IPDR データのエンコーディング形式

1. 背景・目的

OSSI3.0 に規定される IPDR Collector の開発・構築・運用には、IPDR データエンコーディング形式の理解が必要となる。

本 Tips では、IPDR データの構造、IPDR データ型、レコード構造及び、IPDR.org の Java API について説明する。

2. 対象読者

ネットワーク事業者向け業務アプリケーションの設計者、プログラム開発者

3. 参考文献・関連文書

DOCSIS OSSI 1.1/2.0/3.0 (www.cablelabs.com)

IPDR/SP Protocol Specification Version 2.1, November 2004

IPDR/XDR File Encoding Format, Version 3.5.1, IPDR.org, November 2004

4. その他

本 Tips 中の図表番号につき、OSSI からの抜粋には原文の番号をそのまま流用し、独自に作成した図表には”Tips-*”の形式で番号を付与した。

5. 最終更新日

2011 年 5 月 19 日

文書番号: OSSBN-TIPS-11-05-001/01

6. 詳細

6.1 IPDR データの構造

IPDR データは、データ構造を表わすメタデータである IPDR サービス定義と、IPDR データ本体により構成される。IPDR データの構造を Figure 6-1-1 に示す。

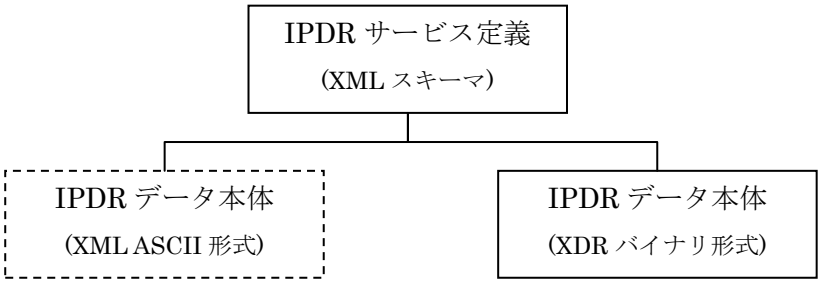

Figure 6-1-1 IPDR データの構造

IPDR サービス定義と IPDR データ本体は分離され、IPDR データ本体中の各イベントレコードには値のみが格納される。IPDR データ本体の保存形式は、XML と XDR から選択する。

XML の ASCII 文字列表現に対し、XDR は値をバイナリ形式で表現する。XML に比し、XDR では IPDR データ本体のサイズを削減できるため、読み書き処理の速度向上、通信処理の高速化、保存領域の省サイズ化を期待できる。

XML, XDR の双方共、IPDR データ本体はレコード単位で構造化されており、デコードアプリケーションはファイルを最後まで読み込まず、読み込んだ分の処理を都度実行できる。エンコード時も同様であり、全レコードの一括処理が不要のため、エンコード・デコード処理に必要な一時バッファ目的のメモリ量を節約できる。

XML エンコード例を示す。空白・改行を含めない場合、219 バイトとなる。

```
<IPDR xsi:type="AA-Type">
<subscriberId>joe</subscriberId>
<ipAddress>192.168.2.64</ipAddress>
<nasIdentifier>nas1.foo.com</nasId>
<acctInputOctets>13444</acctInputOctets>
<acctOutputOctets>77777</acctOutputOctets>
</IPDR>
```

同じ情報を XDR エンコードした場合、45 バイトすなわち、約 1/5 サイズとなる。

00 00 00 02 00 00 00 01 FF FF FF FF 00 00 00 03

j o e C0 98 40 20 00 00 00 0C n a s 1 .

