

# **OpenCable Specifications**

## **OpenCable Receiver Metrics Gathering Specification**

### **OC-SP-Metrics-I02-070416**

**ISSUED**

#### **Notice**

This document is the result of a cooperative effort undertaken at the direction of Cable Television Laboratories, Inc. for the benefit of the cable industry and its customers. This document may contain references to other documents not owned or controlled by CableLabs. Use and understanding of this document may require access to such other documents. Designing, manufacturing, distributing, using, selling, or servicing products, or providing services, based on this document may require intellectual property licenses from third parties for technology referenced in this document.

Neither CableLabs nor any member company is responsible to any party for any liability of any nature whatsoever resulting from or arising out of use or reliance upon this document, or any document referenced herein. This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, non-infringement, or fitness for a particular purpose of this document, or any document referenced herein.

© Copyright 2006-2007 Cable Television Laboratories, Inc.  
All rights reserved.

## Document Status Sheet

<b>Document Control Number:</b>	OC-SP-Metrics-I02-070416			
<b>Document Title:</b>	OpenCable Receiver Metrics Gathering Specification			
<b>Revision History:</b>	I01 – Released 12/29/06			
	I02 – Released 4/16/07			
<b>Date:</b>	April 16, 2007			
<b>Status:</b>	<i>Work in Progress</i>	<i>Draft</i>	<b>Issued</b>	<i>Closed</i>
<b>Distribution Restrictions:</b>	<i>Author Only</i>	<i>GL/Member</i>	<i>GL/Member/Vendor</i>	<b>Public</b>

### Key to Document Status Codes:

- Work in Progress** An incomplete document, designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
- Draft** A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
- Issued** A stable document, which has undergone rigorous member and vendor review and is suitable for product design and development, cross-vendor interoperability, and for certification testing.
- Closed** A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

### Trademarks:

CableLabs<sup>®</sup>, DOCSIS<sup>®</sup>, EuroDOCSIS<sup>™</sup>, eDOCSIS<sup>™</sup>, M-CMTS<sup>™</sup>, PacketCable<sup>™</sup>, EuroPacketCable<sup>™</sup>, PCMM<sup>™</sup>, CableHome<sup>®</sup>, CableOffice<sup>™</sup>, OpenCable<sup>™</sup>, OCAP<sup>™</sup>, CableCARD<sup>™</sup>, M-Card<sup>™</sup>, and DCAS<sup>™</sup> are trademarks of Cable Television Laboratories, Inc.

## Contents

<b>1</b>	<b>SCOPE</b> .....	<b>1</b>
1.1	Introduction and Purpose.....	1
1.2	Requirements.....	1
<b>2</b>	<b>REFERENCES</b> .....	<b>2</b>
2.1	Normative References.....	2
2.2	Informative References.....	2
2.3	Reference Acquisition.....	2
<b>3</b>	<b>TERMS AND DEFINITIONS</b> .....	<b>3</b>
<b>4</b>	<b>ABBREVIATIONS AND ACRONYMS</b> .....	<b>3</b>
<b>5</b>	<b>OVERVIEW</b> .....	<b>4</b>
5.1	General Context.....	4
<b>6</b>	<b>METRICS REQUIREMENTS</b> .....	<b>5</b>
6.1	Transmission Protocol.....	5
6.2	Data Model.....	6
<b>ANNEX A</b>	<b>METRICS DATA MODEL SCHEMA (NORMATIVE)</b> .....	<b>8</b>
<b>ANNEX B</b>	<b>SCHEMA DATA ELEMENT REQUIREMENTS (NORMATIVE)</b> .....	<b>17</b>
<b>APPENDIX I</b>	<b>ACKNOWLEDGEMENTS</b> .....	<b>21</b>
<b>APPENDIX II</b>	<b>REVISION HISTORY</b> .....	<b>22</b>

## Figures

Figure 1 - Logical Model of Metrics System.....	4
---	---

This page left blank intentionally.

# 1 SCOPE

## 1.1 Introduction and Purpose

The purpose of this document is to specify metrics collection and transmission mechanisms and interfaces on OpenCable receiver platforms. The data formats and transmission protocols may also be adopted by proprietary systems, such as ETV-enabled cable receivers.

For the purposes of this specification, the term "metrics" refers to specific data, defined herein, that is generated by a receiver and transmitted to a cable headend.

Metrics, as defined in this specification, are intended to complement other measurement systems that may be employed by cable operators.

