interrogators in an RFID Network as
         viewed by an Interrogator Controller."
    ::= { ricInterrogatorObjects 1 }

ricInterrogatorEntry OBJECT-TYPE
    SYNTAX  RicInterrogatorEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "An entry containing information about a particular interrogator."
    INDEX   { ricInterrogatorName }
    ::= { ricInterrogatorTable 1 }

RicInterrogatorEntry ::=
    SEQUENCE {
        ricInterrogatorName
            DisplayString,
        ricInterrogatorIpAddr
            IpAddress,
        ricInterrogatorRicIpAddr
```

```
        IpAddress,
    ricInterrogatorDescription
        DisplayString,
    ricInterrogatorConnectionAdminStatus
        INTEGER,
    ricInterrogatorConnectionOperStatus
        INTEGER,
    ricInterrogatorConnectionTime
        TimeStamp,
    ricInterrogatorFirmwareVersion
        DisplayString,
    ricInterrogatorAggregateReadReqs
        Counter32,
    ricInterrogatorAggregateTagReads
        Counter32,
  ricInterrogatorLastTagReadTime
        Gauge32,
    ricInterrogatorAggregateGpiEvents
        Counter32,
    ricInterrogatorLastGpiEventTime
        Gauge32,
    ricInterrogatorErrorMessage
        DisplayString

  }

ricInterrogatorName OBJECT-TYPE
   SYNTAX  DisplayString
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "A administrative name or IP hostname for this interrogator. Because this is used as the
   index for this table, it is mandatory, but it is not necessary that the value is
   a proper IP hostname to be resolved.  If it is not, another mechanism must be
   used to provide a valid value for ricInterrogatorIpAddr."
    ::= { ricInterrogatorEntry 1 }

ricInterrogatorIpAddr OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "The IP address of the interrogator that is used for the network connection
      between the RIC and the interrogator."
   ::= { ricInterrogatorEntry 2 }

ricInterrogatorRicIpAddr OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "The IP address of the RIC that is connected to this interrogator and is
      responsible for control of the interrogator in the RFID Network.  This
      RIC IP address may not be the primary RIC IP address in the
      ricControllerEntry if the controlling RIC has more than one interface."
   ::= { ricInterrogatorEntry 3 }

ricInterrogatorDescription OBJECT-TYPE
   SYNTAX  DisplayString
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "A textual description for this interrogator."
   ::= { ricInterrogatorEntry 4 }

ricInterrogatorConnectionAdminStatus OBJECT-TYPE
```

```
  SYNTAX    INTEGER {
                unknown(0),
                enabled(1),
                disabled(2)
                }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The desired administrative status of the connection between
        the RIC and the interrogator."
    ::= { ricInterrogatorEntry 5 }

ricInterrogatorConnectionOperStatus OBJECT-TYPE
    SYNTAX    INTEGER {
                  unknown(0),
                  connected(1),
                  disconnected(2)
                  }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The current operational status of the connection between
        the RIC and the interrogator."
    ::= { ricInterrogatorEntry 6 }

ricInterrogatorConnectionTime OBJECT-TYPE
    SYNTAX  TimeStamp
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The amount of time the current RIC/interrogator connection has been up."
    ::= { ricInterrogatorEntry 7 }

ricInterrogatorFirmwareVersion OBJECT-TYPE
    SYNTAX  DisplayString
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The firmware version running on this interrogator as retrieved from
        the interrogator by the RIC.  If the interrogator is unable to communicate
        its firmware version over the network protocol, the value of
        'Unknown' will be displayed."
    ::= { ricInterrogatorEntry 8 }

ricInterrogatorAggregateReadReqs OBJECT-TYPE
    SYNTAX  Counter32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Aggregated read requests (sum across all read points) from the
        RIC to the interrogator."
    ::= { ricInterrogatorEntry 9 }

ricInterrogatorAggregateTagReads OBJECT-TYPE
    SYNTAX  Counter32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Aggregated tag reads (sum across all read points) received by
        the RIC from the interrogator."
    ::= { ricInterrogatorEntry 10 }

ricInterrogatorLastTagReadTime OBJECT-TYPE
    SYNTAX  Gauge32
    ACCESS  read-only
    STATUS  mandatory
```

