総務省

情報流通連携基盤の公共交通分野における実証に係る請負

公共交通オープンデータ化  
ガイドライン

平成26年3月20日

株式会社横須賀テレコムリサーチパーク

# はじめに

公共交通オープンデータは各交通事業者様からご提供頂いたデータから構築される。この提供頂く情報は各社毎に異なるフォーマット・ファイル形式となっており、仮に何も処理を行わず提供されたままの形式で利用者へ提供を行うと非常に扱いづらいデータとなってしまう。

公共交通オープンデータは扱う情報を分類しそれぞれに共通のフォーマットを定義することで、利用者に対して統一したフォーマットで多数の交通事業者様の情報を提供することができる。このシステムは各交通事業者様から提供して頂いたデータを公共共通オープンデータ規格に変換し、データベースで管理することで実現する。

本ガイドラインでは公共交通オープンデータ形式への変換に際し、各交通事業者様から提供して頂くデータ形式・フォーマットについて留意事項を記載する。具体的には時刻表、地物情報を例に、よくある問題点、注意事項、望ましい対応などについて述べる。

# ****時刻表****

時刻表の元となるデータは、現在２つの形式が存在する。

* CSV・Excelなどの構造化された形式
* Webページに公開されている形式

前提として、形式は異なっていたとしても情報量は変わらない、ということである。

この異なった形式のデータから、公共交通オープンデータに用いる必要な情報の抽出を行う。以下に公共交通オープンデータ規格における時刻表に必要な情報の一覧を示す。

表 1 時刻表情報項目一覧

|  |  |  |
| --- | --- | --- |
| **odpt:TrainTimetable** | **値域** | **説明** |
| @id | @id | 固有識別子 |
| @type | odpt:TrainTimetable | クラス指定 |
| [acl:memberOf](http://aclmemberof/) | acl:Dataset | データの所属するデータセットのURI |
| [dc:date](http://dctime/) | [xsd:dateTime](http://xsddatetime/) | データ生成時刻、e.g. 2013–01–13T15:10:00+0900、ISO8601形式 |
| [odpt:trainNumber](http://odpttrainnumber/) | [xsd:string](http://xsdstring/) | 列車番号 |
| [odpt:railwayName](http://odptrailwayname/) | [xsd:string](http://xsdstring/) | 路線名 |
| [odpt:operator](http://odptoperator/) | [xsd:string](http://xsdstring/) | 鉄道事業者名 |
| [odpt:weekdays](http://odptweekdays/) | Array | 平日（出発時間と出発駅の組か、到着時間と到着駅の組のリストを格納） |
| [odpt:saturdays](http://odptsaturdays/) | Array | 土曜日（出発時間と出発駅の組か、到着時間と到着駅の組のリストを格納） |
| [odpt:holidays](http://odptholidays/) | Array | 休日（出発時間と出発駅の組か、到着時間と到着駅の組のリストを格納） |
| [odpt:departureTime](http://odptdeparturetime/) | [xsd:](http://xsdstring/)dateTime | 出発時間 |
| [odpt:arrivalTime](http://odptarrivaltime/) | [xsd:](http://xsdstring/)dateTime | 到着時間 |
| [odpt:destinationStation](http://odptdestinationstation/) | [xsd:](http://xsdstring/)string | 出発駅 |
| [odpt:arrivalStation](http://odptarrivalstation/) | [xsd:](http://xsdstring/)string | 到着駅 |

@id,@typeは、それぞれ情報流通連携基盤に登録する際に指定するため、変換時に入力を行う必要はない。これらの情報を公共交通オープンデータ規格のフォーマットに変換を行う際のテンプレートを以下に記載する。

{

"@id": ID,

"@type": "odpt:TrainTimetable",

"acl:memberOf": データセットメンバー名,

"odpt:trainNumber": 列車番号,

"odpt:railwayName": 路線名,

"odpt:operator": 運営会社名,

"dc:date": データ生成時刻,

"odpt:weekdays": [

{

"odpt:destinationStation": 発駅名,

"odpt:departureTime": 発車時刻

},

{

"odpt:destinationStation": 発駅名,

"odpt:departureTime": 発車時刻

}

],

"odpt:saturdays": [

],

"odpt:holidays": [

]

}

## CSV・Excelなどの構造化された形式

CSV・Excelなどの構造化された形式は、提供元よりその構造を解説したドキュメントが提供されるので、それに従って公共交通オープンデータ規格に沿った形へ変換するスクリプトを作成する。  
以下に例としてCSVとして提供された列車時刻表を示す。

# 元データ例

2103G,大崎,大崎,平日

大崎,,21:03

五反田,,21:05

目黒,,21:08

....

大崎,21:57,

上記例であると、仕様書からそれぞれのカラムに入った情報は以下の意味があると仮定する。

列車番号,始\_=95\_\_発駅,終着駅,運行日

駅名,着時刻,発車時刻

駅名,着時刻,発車時刻

駅名,着時刻,発車時刻

...

駅名,着時刻,発車時刻

このそれぞれのカラムの意味を使って、公共交通情報オープンデータ規格へ変換を行う。

// 加工結果

{

"@id": "urn:urn:50eb1c3f-bfab-4a0b-a15a-3a990577fc8d",

"@type": "odpt:TrainTimetable",

"acl:memberOf": "JR-East",

"odpt:trainNumber": "2103G",

"odpt:railwayName": "山手線",

"odpt:operator": "東日本旅客鉄道",

"dc:date": "2012-10-15T17:48:53+09:00",

"odpt:holidays": [

{

"odpt:destinationStation": "大崎",

"odpt:departureTime": "21:03"

},

{

"odpt:destinationStation": "五反田",

"odpt:departureTime": "21:05"

},

...