## 1.2 Requirements

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"SHALL"	This word means that the item is an absolute requirement of this specification.
"SHALL NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

## 2 REFERENCES

### 2.1 Normative References

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

- [ATIS] Usage Data Management For Packet-Based Services – Service-Neutral Protocol Specification For Billing Applications, ATIS-0300075.1.2006, August 2006.  
<https://www.atis.org/docstore/product.aspx?id=22606>
- [IPDR/SP] IPDR/SP Protocol Specification Version 2.2, August 25, 2006.  
<http://www.ipdr.org/public/DocumentMap/SP2.2.pdf>
- [IPDR/XDR] IPDR/XDR File Encoding Format, Version 3.5.1, November 2004.  
<http://www.ipdr.org/public/DocumentMap/XDR3.5.1.pdf>

### 2.2 Informative References

This document uses the following informative references.

- [OCAP 1.1] OpenCable Applications Platform, Profile 1.1, OC-SP-OCAP1.1-I01-061229, December 29, 2006, Cable Television Laboratories, Inc.
- [ETV] Enhanced TV Binary Interchange Format, OC-SP-ETV-BIF1.0-I03-060714, July 14, 2006, Cable Television Laboratories, Inc.
- [IPDR/SSDG] IPDR/SSDG Service Specification Design Guide, Version 3.5.1, November 2004.  
<http://www.ipdr.org/public/DocumentMap/SSDG3.5.1.pdf>

### 2.3 Reference Acquisition

[OCAP 1.1] and [ETV] are available from CableLabs at [www.opencable.com](http://www.opencable.com)

### 3 TERMS AND DEFINITIONS

This specification uses the following terms:

<b>Data Message</b>	A message transmitted between an IPDR Exporter and Collector across the Streaming Protocol, containing a common Streaming Protocol header and an optional payload consisting of control and Data Records.
<b>Data Record</b>	The binary encoding of an IPDR record.
<b>IPDR</b>	Internet Protocol Detail Record, the fundamental unit of data transferred between an Exporter and a Collector. It is defined by a Template and contains Fields.
<b>IPDR/SP</b>	Internet Protocol Detail Record/Streaming Protocol, the protocol used to transfer Data Messages and Data Records between Exporter and Collectors.
<b>IPDR-Type</b>	A constraint on the value and format of an individual Field within a Data Record; e.g., dateTime.
<b>IPDR/SP Collector functionality</b>	An implementation on the data receiving side of the IPDR/SP. It enables the reception and collection of Data Records from IPDR/SP Exporters. It is typically part of an OSS/BSS, or a mediation system.
<b>IPDR/SP Exporter functionality</b>	An implementation on the data-producing side of the IPDR/SP. It enables formatting and sending of Data Records to an interested consumer system, e.g., at a cable headend using the IPDR/SP.
<b>Authorized Collector</b>	An Event Tracking API-compliant server that implements the receiving side of the IPDR Streaming Protocol, and which has been authorized to participate in the overall Collection System.

### 4 ABBREVIATIONS AND ACRONYMS

This specification uses the following abbreviations:

<b>IPDR</b>	Internet Protocol Detail Record
<b>IPDR/SP</b>	Internet Protocol Detail Record Streaming Protocol

## 5 OVERVIEW

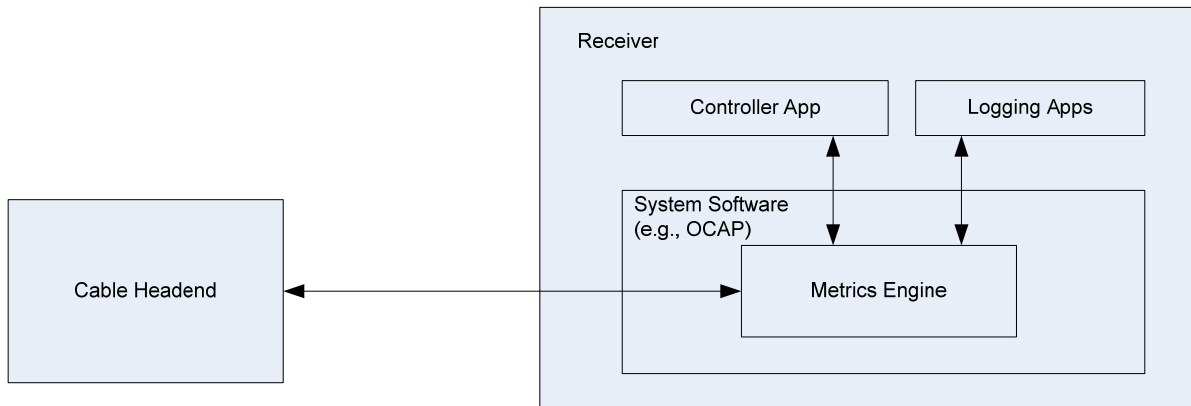
### 5.1 General Context

Metrics are a set of well-defined data that are reported to a cable network by a receiver, for the purpose of measuring certain aspects of cable service delivery. This document specifies a data model and transmission protocol that will be supported by OpenCable and non-OpenCable Host devices. Platforms that support this specification are expected to define the semantics for the data points that are syntactically defined herein.