```
    DESCRIPTION
        "Last tag read time of any read point received from this interrogator."
    ::= { ricInterrogatorEntry 11 }


ricInterrogatorAggregateGpiEvents OBJECT-TYPE
    SYNTAX  Counter32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Aggregated General Purpose Input events (sum across all inputs) received by
        the RIC from the interrogator."
    ::= { ricInterrogatorEntry 12 }

ricInterrogatorLastGpiEventTime OBJECT-TYPE
    SYNTAX  Gauge32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Last General Purpose Input event time of any input received from this interrogator."
    ::= { ricInterrogatorEntry 13 }


ricInterrogatorErrorMessage OBJECT-TYPE
    SYNTAX  DisplayString
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Error message providing information if interrogator operational state is
   not connected."
    ::= { ricInterrogatorEntry 14 }



--
--  Notifications for the interrogatorMIB
--

ricInterrogatorConnected NOTIFICATION-TYPE
    OBJECTS { ricInterrogatorIpAddr }
    STATUS      current
    DESCRIPTION
  "A notification to be sent when the RIC connects to a interrogator."
    ::= { ricInterrogatorNotifications 1 }


ricInterrogatorDisconnected NOTIFICATION-TYPE
    OBJECTS { ricInterrogatorIpAddr }
    STATUS      current
    DESCRIPTION
  "A notification to be sent when the RIC loses a connection to
        a interrogator."
    ::= { ricInterrogatorNotifications 2 }


--
-- RFID RIC Read Point representation MIB in the SSI Device Management MIB
--
-- This MIB provides the representation of read points (necessary connected
-- to RFID interrogators) from the viewpoint of an RFID Interrogator
-- Controller.  The MIB for an interrogator within SSI Device Management which
-- provide much more detail about read point operations is provided by the
-- EPCglobal Reader Management MIB as described in ISO/IEC 24791-3.
--

ricReadPointMIB OBJECT IDENTIFIER ::= { rfidSsiDeviceManagementObjects 12 }
```

```
ricReadPointObjects OBJECT IDENTIFIER ::= { ricReadPointMIB 1 }


ricReadPointTable OBJECT-TYPE
   SYNTAX  SEQUENCE OF RicReadPointEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION
      "A table of information about the Read-Points in an RFID Network.
      A Read-Point is one of the antennas on an RFID interrogator
      represented in the ricInterrogatorTable of the rfidSsiDeviceManagementMIB."
   ::= { ricReadPointObjects 1 }

ricReadPointEntry OBJECT-TYPE
   SYNTAX  RicReadPointEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION
      "An entry containing information about a particular read-point."
   INDEX   { ricReadPointInterrogatorName, ricReadPointIndex }
   ::= { ricReadPointTable 1 }

RicReadPointEntry ::=
   SEQUENCE {
      ricReadPointInterrogatorName
         DisplayString,
      ricReadPointIndex
         INTEGER,
      ricReadPointAdminStatus
         INTEGER,
      ricReadPointOperStatus
         INTEGER,
      ricReadPointReadReqs
         Counter32,
      ricReadPointTagReads
         Counter32,
  ricReadPointErrorMessage
         DisplayString
   }

ricReadPointInterrogatorName OBJECT-TYPE
   SYNTAX  DisplayString
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "The administrative name or IP hostname of the interrogator associated with this read-point.
      This is the ricInterrogatorName from the interrogatorTable in the RFID Network MIB."
   ::= { ricReadPointEntry 1 }

ricReadPointIndex OBJECT-TYPE
   SYNTAX  INTEGER
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
      "A unique index for this read-point.  This value may be assigned
      by the RIC or by the interrogator and communicated to the RIC. "
   ::= { ricReadPointEntry 2 }

ricReadPointAdminStatus OBJECT-TYPE
   SYNTAX  INTEGER {
               unknown(0),
               enabled(1),
               disabled(2)
               }
   ACCESS  read-only
```