f o o . c o m 34 84 00 01 2F DF

IPDR データ本体を格納する IPDRDoc ファイルの内部構造を Figure 6-1-2 に示す。

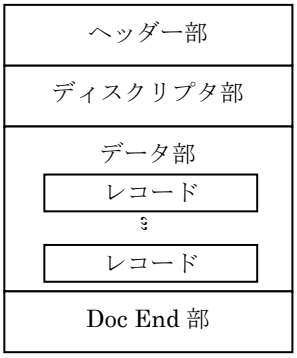

Figure 6-1-2 IPDRDoc ファイルの内部構造

Figure 6-1-2 に示す IPDRDoc ファイルは、Exporter から Collector への IPDR/SP ストリームとは形式が異なる。IPDR/SP では、Data メッセージタイプにより送信される対象情報は、Figure 6-1-2 のデータ部に限定され、ヘッダー部、ディスクリプタ部、Doc End 部の情報は、他のメッセージタイプから取得、ないしは Collector 自身が生成する。
Figure 6-1-2 のファイル形式は、Collector が収集した IPDR データをローカルファイルシステムに保存する場合及び、他の業務アプリケーションに情報を提供する場合に使用される。
本 Tips は、Exporter から送信される IPDR データの、Collector クライアントによる受信・デコードを目的としているため、ヘッダー部、ディスクリプタ部、Doc End 部の構造定義、格納値、IPDR 定義とのマッピングについては、説明を省略する。
これらの詳細な仕様については、DOCSIS OSSI3.0 の該当箇所を参照されたい。

6.2 IPDR データ型

ディスクリプタ部には、データ部の各レコードの項目名である attributeName 及び、データ型を表わす typeId のペアが、配列形式で descriptorId と共に格納される。

typeId の一覧を Figure 6.2 に示す。

分類	typeId (16 進)	typeId (10 進)	データ型	備考
プリミティブ	0x00000021	33	int	32 ビット符号付整数。 値範囲: -2147483648 ~ 2147483647 1 の 16 進表現: 00000001 -2 の 16 進表現: FFFFFFFE
	0x00000022	34	unsignedInt	32 ビット符号無整数。 値範囲: 0 ~ 4294967295
	0x00000023	35	long	64 ビット符号付整数。
	0x00000024	36	unsignedLong	64 ビット符号無整数。
	0x00000025	37	float	32 ビット浮動小数点整数。
	0x00000026	38	double	64 ビット不動小数点整数。
	0x00000027	39	base64Binary	ベース 64 バイナリ。
	0x00000027	39	hexBinary	16 進バイナリ。
	0x00000028	40	string	可変長文字列。 RFC1832 準拠、但しパディングなし。
	0x00000029	41	boolean	8 ビット真偽子。0: 偽 1: 真
	0x0000002a	42	byte	8 ビットバイト。 値範囲: -128 ~ 127 -1 のバイト表現: FF
	0x0000002b	43	unsignedByte	8 ビットバイト。 値範囲: 0 ~ 255 255 のバイト表現: FF
	0x0000002c	44	Short	16 ビット符号付整数。
	0x0000002d	45	unsignedShort	16 ビット符号無整数。
派生型	0x00000122	290	DateTime	日付時刻型。
	0x00000224	548	ipdr:dateTimeMsec	ミリ秒精度の日付時刻型。
	0x00000322	802	ipdr:ipV4Addr	IPv4 アドレス。
	0x00000427	1063	ipdr:ipV6Addr	IPv6 アドレス。
	0x00000527	1319	ipdr:uuid	UUID。
	0x00000623	1571	ipdr:dateTimeUsec	マイクロ秒精度の日付時刻型。
	0x00000723	1827	ipdr:macAddress	MAC アドレス。

Figure 6.2 typeId の一覧

IPDR/SP は、Figure 6.2 の typeId 定義に従い、通信・デコード処理を行う。

typeId は 32bit Integer で表現される。typeId の上位 2 バイトはそれぞれ、「構造化」フラグ、「アレイ」フラグとして使われる。

派生型の場合、例えば dateTime の「122」の「1」が派生型の typeId、「22」が dateTime のプリミティブ表現すなわち unsignedInt を表わす。