The means for metrics collection on the receiver is composed of a log, which contains log records. A log and log records are created and stored in a format determined by an implementer. Log records are created by the receiver in response to well-defined events, such as operations performed by a receiver on behalf of applications, or generated directly by applications. A log and its log records are transmitted from the receiver in a well-known format, under control of logic implemented by the network operator.

There are three logical metrics components: a so-called "metrics engine" implemented within a receiver; one or more privileged applications that configure and manage the metrics engine, collectively called the "metrics controller"; and any other "metrics logging" applications that interface with the metrics engine to generate application-specific log records.

This specification defines requirements for a metrics engine and the interface between a metrics engine and a cable headend. Metrics data is considered private to the cable network. Functions such as storage, aggregation, and reporting are the responsibility of a network operator. This interface provides a well-known data format and protocol to ensure that metrics are comparable across cable networks.



**Figure 1 - Logical Model of Metrics System**

Interfaces between metrics controller applications and the metric engine are defined in [OCAP 1.1]. Interfaces between logging applications and the metrics engine are defined in [OCAP 1.1] and [ETV].

The metrics engine incorporates IPDR/SP Exporter functionality. A cable headend that supports metrics implements IPDR/SP Collector functionality. See [IPDR/SP] for details.

## 6 METRICS REQUIREMENTS

This section defines requirements for metrics and the interface between a metrics engine and a cable headend.

### 6.1 Transmission Protocol

IPDR/SP is the protocol for transmitting metrics data to a cable headend. It is an American National Standard ATIS-0300075.1.2006 [ATIS]. The IPDR/SP utilizes a light-weight, session-oriented, real-time streaming protocol to quickly and reliably deliver metrics data, a.k.a. IPDRs, from an "Exporter" to a "Collector" (typically a type of mediation system). Receivers that support OpenCable metrics SHALL implement IPDR/SP Exporter functionality. For brevity, the term "metrics engine" is to be understood to include IPDR/SP Exporter functionality.

The IPDR/SP utilizes a slightly augmented version of the XDR protocol, described in IETF RFC 1832, to encode IPDRs. IPDR/SP uses many of the primitive types defined by the IETF XDR protocol to encode as much information as possible in a compact binary representation.

The IPDR/SP uses TCP as its transport layer protocol. The transport layer acknowledgements are used to ensure the reliable delivery of data packets and detection of lost Exporters. In addition, IPDR/SP allows another level of application reliability as described below.

### IPDR/SP Templates

The IPDR/SP uses a highly-efficient encoding scheme, where only the values for each field of a particular type of IPDR are encapsulated and sent in a Data Message from the Exporter to the Collector. The field names, types, and lengths are not included in the IPDRs; instead, the field names, types, and lengths are specified indirectly by referring to a session-specific Template that describes contents of that type of IPDR.

Within the IPDR/SP, IPDRs are always exchanged within the context of a particular session that exists between a particular Exporter and a particular Collector. During the session establishment phase, the Exporter provides the Collector with a list of Templates, where each Template describes the layout of a particular type of Data Record (a Data Record is a binary encoding of an IPDR record) that can be sent to the Collector. Data Records are always packaged into a Data Message before being sent to the Collector.

Each Template definition contains a TemplateID, an IPDR-Type name, an explicit reference to the XML Schema which describes the IPDR-Type referred to by the Template, and an ordered list of triplets, where each triplet consists of a field name, field ID, and type. These field names and types must correspond to those defined in the corresponding IPDR Schema for the particular IPDR-Type referred to by the Template.

A particular Template not only specifies an IPDR-Type and corresponding XML Schema, but also specifies the particular subset of fields, and the order in which those fields will be laid out within Data Records referencing that Template. A Template for a particular IPDR-Type does not have to contain all possible fields defined in the XML Schema for that IPDR-Type; it might exclude certain optional or conditional fields. As such, there could be multiple Templates referring to the same IPDR-Type within a particular session.

Typically the Exporter sends a list of supported Templates to the Collector during the Session Establishment phase; however, the IPDR/SP also provides a mechanism whereby a Collector can negotiate with an Exporter regarding the Templates to be used during that session.

Once a session has been established, Data Messages are sent from the Exporter to the Collector. Each Data Message contains a sequence number, a TemplateID, which points to the particular Template that describes the contents of the Data Record contained within this particular Data Message, followed by a Data Record (a binary-encoded

IPDR). An active session can simultaneously support many different IPDR-Types and many different subsets of fields for a particular IPDR-Type.