```
    STATUS  mandatory
    DESCRIPTION
        "The desired admin status of this read-point on the interrogator."
    ::= { ricReadPointEntry 3 }

ricReadPointOperStatus OBJECT-TYPE
    SYNTAX  INTEGER {
                unknown(0),
                up(1),
                down(2)
                }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The current operational status of this read-point on the interrogator."
    ::= { ricReadPointEntry 4 }

ricReadPointReadReqs OBJECT-TYPE
    SYNTAX  Counter32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of read requests issued for this read-point by the RNC
        to the interrogator."
    ::= { ricReadPointEntry 5 }

ricReadPointTagReads OBJECT-TYPE
    SYNTAX  Counter32
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of tags read by this read-point and reported to the RNC
        by the interrogator."
    ::= { ricReadPointEntry 6 }

ricReadPointErrorMessage OBJECT-TYPE
    SYNTAX  DisplayString
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Error message if available if read-point operational state is down."
    ::= { ricReadPointEntry 7 }


--
-- Conformance information
--

rfidSsiDeviceManagementCompliances OBJECT IDENTIFIER ::= { rfidSsiDeviceManagementConformance 1 }
rfidSsiDeviceManagementGroups      OBJECT IDENTIFIER ::= { rfidSsiDeviceManagementConformance 2 }

--
-- Compliance statements
--

rfidSsiDeviceManagementCompliances MODULE-COMPLIANCE
    STATUS  current
    DESCRIPTION
          "The compliance statements for RFID SSI Device Management entities."
    MODULE
        MANDATORY-GROUPS { rfidSsiDeviceManagementNotificationGroup,
                           rfidSsiDeviceManagementSsiGroup,
                           rfidSsiDeviceManagementControllerGroup,
                           rfidSsiDeviceManagementInterrogatorGroup
                         }
    ::= { rfidSsiDeviceManagementCompliances 1 }
```

```
--
-- Units of Conformance
--

rfidSsiDeviceManagementNotificationGroup NOTIFICATION-GROUP
   NOTIFICATIONS { ricStartup,
                   ricShutdown,
                   ricInterrogatorConnected,
                   ricInterrogatorDisconnected
                 }
   STATUS    current
   DESCRIPTION
         "The group of notifications about state changes in the SSI."
   ::= { rfidSsiDeviceManagementGroups 1 }

rfidSsiDeviceManagementSsiGroup OBJECT-GROUP
   OBJECTS   { rfidSsiDeviceManagementDescription,
               rfidSsiDeviceManagementLocation }
   STATUS    current
   DESCRIPTION
         "The group of data representing the overall SSI."
   ::= { rfidSsiDeviceManagementGroups 2 }

rfidSsiDeviceManagementControllerGroup OBJECT-GROUP
   OBJECTS   { ricControllerIpAddr,
               ricControllerName,
               ricControllerDescription }
   STATUS    current
   DESCRIPTION
         "The group of data representing status of an RFID interrogator
         controller."
   ::= { rfidSsiDeviceManagementGroups 3 }

rfidSsiDeviceManagementInterrogatorGroup OBJECT-GROUP
   OBJECTS   { ricInterrogatorName,
               ricInterrogatorIpAddr,
               ricInterrogatorName,
               ricInterrogatorDescription,
               ricInterrogatorConnectionAdminStatus,
               ricInterrogatorConnectionOperStatus,
               ricInterrogatorConnectionTime,
               ricInterrogatorFirmwareVersion,
               ricInterrogatorAggregateReadReqs,
               ricInterrogatorAggregateTagReads,
      ricInterrogatorAggregateGpiEvents
      ricInterrogatorLastGpiEventTime
      ricInterrogatorErrorMessage
               ricReadPointInterrogatorName,
               ricReadPointIndex,
               ricReadPointAdminStatus,
               ricReadPointOperStatus,
               ricReadPointReadReqs,
               ricReadPointTagReads,
      ricReadPointErrorMessage
            }
   STATUS    current
   DESCRIPTION
         "The group of data representing an RFID interrogator as
         from the perspective of an interrogator controller."
   ::= { rfidSsiDeviceManagementGroups 4 }