なお Collector アプリケーションを Java によりプログラミングする場合、Java では符号無整数がプリミティブとしてサポートされておらず、値によっては型変換を要するので注意されたい（例: unsignedInt を long にキャスト、等）。

6.3 IPDR データ部のレコード構造

IPDR データ部のレコード構造を Figure 6.3 に示す。

descriptorId は、ディスクリプタ部に格納される IPDR/SP のテンプレートに等価な情報を一意に指定する ID であり、格納データの項目順・項目名・データ型を表わす。

IPDR/SP の行番号である SequenceNum が含まれない点に注意されたい。

項目名	データ型	説明
descriptorId	int	ディスクリプタ ID。 各レコードに適用される構造定義を指定する。 ディスクリプタ部に定義済の ID のみ指定可。
IPDRRecordData	Data	レコードデータ。

Figure 6-3 IPDR データ部のレコード構造

6.4 IPDR.org の Java API

IPDR の規格標準化を推進した IPDR.org は、IPDRDoc のエンコード・デコード処理を C および Java API の形式にパッケージングし、ネット上で無償配布している。

同 Java API は、以下の URL から入手できる。

http://en.sourceforge.jp/projects/sfnet_ipdr/

アーガイブは org.ipdr.java_2.0.0 であり、解凍すると Figure 6-4 の内容が得られる。

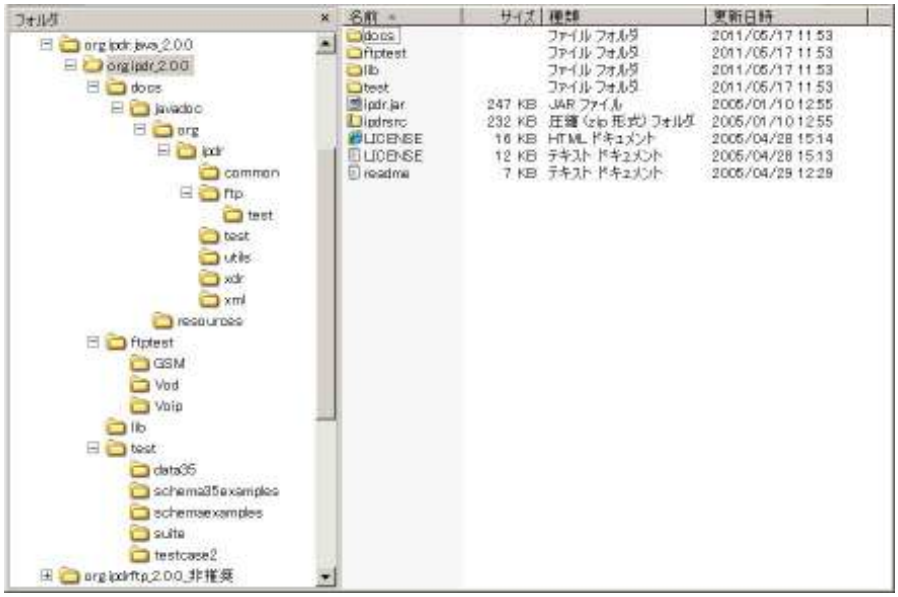

Figure 6-4 Java API の内容

org.ipdr_2.0.0 フォルダ内の LICENSE.html には、Common Public License(CPL)という規約が格納されている。CPL では、コードの二次利用、商用利用、コピー再配布を全て無償で許諾するが、利用に伴い問題が生じた場合、ライセンサーは一切責任を負わない。

org.ipdr_2.0.0 フォルダ内の readme.txt では、本 API のバージョンである 2.0.0 は IPDR NDM-U3.5 規格に準拠している事、本 API の機能は IPDR ファイルの読み書きに特化しており IPDR/SP 関連の機能は含まれない事が説明されている。

その他の格納内容は、以下の通りである。

(1) ipdr.jar

本 API の本体。
なお本 API では、XML 形式ファイルの解析処理に、Xerces Java Parser (Version 2.6.2) を使用するが、org.ipdr_2.0.0/lib/の xercesImpl.jar、xml-apis.jar に Xerces のライブラリが格納されているので、使用時にはこれらの jar もクラスパスに含める必要がある。