The use of Templates makes it possible for Exporters to deliver only the actual data values for each usage event without having to include any data descriptors in each Data Record. This significantly reduces the volume of information sent over communication links. The use of Templates also has the added advantage of enabling Data Records to contain only a pertinent subset of all fields for a particular IPDR-Type, thereby reducing the number of "empty" fields. This also reduces the volume of information sent over communication links.

## System level reliability

During Session Establishment, the Exporter and the Collector agree on how often the Collector will send a Data Acknowledge Message back to the Exporter. Each Data Acknowledge Message refers to the sequence number of the most recently received and stored Data Message. By sending a Data Acknowledge Message, the Collector tells the Exporter that all Data Messages with sequence numbers less than or equal to the sequence number contained within the Data Acknowledge Message have been received and transferred to some type of persistent storage.

Once the Collector has acknowledged receipt of a particular Data Message, the Exporter no longer needs to maintain the Data Record contained within that Data Message. If the Collector goes down, or if the connection between the Exporter and Collector is disrupted, the Exporter can compare its most recently generated Data Message sequence number with the last acknowledged sequence number to know which Data Messages and Data Records it needs to preserve and either resend to the Collector or send to alternate Collectors. The Exporter can continue to stream new Data Messages to a Collector, regardless of whether previous Data Messages have been acknowledged, as long as the Collector sends intermittent Data Acknowledge Messages at the frequency agreed to at the start of the session.

A metrics engine SHALL adhere to the following requirements:

- The metrics engine SHALL NOT attempt to connect to an unauthorized Collector.
- The metrics engine SHALL NOT generate data elements unless connected to and configured to by an authorized Collector.

This specification does not place requirements on Collectors. The following guidelines may be used by cable operators in order to support metrics collection.

- The Collectors SHOULD implement the IPDR/SP Collector functionality.
- One or more redundant Collectors SHOULD be configured such that there MAY be more than one Collector connected to one Exporter. This configuration provides high availability and improved robustness of the collection system.

## 6.2 Data Model

Within the IPDR/SP, every IPDR refers to its particular IPDR-Type (or type of event message), which then defines the contents that an IPDR of that IPDR-Type is expected to contain. IPDR/SP requires that all IPDR-Types be specified using the XML schema language. Defining event messages within IPDR/SP is a relatively easy and straightforward process.

[IPDR/SSDG] defines a human readable format for defining a generic data model, as an XML Schema. Annex A of this document defines a data model for OpenCable metrics using this format. Receivers that support OpenCable metrics SHALL implement the data model defined in Annex A.

The XML schema defining a particular IPDR-Type SHALL contain the type-name for that IPDR-Type and a list of all possible fields that MAY appear within an IPDR of that type. Each field has a name, type, and an attribute

specifying whether that field is required, optional, or conditional. Additionally, each field MAY be annotated with supplemental textual information, explaining the nature of the data contained in that field.

The IPDR/SP gives implementers nearly complete freedom to define the fields that are contained within a particular IPDR -Type.

XML schema provides an easy way for cable operators and vendors to add their own extensions to the base event message types defined in this specification. Extensions MAY be added simply by creating a new XML schema, defining a new IPDR-Type name, importing fields from pre-existing "standard" IPDR-Types, and then adding definitions only for any new implementation-specific fields.

Since XML schema supports namespaces, there is a built-in mechanism for reducing the chances of field name collision, especially when using implementation specific extensions, by placing implementation-specific fields into their own unique namespace. The "ipdr" namespace, which contains all of the IPDR defined fields, is managed by IPDR.org. Other entities are free to create their own namespaces, and place their fields within their own unique namespaces.

IPDR/SP defines a mechanism for transforming the XML human readable data model into a binary format optimized for on-the-wire transmission, via XDR. As part of their requirement to implement an IPDR/SP Exporter, receivers that support OpenCable metrics SHALL be able to transmit a binary representation of the data model defined in Annex A.

Since XML schema is used to describe the contents of IPDRs, it is a simple exercise to translate an XDR encoded IPDR to an XML encoding, and vice versa. While it would be undesirable to send XML-encoded IPDRs over a high-traffic connection, once the IPDRs have arrived at a Collector there are circumstances where being able to translate those IPDRs from XDR to XML can be quite helpful for debugging, human readability, or translation to other encoding schemes.