END
```

Annex C
(normative)

RDMP WSDLs and XSDs

C.1 Device WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:wsdp="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01"
  xmlns:wsp="http://www.w3.org/ns/ws-policy"
  xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"
  targetNamespace="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
  name="WSDRFIDDevice">
    <wsp:Policy wsu:Id="DevicePolicy">
        <wsdp:Profile />
    </wsp:Policy>
    <wsdl:portType name="ISO24791-3" />
</wsdl:definitions>
```

C.2 Firmware Update Service WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions
  targetNamespace="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
  xmlns:tns="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <wsdl:types>
    <xsd:schema targetNamespace="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp">
      <xsd:include schemaLocation="FirmwareUpdate.xsd"/>
    </xsd:schema>
  </wsdl:types>

  <wsdl:message name="FirmwareUpdate_DownloadAndCommit_InputMessage">
    <wsdl:part name="parameters" element="tns:DownloadAndCommit"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadAndCommit_OutputMessage">
    <wsdl:part name="parameters" element="tns:DownloadAndCommitResponse"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadAndScheduleCommit_InputMessage">
    <wsdl:part name="parameters" element="tns:DownloadAndScheduleCommit"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadAndScheduleCommit_OutputMessage">
    <wsdl:part name="parameters"
element="tns:DownloadAndScheduleCommitResponse"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_GetStatus_InputMessage">
```

```
    <wsdl:part name="parameters" element="tns:GetStatus"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_GetStatus_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetStatusResponse"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_CheckFirmwareApplicability_InputMessage">
    <wsdl:part name="parameters" element="tns:CheckFirmwareApplicability"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_CheckFirmwareApplicability_OutputMessage">
    <wsdl:part name="parameters"
element="tns:CheckFirmwareApplicabilityResponse"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_Cancel_InputMessage">
    <wsdl:part name="parameters" element="tns:Cancel"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_Cancel_OutputMessage">
    <wsdl:part name="parameters" element="tns:CancelResponse"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_CommittedEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:CommittedEvent"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadedEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:DownloadedEvent"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_CommitProgressEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:CommitProgressEvent"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_CommitErrorEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:CommitErrorEvent"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadErrorEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:CommitErrorEvent"/>
  </wsdl:message>
  <wsdl:message name="FirmwareUpdate_DownloadProgressEvent_OutputMessage">
    <wsdl:part name="parameters" element="tns:DownloadProgressEvent"/>
  </wsdl:message>

  <wsdl:portType name="FirmwareUpdate">
    <wsdl:operation name="DownloadAndCommit">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndCommit"
message="tns:FirmwareUpdate_DownloadAndCommit_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndCommitResponse"
message="tns:FirmwareUpdate_DownloadAndCommit_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="DownloadAndScheduleCommit">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndScheduleCommit"
message="tns:FirmwareUpdate_DownloadAndScheduleCommit_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndScheduleCommitResponse"
message="tns:FirmwareUpdate_DownloadAndScheduleCommit_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetStatus">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/GetStatus"
message="tns:FirmwareUpdate_GetStatus_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/GetStatusResponse"
message="tns:FirmwareUpdate_GetStatus_OutputMessage"/>
```