(2) ipdrsrc.zip

ソースコード、および、ant が使用する build.xml 形式のスクリプト。

(3) docs

本 API の Java Doc、開発指針、インストール手順、readme.txt。

(4) lib

xercesImpl.jar、xml-apis.jar。

(5) test / ftptest

API の実行コード例とサンプルデータ。

なお、本 API の実行プログラムの機能は、概ね以下の通りである。

- CSV から XDR へのファイル形式変換(IPDRDocWriter を使用)
- CSV から XML へのファイル形式変換(IPDRDocWriter を使用)
- XDR から CSV へのファイル形式変換(IPDRDocReader を使用)
- XML から CSV へのファイル形式変換(IPDRDocReader を使用)
- XML から XDR へのファイル形式変換(IPDRDocWriter と IPDRDocReader を使用)
- XDR から XML へのファイル形式変換(IPDRDocWriter と IPDRDocReader を使用)

6.5 考察

XDR ファイル形式は、大量データの保存・転送用途に優れている反面、データの取り扱いに特殊なファイル処理を必要とするため、開発工数が増加する。そこで IPDR.org では、開発上の困難を緩和するために、読み書きに特化した部分を API 化し、ネット上で無償配布する等、様々な普及促進の努力を行ってきた。

しかしながら、そもそもケーブルネットワーク上での IPDR の現実的な用途は、SNMP に比べ低負荷・高速・高信頼な特性を生かした CMTS からの情報取得に限られる。すなわち、CMTS から定期的ないしはリアルタイムに情報を収集する監視アプリケーションを、複数の SNMP マネージャによる構成から IPDR Collector クライアントに置換・集約すれば、加入者のサービストラフィックへの影響を最小限に抑制しながら、従量制課金情報やサービスの状態情報を高速に収集できる。

一方で、ケーブルシステムのネットワーク構造上、CMTS のバックエンドに位置する管理系のネットワークは、比較的低コストで高速化できる。また、継続的な技術革新により、大量データを保存・検索するストレージ技術が格段の進歩を遂げており、今日では XDR 形式による保存サイズの圧縮は、必ずしも重要ではない。

現実的には、CMTS から IPDR/SP で情報を取得後、XML 形式やテキスト・CSV 形式等、業務アプリケーションやシステム管理者がより簡単に参照できるファイル形式で、取得した情報を保存・出力する収集システムの方が、実際の OSS/BSS 環境上は使いやすい。従って、上述の IPDR.org の API ライブラリの応用は、IPDRDocWriter による XML 形式ファイルの出力処理に限定すべきである。

すなわち、上位アプリケーションに IPDR の入力インタフェースが既に実装されている場合に限り、XML 形式ファイルを出力し、上位アプリケーション向けの XML 形式ファイルの出力処理に、IPDRDocWriter を利用する。一方で、上位アプリケーションが IPDR の入力インタフェースを持たない場合、開発工数・閲覧性・保守性の観点から、テキストないしは CSV 形式のファイルを出力し、上位アプリケーションに受け渡す。

この場合のテキスト形式ファイルのレコード構造例を Figure 6-5 に示す。ファイル名には IPDR/SP の UUID を使用し、CMTS と収集タイミングを特定可能とする。項目とデータ型の情報は、別形式で受け渡すか、ファイルの先頭行に格納すれば良い。

項目名	データ型	説明
flags	char	IPDR/SP のレコードフラグ。
sequenceNum	long	IPDR/SP のレコード番号。
以降、IPDR/SP 受信データの各項目を、ASCII 文字列表現で格納する。 プリミティブ及び派生型は、Java が解釈可能な文字列リテラル表現に置換する。 各項目はタブ区切、各レコードは改行区切とする。		

Figure 6-5 テキスト形式ファイルのレコード構造例

以上