## Annex A Metrics Data Model Schema (Normative)

```

<?xml version="1.0" encoding="UTF-8"?>

<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:OCAP="http://www.ipdr.org/namespaces/CableLabs(R)/OCAP(TM)"
xmlns:ipdr="http://www.ipdr.org/namespaces/ipdr"
targetNamespace="http://www.ipdr.org/namespaces/CableLabs(R)/OCAP(TM)" version="3.5-
A.0" elementFormDefault="qualified" attributeFormDefault="unqualified">
  <import namespace="http://www.ipdr.org/namespaces/ipdr"
schemaLocation="http://www.ipdr.org/public/IPDRDoc3.5.1.xsd"/>
  <include schemaLocation="http://www.ipdr.org/public/IPDRTypes.xsd"/>

  <element name="protocolVersionMajor" type="byte">
  </element>

  <element name="protocolVersionMinor" type="byte">
  </element>

  <element name="hostID" type="string"/>

  <element name="hostType">
    <simpleType>
      <restriction base="int">
        <enumeration value="1">
          <annotation>
            <appinfo>
              <ipdr:enumMeaning>
                OCAP
              </ipdr:enumMeaning>
            </appinfo>
          </annotation>
        </enumeration>
        <enumeration value="2">
          <annotation>
            <appinfo>
              <ipdr:enumMeaning>
                ETV
              </ipdr:enumMeaning>
            </appinfo>
          </annotation>
        </enumeration>
        <enumeration value="3">
          <annotation>
            <appinfo>
              <ipdr:enumMeaning>
                other
              </ipdr:enumMeaning>
            </appinfo>
          </annotation>
        </enumeration>
      </restriction>
    </simpleType>
  </element>

  <element name="createTime" type="ipdr:dateTimeMsec">
  </element>

  <element name="transmitTime" type="ipdr:dateTimeMsec">
  </element>

  <element name="privateData" type="hexBinary">
  </element>

  <element name="serviceSelection">
    <complexType>

```

```

    <sequence>
      <element ref="createTime"/>
      <element name="sourceID" type="int">
    </element>
  </sequence>
</complexType>
</element>

<element name="tuning">
  <complexType>
    <sequence>
      <element ref="createTime"/>
      <element name="tunerID" type="int">
    </element>
      <element name="frequency" type="int">
    </element>
      <element name="programNumber" type="int">
    </element>
    </sequence>
  </complexType>
</element>

<element name="SDV">
  <complexType>
    <sequence>
      <element ref="createTime"/>
      <element name="sourceID" type="int">
    </element>
    </sequence>
  </complexType>
</element>

<element name="servicePresentation">
  <complexType>
    <sequence>
      <element ref="createTime"/>
      <element name="sourceID" type="int">
    </element>
      <element name="aspectRatio">
        <simpleType>
          <restriction base="int">
            <enumeration value="1">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    4:3
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
            <enumeration value="2">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    16:9
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
          </restriction>
        </simpleType>
      </element>
      <element name="resolution" type="int">
    </element>
      <element name="audio">
        <simpleType>
          <restriction base="int">

```

```

        <enumeration value="0">
            <annotation>
                <appinfo>
                    <ipdr:enumMeaning>
                        No audio
                    </ipdr:enumMeaning>
                </appinfo>
            </annotation>
        </enumeration>
        <enumeration value="1">
            <annotation>
                <appinfo>
                    <ipdr:enumMeaning>
                        Audio
                    </ipdr:enumMeaning>
                </appinfo>
            </annotation>
        </enumeration>
    </restriction>
</simpleType>
</element>
</sequence>
</complexType>
</element>

<element name="segmentPresentation">
    <complexType>
        <sequence>
            <element ref="createTime"/>
            <element name="segmentID" type="int">
            </element>
        </sequence>
    </complexType>
</element>

<element name="DPI">
    <complexType>
        <sequence>
            <element ref="createTime"/>
            <element name="sourceID" type="int">
            </element>
            <element name="replacementID" type="int">
            </element>
            <element name="replacementSource">
                <simpleType>
                    <restriction base="int">
                        <enumeration value="1">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        In transport
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="2">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        Out of transport
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="3">
                            <annotation>
                                <appinfo>