```
    </wsdl:operation>
    <wsdl:operation name="CheckFirmwareApplicability">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CheckFirmwareApplicability"
message="tns:FirmwareUpdate_CheckFirmwareApplicability_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CheckFirmwareApplicabilityResponse"
message="tns:FirmwareUpdate_CheckFirmwareApplicability_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="Cancel">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/Cancel"
message="tns:FirmwareUpdate_Cancel_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CancelResponse"
message="tns:FirmwareUpdate_Cancel_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="CommittedEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommittedEvent"
message="tns:FirmwareUpdate_CommittedEvent_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="DownloadedEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadedEvent"
message="tns:FirmwareUpdate_DownloadedEvent_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="CommitProgressEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommitProgressEvent"
message="tns:FirmwareUpdate_CommitProgressEvent_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="CommitErrorEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommitErrorEvent"
message="tns:FirmwareUpdate_CommitErrorEvent_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="DownloadErrorEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadErrorEvent"
message="tns:FirmwareUpdate_DownloadErrorEvent_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="DownloadProgressEvent">
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadProgressEvent"
message="tns:FirmwareUpdate_DownloadProgressEvent_OutputMessage"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:binding name="DefaultBinding_FirmwareUpdate" type="tns:FirmwareUpdate">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="DownloadAndCommit">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndCommit" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="DownloadAndScheduleCommit">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadAndScheduleCommit" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetStatus">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/GetStatus" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CheckFirmwareApplicability">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CheckFirmwareApplicability" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Cancel">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/Cancel" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CommittedEvent">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommittedEvent" style="document"/>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DownloadedEvent">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadedEvent" style="document"/>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CommitProgressEvent">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommitProgressEvent" style="document"/>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="CommitErrorEvent">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/CommitErrorEvent" style="document"/>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DownloadProgressEvent">
      <soap12:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/FirmwareUpdate/DownloadProgressEvent" style="document"/>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>

</wsdl:definitions>
```

C.3 Firmware Update Service XSD

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
  elementFormDefault="qualified"
  targetNamespace="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://standards.iso.org/iso/24791/-3/2013/01/rdmp">

  <xs:element name="DownloadAndCommit">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="FirmwareLocation" type="xs:anyURI"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>
  <xs:element name="DownloadAndCommitResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="DownloadAndScheduleCommit">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="FirmwareLocation" type="xs:anyURI"/>
        <xs:element name="AtTime" type="xs:dateTime"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>
  <xs:element name="DownloadAndScheduleCommitResponse">
```

```
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="GetStatus">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>
  <xs:element name="GetStatusResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetStatusResult" type="tns:FirmwareUpdateStatus"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="CheckFirmwareApplicability">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="FirmwareVersion" type="xs:string"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>
  <xs:element name="CheckFirmwareApplicabilityResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="CheckFirmwareApplicabilityResult"
type="tns:FirmwareComparisonResult"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="Cancel">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>
  <xs:element name="CancelResponse">
    <xs:complexType>
      <xs:sequence>
```

```
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="CommittedEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="DownloadedEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="CommitProgressEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="CommitProgressEventInfo"
type="tns:CommitProgressEventInfo"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="CommitErrorEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="CommitErrorEventInfo" type="tns:ErrorEventInfo"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="DownloadErrorEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DownloadErrorEventInfo" type="tns:ErrorEventInfo"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="DownloadProgressEvent">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DownloadProgressEventInfo"
type="tns:DownloadProgressEventInfo"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

```
  <xs:complexType name="FirmwareUpdateStatus">
    <xs:sequence>
      <xs:element name="CurrentFirmwareVersion" type="xs:string"/>
      <xs:element name="CurrentState" type="tns:FirmwareUpdateState"/>
      <xs:element minOccurs="0" name="PastErrors" nillable="true"
type="tns:ArrayOfErrorEventInfo"/>
      <xs:element minOccurs="0" name="UploadedVersion" nillable="true"
type="xs:string"/>
      <xs:element minOccurs="0" name="CommitProgress" nillable="true"
type="tns:CommitProgressEventInfo"/>
      <xs:element minOccurs="0" name="DownloadProgress" nillable="true"
type="tns:DownloadProgressEventInfo"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:simpleType name="FirmwareUpdateState">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Ready"/>
      <xs:enumeration value="Downloading"/>
      <xs:enumeration value="Downloaded"/>
      <xs:enumeration value="Committing"/>
      <xs:enumeration value="FatalError"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="FirmwareComparisonResult">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Newer"/>
      <xs:enumeration value="Older"/>
      <xs:enumeration value="SameVersion"/>
      <xs:enumeration value="NotApplicable"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="ArrayOfErrorEventInfo">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="ErrorEventInfo"
nillable="true" type="tns:ErrorEventInfo"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ErrorEventInfo">
    <xs:sequence>
      <xs:element name="Description" type="xs:string"/>
      <xs:element name="Time" type="xs:dateTime"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:complexType name="CommitProgressEventInfo">
    <xs:sequence>
      <xs:element minOccurs ="0" name="Description" nillable="true"
type="xs:string"/>
      <xs:element name="PercentCompleted" type="xs:byte"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
```

```
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:complexType name="DownloadProgressEventInfo">
    <xs:sequence>
      <xs:element name="PercentCompleted" type="xs:byte"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

</xs:schema>
```

C.4 Management Service WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
   xmlns:tns="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
   xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

  <wsdl:types>
    <xsd:schema targetNamespace="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp">
      <xsd:include schemaLocation="Management.xsd"/>
    </xsd:schema>
  </wsdl:types>

  <wsdl:message name="Management_GetSources_InputMessage">
    <wsdl:part name="parameters" element="tns:GetSources"/>
  </wsdl:message>
  <wsdl:message name="Management_GetSources_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetSourcesResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_GetAllPropertyMetadata_InputMessage">
    <wsdl:part name="parameters" element="tns:GetAllPropertyMetadata"/>
  </wsdl:message>
  <wsdl:message name="Management_GetAllPropertyMetadata_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetAllPropertyMetadataResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyMetadata_InputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyMetadata"/>
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyMetadata_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyMetadataResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyValue_InputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyValue"/>
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyValue_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyValueResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyValuesByGroup_InputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyValuesByGroup"/>
```

```
  </wsdl:message>
  <wsdl:message name="Management_GetPropertyValuesByGroup_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetPropertyValuesByGroupResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_GetAllPropertyValues_InputMessage">
    <wsdl:part name="parameters" element="tns:GetAllPropertyValues"/>
  </wsdl:message>
  <wsdl:message name="Management_GetAllPropertyValues_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetAllPropertyValuesResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_SetPropertyProfile_InputMessage">
    <wsdl:part name="parameters" element="tns:SetPropertyProfile"/>
  </wsdl:message>
  <wsdl:message name="Management_SetPropertyProfile_OutputMessage">
    <wsdl:part name="parameters" element="tns:SetPropertyProfileResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_Reboot_InputMessage">
    <wsdl:part name="parameters" element="tns:Reboot"/>
  </wsdl:message>
  <wsdl:message name="Management_Reboot_OutputMessage">
    <wsdl:part name="parameters" element="tns:RebootResponse"/>
  </wsdl:message>
  <wsdl:message name="Management_ResetToFactorySettings_InputMessage">
    <wsdl:part name="parameters" element="tns:ResetToFactorySettings"/>
  </wsdl:message>
  <wsdl:message name="Management_ResetToFactorySettings_OutputMessage">
    <wsdl:part name="parameters" element="tns:ResetToFactorySettingsResponse"/>
  </wsdl:message>
  <wsdl:message
name="Management_ResetToFactorySettingsExceptNetwork_InputMessage">
    <wsdl:part name="parameters"
element="tns:ResetToFactorySettingsExceptNetwork"/>
  </wsdl:message>
  <wsdl:message
name="Management_ResetToFactorySettingsExceptNetwork_OutputMessage">
    <wsdl:part name="parameters"
element="tns:ResetToFactorySettingsExceptNetworkResponse"/>
  </wsdl:message>

  <!--Service Defination for Management-->
  <wsdl:portType name="Management">
    <wsdl:operation name="GetSources">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetSources"
message="tns:Management_GetSources_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetSourcesResponse"
message="tns:Management_GetSources_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetAllPropertyMetadata">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyMetadata"
message="tns:Management_GetAllPropertyMetadata_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyMetadataResponse"
message="tns:Management_GetAllPropertyMetadata_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetPropertyMetadata">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyMetadata"
message="tns:Management_GetPropertyMetadata_InputMessage"/>
```

```
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyMetadataResponse"
message="tns:Management_GetPropertyMetadata_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetPropertyValue">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValue"
message="tns:Management_GetPropertyValue_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValueResponse"
message="tns:Management_GetPropertyValue_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetPropertyValuesByGroup">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValuesByGroup"