```

```

        <ipdr:enumMeaning>
            From Memory
        </ipdr:enumMeaning>
    </appinfo>
</annotation>
</enumeration>
</restriction>
</simpleType>
</element>
<element name="switchLevel" type="int">
    <simpleType>
        <restriction base="int">
            <enumeration value="0">
                <annotation>
                    <appinfo>
                        <ipdr:enumMeaning>
                            L0
                        </ipdr:enumMeaning>
                    </appinfo>
                </annotation>
            </enumeration>
            <enumeration value="1">
                <annotation>
                    <appinfo>
                        <ipdr:enumMeaning>
                            L1
                        </ipdr:enumMeaning>
                    </appinfo>
                </annotation>
            </enumeration>
        </restriction>
    </simpleType>
</element>
</sequence>
</complexType>
</element>

<element name="inputEvent">
    <complexType>
        <sequence>
            <element ref="createTime"/>
            <element name="keyCode" type="int">
            </element>
        </sequence>
    </complexType>
</element>

<element name="applicationLoadRequested">
    <complexType>
        <sequence>
            <element ref="createTime"/>
            <element name="appID" type="int">
            </element>
            <element name="appType">
                <simpleType>
                    <restriction base="int">
                        <enumeration value="1">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        OCAP
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="2">
                            <annotation>

```

```

        <appinfo>
          <ipdr:enumMeaning>
            ETV
          </ipdr:enumMeaning>
        </appinfo>
      </annotation>
    </enumeration>
  </restriction>
</simpleType>
</element>
</sequence>
</complexType>
</element>
<element name="applicationLoaded">
  <complexType>
    <sequence>
      <element ref="createTime"/>
      <element name="appID" type="int">
      </element>
      <element name="appType">
        <simpleType>
          <restriction base="int">
            <enumeration value="1">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    OCAP
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
            <enumeration value="2">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    ETV
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
          </restriction>
        </simpleType>
      </element>
      <element name="numResourceFiles" type="int">
      </element>
      <element name="transportProtocol">
        <simpleType>
          <restriction base="int">
            <enumeration value="1">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    IBOC
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
            <enumeration value="2">
              <annotation>
                <appinfo>
                  <ipdr:enumMeaning>
                    OBOC
                  </ipdr:enumMeaning>
                </appinfo>
              </annotation>
            </enumeration>
          </restriction>
        </simpleType>
      </element>
    </sequence>
  </complexType>
</element>

```

```

        <enumeration value="3">
            <annotation>
                <appinfo>
                    <ipdr:enumMeaning>
                        HTTP
                    </ipdr:enumMeaning>
                </appinfo>
            </annotation>
        </enumeration>
    </restriction>
</simpleType>
</element>
<element name="controlCode">
    <simpleType>
        <restriction base="int">
            <enumeration value="1">
                <annotation>
                    <appinfo>
                        <ipdr:enumMeaning>
                            LOADED
                        </ipdr:enumMeaning>
                    </appinfo>
                </annotation>
            </enumeration>
            <enumeration value="2">
                <annotation>
                    <appinfo>
                        <ipdr:enumMeaning>
                            PAUSED
                        </ipdr:enumMeaning>
                    </appinfo>
                </annotation>
            </enumeration>
            <enumeration value="3">
                <annotation>
                    <appinfo>
                        <ipdr:enumMeaning>
                            RUNNING
                        </ipdr:enumMeaning>
                    </appinfo>
                </annotation>
            </enumeration>
        </restriction>
    </simpleType>
</element>
</sequence>
</complexType>
</element>

<element name="applicationLifecycle">
    <complexType>
        <sequence>
            <element ref="createTime"/>
            <element name="appID" type="int"/>
            <element name="appType">
                <simpleType>
                    <restriction base="int">
                        <enumeration value="1">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        OCAP
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="2">

```

```

        <annotation>
          <appinfo>
            <ipdr:enumMeaning>
              ETV
            </ipdr:enumMeaning>
          </appinfo>
        </annotation>
      </enumeration>
    </restriction>
  </simpleType>
</element>
<element name="lifecycleState">
  <simpleType>
    <restriction base="int">
      <enumeration value="1">
        <annotation>
          <appinfo>
            <ipdr:enumMeaning>
              LAUNCHED
            </ipdr:enumMeaning>
          </appinfo>
        </annotation>
      </enumeration>
      <enumeration value="2">
        <annotation>
          <appinfo>
            <ipdr:enumMeaning>
              PAUSED
            </ipdr:enumMeaning>
          </appinfo>
        </annotation>
      </enumeration>
      <enumeration value="3">
        <annotation>
          <appinfo>
            <ipdr:enumMeaning>
              RESUMED
            </ipdr:enumMeaning>
          </appinfo>
        </annotation>
      </enumeration>
      <enumeration value="4">
        <annotation>
          <appinfo>
            <ipdr:enumMeaning>
              TERMINATED
            </ipdr:enumMeaning>
          </appinfo>
        </annotation>
      </enumeration>
    </restriction>
  </simpleType>
</element>
</sequence>
</complexType>
</element>

<element name="environmentSelection">
  <complexType>
    <sequence>
      <element ref="createTime"/>
      <element name="environment">
        <simpleType>
          <restriction base="int">
            <enumeration value="1">
              <annotation>
                <appinfo>