message="tns:Management_GetPropertyValuesByGroup_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValuesByGroupResponse"
message="tns:Management_GetPropertyValuesByGroup_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="GetAllPropertyValues">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyValues"
message="tns:Management_GetAllPropertyValues_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyValuesResponse"
message="tns:Management_GetAllPropertyValues_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="SetPropertyProfile">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/SetPropertyProfile"
message="tns:Management_SetPropertyProfile_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/SetPropertyProfileResponse"
message="tns:Management_SetPropertyProfile_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="Reboot">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/Reboot" message="tns:Management_Reboot_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/RebootResponse"
message="tns:Management_Reboot_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="ResetToFactorySettings">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettings"
message="tns:Management_ResetToFactorySettings_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettingsResponse"
message="tns:Management_ResetToFactorySettings_OutputMessage"/>
    </wsdl:operation>
    <wsdl:operation name="ResetToFactorySettingsExceptNetwork">
      <wsdl:input wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettingsExceptNetwork"
message="tns:Management_ResetToFactorySettingsExceptNetwork_InputMessage"/>
      <wsdl:output wsaw:Action="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettingsExceptNetworkResponse"
message="tns:Management_ResetToFactorySettingsExceptNetwork_OutputMessage"/>
    </wsdl:operation>
  </wsdl:portType>
```

```
  <!--Binding for Management-->
  <wsdl:binding name="DefaultBinding_Management" type="tns:Management">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>

    <wsdl:operation name="GetSources">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetSources" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetAllPropertyMetadata">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyMetadata" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetPropertyMetadata">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyMetadata" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetPropertyValue">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValue" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetPropertyValuesByGroup">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetPropertyValuesByGroup" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetAllPropertyValues">
```

```
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/GetAllPropertyValues" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="SetPropertyProfile">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/SetPropertyProfile" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="Reboot">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/Reboot" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="ResetToFactorySettings">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettings" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="ResetToFactorySettingsExceptNetwork">
      <soap:operation soapAction="http://standards.iso.org/iso/24791/-
3/2013/01/rdmp/Management/ResetToFactorySettingsExceptNetwork" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

C.5 Management Service XSD

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
```

```
  elementFormDefault="qualified"
  targetNamespace="http://standards.iso.org/iso/24791/-3/2013/01/rdmp"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://standards.iso.org/iso/24791/-3/2013/01/rdmp">
<xs:include schemaLocation="LLRP.xsd"/>
  <xs:element name="GetSources">
  <xs:complexType>
    <xs:sequence>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  </xs:element>
  <xs:element name="GetSourcesResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetSourcesResult" type="tns:Sources"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="Sources">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="Source"
nillable="true" type="tns:Source" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="Source">
    <xs:sequence>
      <xs:element name="SourceName" type="xs:token"/>
      <xs:element name="SourceType" type="xs:QName"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <xs:element name="GetAllPropertyMetadata">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" nillable="true" name="SourceName"
type="xs:token"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetAllPropertyMetadataResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetAllPropertyMetadataResult"
type="tns:PropertyMetadataDictionary"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="PropertyMetadataDictionary">
    <xs:sequence>
```

```
      <xs:element minOccurs="0" maxOccurs="unbounded" name="PropertyMetadataPair"
nillable="true" type="tns:PropertyMetadataPair"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="PropertyMetadataPair">
    <xs:sequence>
      <xs:element name="PropertyIdentifier" type="xs:QName"/>
      <xs:element name="PropertyMetadata" type="tns:PropertyMetadata"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="PropertyMetadata">
    <xs:sequence>
      <xs:element name="Type" type="xs:QName" />
      <xs:element name="Description" type="xs:string" />
      <xs:element name="Writeable" type="xs:boolean" />
      <xs:element name="IsPersistent" type="xs:boolean" />
      <xs:element name="RequiresRestart" type="xs:boolean" />
      <xs:element minOccurs="0" maxOccurs="1" name="DefaultValue"
type="tns:PropertyValue" />
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:complexType name="PropertyValue">

    <xs:choice>
      <xs:element name="String" type="xs:string" />
      <xs:element name="Integer" type="xs:int" />
      <xs:element name="TagDataSelector" type="tns:TagDataSelector" />
      <xs:element name="QName" type="xs:QName" />
      <xs:element name="Boolean" type="xs:boolean"/>
      <xs:element name="Float" type="xs:float"/>
      <xs:element name="Double" type="xs:double"/>
      <xs:element name="Uri" type="xs:anyURI"/>
      <xs:element name="DateTime" type="xs:dateTime"/>
      <xs:element name="LLRPStatus" type="tns:LLRPStatus"/>
      <xs:element name="GeneralDeviceCapabilities"
type="tns:GeneralDeviceCapabilities"/>
      <xs:element name="RegulatoryCapabilities"
type="tns:RegulatoryCapabilities"/>
      <xs:element name="AntennaProperties" type="tns:AntennaProperties"/>
      <xs:element name="AntennaConfiguration" type="tns:AntennaConfiguration"/>
      <xs:element name="KeepaliveSpec" type="tns:KeepaliveSpec"/>
      <xs:element name="EventsAndReports" type="tns:EventsAndReports"/>
      <xs:element name="Identification" type="tns:Identification"/>
      <xs:any namespace="##other" processContents="lax"/>
    </xs:choice>

  </xs:complexType>

  <xs:complexType name="TagDataSelector">
    <xs:sequence>
      <xs:element name="ID" type="xs:boolean"/>
      <xs:element name="Data" type="xs:boolean"/>
      <xs:element name="SourceName" type="xs:boolean"/>
      <xs:element name="NumberingSystemIdentifier" type="xs:boolean"/>
      <xs:element name="Time" type="xs:boolean"/>
```

```
      <xs:element name="TagType" type="xs:boolean"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="GetPropertyMetadata">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" name="SourceName" nillable="true"
type="xs:token"/>
        <xs:element name="PropertyIdentifier" type="xs:QName"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetPropertyMetadataResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetPropertyMetadataResult"
type="tns:PropertyMetadata"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="GetPropertyValue">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" name="SourceName" nillable="true"
type="xs:token"/>
        <xs:element name="PropertyIdentifier" type="xs:QName"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetPropertyValueResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetPropertyValueResult" type="tns:PropertyValue"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="GetPropertyValuesByGroup">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" nillable="true" name="SourceName"
type="xs:token"/>
        <xs:element name="GroupURI" type="xs:anyURI"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetPropertyValuesByGroupResponse">
    <xs:complexType>
      <xs:sequence>
```

```
        <xs:element name="GetPropertyValuesByGroupResult"
type="tns:PropertyProfile"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="PropertyProfile">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="PropertyValuePair"
nillable="true" type="tns:PropertyValuePair" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="PropertyValuePair">
    <xs:sequence>
      <xs:element name="PropertyIdentifier" type="xs:QName"/>
      <xs:element name="PropertyValue" type="tns:PropertyValue" nillable="true"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="GetAllPropertyValues">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" nillable="true" name="SourceName"
type="xs:token"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetAllPropertyValuesResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GetAllPropertyValuesResult"
type="tns:PropertyProfile"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="SetPropertyProfile">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" nillable="true" name="SourceName"
type="xs:token"/>
        <xs:element name="PropertyProfile" type="tns:PropertyProfile"/>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetPropertyProfileResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other"
processContents="lax"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
```