```

```

        <ipdr:enumMeaning>
            Cable
        </ipdr:enumMeaning>
    </appinfo>
</annotation>
</enumeration>
<enumeration value="2">
    <annotation>
        <appinfo>
            <ipdr:enumMeaning>
                Non cable
            </ipdr:enumMeaning>
        </appinfo>
    </annotation>
</enumeration>
</restriction>
</simpleType>
</element>
</sequence>
</complexType>
</element>

<element name="applicationMode">
    <complexType>
        <sequence>
            <element ref=" createTime"/>
            <element name="appID" type="int"/>
            <element name="appType"/>
            <element name="mode">
                <simpleType>
                    <restriction base="int">
                        <enumeration value="1">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        CROSS_ENVIRONMENT
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="2">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        BACKGROUND
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                        <enumeration value="3">
                            <annotation>
                                <appinfo>
                                    <ipdr:enumMeaning>
                                        NORMAL
                                    </ipdr:enumMeaning>
                                </appinfo>
                            </annotation>
                        </enumeration>
                    </restriction>
                </simpleType>
            </element>
        </sequence>
    </complexType>
</element>

<element name="application">
    <complexType>

```

```

    <sequence>
      <element ref="createTime"/>
      <element name="privateData" type="hexBinary">
        </element>
      </sequence>
    </complexType>
  </element>

<complexType name="OCAP-Type">
  <complexContent>
    <extension base="ipdr:IPDRType">
      <sequence>
        <element ref="protocolVersionMajor"/>
        <element ref="protocolVersionMinor"/>
        <element ref="hostID"/>
        <element ref="hostType"/>
        <element ref="createTime"/>
        <element ref="transmitTime"/>
        <element ref="privateData"/>
        <element ref="serviceSelection" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="tuning" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="SDV" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="servicePresentation" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="segmentPresentation" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="DPI" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="inputEvent" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="applicationLoadRequested" minOccurs="0"
maxOccurs="unbounded"/>
        <element ref="applicationLoaded" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="applicationLifecycle" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="environmentSelection" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="applicationMode" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="application" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
</schema>

```

## Annex B Schema Data Element Requirements (Normative)

This specification may be referenced by [ETV], [OCAP 1.1], or other platforms.

Where semantics for the data elements of Annex A may be consistent across all platforms, they are defined in this section.

Where semantics for the data elements of Annex A are platform specific, a platform definition that includes this data model specification is expected to further define the element. In such cases, Platform Specific Semantics will further define an element.

### B.1 protocolVersionMajor

This 8-bit field SHALL indicate the major version of this specification. This field SHALL have the value *1*.

### B.2 protocolVersionMinor

This 8-bit field SHALL indicate the minor version of this specification. This field SHALL have the value *0*.

### B.3 hostID

This string field SHALL indicate a unique host identifier. This value SHALL be defined by Platform Specific Semantics.

### B.4 hostType

This enumerated integer field SHALL indicate the Host type. This value SHALL be one of: *1* for OCAP, *2* for ETV, or *3* for other, as selected by Platform Specific Semantics.

### B.5 createTime

This dateTimeMsec field SHALL indicate when the metrics log was created or last flushed by the platform.

### B.6 transmitTime

This dateTimeMsec field SHALL indicate when the metrics log was last transmitted to the headend by the platform.

### B.7 privateData

This hexBinary field MAY be generated by the platform for private use.

### B.8 serviceSelection

This complex field SHALL indicate the time when a new service has been selected, together with the integer Source ID of that new service.

The *sourceID* value SHALL be defined by Platform Specific Semantics.

## B.9 tuning

This complex field SHALL indicate the time when an RF tuning action has been performed, together with the integer number of the tuner that performed the tuning operation, the integer frequency of the MPEG-2 transport stream tuned to, and the integer program number of the MPEG-2 transport stream tuned to.

The *tunerID* value SHALL be defined by Platform Specific Semantics.

The *frequency* value SHALL be equal to the RF frequency acquired by the receiver.

The *programNumber* value SHALL be equal to the MPEG-2 Program Number acquired by the receiver.

## B.10 SDV

This complex field SHALL indicate the time when a new Switched Digital Service has been selected, and the integer Source ID of that new service.

The *sourceID* value SHALL be defined by Platform Specific Semantics.

## B.11 servicePresentation

This complex field SHALL indicate the time when presentation of a new service has started, together with the integer Source ID of the service being presented, the enumerated integer aspect ratio of the display area of the presentation, the integer resolution of the display area of the presentation, and the enumerated integer indication of whether audio is on or off for the service being presented.

The *sourceID* value SHALL be defined by Platform Specific Semantics.

The *aspectRatio* value SHALL be one of: 1 for 4:3, 2 for 16:9.

The *resolution* value SHALL be defined by Platform Specific Semantics.

The *audio* value SHALL be one of: 0 for audio off, 1 for audio on.

## B.12 segmentPresentation

This complex field SHALL indicate the time when a new segmentation descriptor is detected within a presentation by the platform, together with the integer Segment ID of that new segment as taken from that descriptor. Presentations MAY be from broadcast, from a DVR recording, from SDV, or from an On-Demand session.

The *segmentID* value SHALL be defined by Platform Specific Semantics.

## B.13 DPI

This complex field SHALL indicate the time when an existing service has been replaced by new Digital Program Insertion service, together with the integer Source ID of the service being replaced, the integer Source ID of the new destination service, the enumerated integer source of the replacement, and the integer switch level of the replacement.

The *sourceID* and *replacementID* values SHALL be defined by Platform Specific Semantics.

The *replacementSource* value SHALL be one of: 1 for In-transport, 2 for Out-of-transport, 3 for From-Memory

The *switchLevel* value SHALL be one of: 0 for L0 switch, 1 for L1 switch.

## B.14 inputEvent

This complex field SHALL indicate the time when a key click from a remote control or other input device has been detected by the platform, together with the integer KeyCode value itself.

The *keyCode* value SHALL be defined by Platform Specific Semantics.

## B.15 applicationLoadRequested

This complex field SHALL indicate the time when the platform has received a request to load an application, together with an integer value that identifies the application and an enumerated integer application type.

The *appID* value SHALL be defined by Platform Specific Semantics.

The *appType* value SHALL be one of: 1 for OCAP, 2 for ETV, as selected by Platform Specific Semantics.

## B.16 applicationLoaded

This complex field SHALL indicate the time when the platform has successfully loaded an application, together with an integer value that identifies the application, an enumerated integer application type, an integer value that specifies the number of resource files that have currently been loaded for the application, the enumerated integer transport load mechanism, and the enumerated initial state of the application.

The *appID* value SHALL be defined by Platform Specific Semantics.

The *appType* value SHALL be one of: 1 for OCAP, 2 for ETV, as selected by Platform Specific Semantics.

The *numResourceFiles* value SHALL be defined by Platform Specific Semantics.

The *transportProtocol* value SHALL be one of: 1 for In-Band-Object-Carousel, 2 for Out-Of-Band-Object-Carousel, 3 for HTTP, as selected by Platform Specific Semantics.

The *controlCode* value SHALL be one of: 1 for LOADED, 2 for PAUSED, 3 for RUNNING, as selected by Platform Specific Semantics.

## B.17 applicationLifecycle

This complex field SHALL indicate the time when the platform has modified the state of an application, together with an integer value that identifies the application and an enumerated integer application type.

The *appID* value SHALL be defined by Platform Specific Semantics.

The *appType* value SHALL be one of: 1 for OCAP, 2 for ETV, as selected by Platform Specific Semantics.

The *lifeCycle* value SHALL be one of: 1 for LAUNCHED, 2 for PAUSED, 3 for RESUMED, 4 for TERMINATED, as selected by Platform Specific Semantics.

## B.18 environmentSelection

This complex field SHALL indicate the time when the platform has successfully switched to a new operating environment, together with the enumerated integer environment identifier.

The *environment* value SHALL be one of: 1 for Cable, 2 for Non-Cable, as selected by Platform Specific Semantics.

## B.19 applicationMode

This complex field SHALL indicate the time when the platform has modified the mode of an application, together with an integer value that identifies the application and an enumerated integer application type.

The *appID* value SHALL be defined by Platform Specific Semantics.

The *appType* value SHALL be one of: 1 for OCAP, 2 for ETV, as selected by Platform Specific Semantics.

The *mode* value SHALL be one of: 1 for CROSS-ENVIRONMENT, 2 for BACKGROUND, 3 for NORMAL, as selected by Platform Specific Semantics.

## B.20 application

This complex field SHALL indicate the time when an application has conveyed a private data set to the platform, together with the binary data itself. The hexBinary *privateData* value SHALL be forwarded to the metrics collection without further interpretation.

## **Appendix I      Acknowledgements**

We wish to thank the vendor participants contributing directly to this document:

Amit Kleinmann of Amdocs

Steve Cotton of IPDR.org

## Appendix II      Revision History

The following ECN was incorporated into I02 of this specification.

<b>EC Identifier</b>	<b>Date Accepted</b>	<b>Title of EC</b>
Metrics-N-07.0992-1	3/27/07	Break requirements out of schema into new section

---