}

それぞれの情報に対して、情報の意味（例:“odpt:trainNumber”）をキーとして、値（例:“2103G”）を格納している。CSVに公共交通オープンデータ規格として不足している情報がある場合、変換時にメタデータの付与を行う。この操作は予めメタデータを作成しておけば作業量としてさしたる問題にはならない。

### 元データ作成時における注意点

元データがたとえCSV・Excelだとしても適切な形で情報が入力されていないと、変換に際して非常にコストが高くなり、最悪の場合変換が行えない。以下に元データ作成時における留意点を挙げる。

* 仕様書には全てのパターンを記載する
  + 見ただけでわかる部分もあるが、接続・種別・時間計算などのルールが明文化されていなケースがあり、正確なデータの変換には全てのパターンを網羅した形での仕様書が必要
* 表の枠線色、枠線の太さ、フォント色、フォントサイズに意味を持たせない
  + Excel形式でよく見られるが、装飾に意味を持たせると変換が非常に困難になり、最悪の場合変換が行えない
  + 見た目での情報識別は人間には見やすいが、コンピュータによる処理は非常に困難
* CSVで全ての情報を伝えられるフォーマットが望ましい
  + 装飾無しで、二次元の表情報としてだけで全ての情報を伝えられるのが望ましい
* CSVに三次元以上の情報を持たせない
  + 例）一行目は駅の詳細情報、二行目以降はその駅の発着時刻 ＝ 三次元分のデータを1つのCSVファイルで保持している
  + 1つのCSVで表現すべきは二次元のデータ（＝縦軸x横軸）まで
  + 複数の次元に跨る情報は、情報毎に固有IDをもたせ、その固有IDをキーとしてJOIN（結合処理）を行うようにする
  + 例）駅に対して駅IDを付与し、駅IDと駅名の対応一覧、駅の詳細情報、駅の発着時刻情報という３つのCSVに分割して情報を記述する
  + このような形式であると既存のRDB（Relational DataBase）へのインポートも容易になる

## ****Webページに公開された形式****

Webページに公開された形式は、予め提供元（運営事業者様）へ情報取得・加工の許諾を頂き、その上でHTML(Hyper Text Markup Language)から公共交通オープンデータ規格に沿った形へ変換を行った。　以下に全体的な流れを記載する。

1. 提供元WebページからHTMLを取得（クローリング）
2. HTMLから抽出を行う（スクレイピング）
3. 公共交通オープンデータ規格へ変換する
4. 情報流通連携基盤へデータを登録する

内部的な処理は、HTML/XHTML/XMLから情報を抽出する際に一般的に用いられるXPath（XML Path Language）と呼ばれるマークアップ文章の特定部分を指定する言語を用いて、必要な情報の抽出を行った。以下にこの変換の例を挙げる。

// 元データ例

<table class="train-timetable">e

<colgroup><col class="first">

<col class="second">

</colgroup><thead>

<tr>

<th scope="row" colspan="2" class="style1">列車番号</th>

<th scope="col" class="">5701</th>

....

<th scope="col" class="rapid">3001</th>

<th scope="col" class="">5201</th>

<th scope="col" class="semirapid">4001</th>

<th scope="col" class="">5007</th>

<th scope="col" class="semirapid">4003</th>

<th scope="col" class="">5203</th>

<th scope="col" class="">5205</th>

<th scope="col" class="semirapid">4005</th>

<th scope="col" class="">5009</th>

</tr>

<tr>

<th scope="row" colspan="2" class="style1">列車種別</th>

<td class="">普通</td>

<td class="">普通</td>

<td class="">普通</td>

<td class="">普通</td>

<td class="">普通</td>

<td class="rapid-inverted">快速</td>

<td class="">普通</td>

....

<th scope="row">着</th>

<td class="">5:37</td>

<td class="">6:06</td>

<td class="">6:27</td>

<td class="">6:43</td>

<td class="">6:59</td>

<td class="rapid">6:50</td>

<td class="">・・</td>

<td class="semirapid">7:11</td>

<td class="">7:29</td>

<td class="semirapid">7:23</td>

<td class="">・・</td>

<td class="">・・</td>

<td class="semirapid">7:38</td>

<td class="">7:53</td>

</tr>

</tbody>

</table>

# 加工結果

"@id": "urn:urn:50eb1c3f-bfab-4a0b-a15a-3a990577fc8c",

"@type": "odpt:TrainTimetable",

"acl:memberOf": "TX",

"odpt:trainNumber": "5201",

"odpt:railwayName": "つくばエクスプレス",

"odpt:operator": "首都圏新都市鉄道",

"dc:date": "2012-10-15T17:48:53+09:00",

"odpt:holidays": [

{

"odpt:destinationStation": "つくば",

"odpt:departureTime": "5:37"

},

### 元データ作成時における注意点

元データがWebページに場合、基本的にはヒトが見やす形でデザインされたHTMLである。HTMLはMarkup Language、すなわち構造化文章を作成するための規約であるが、誤った構造化が行われているといった場合がある。ここではHTMLを元データとする場合の作成時における留意点を挙げる。

* 装飾は全てCSS（Cascading Style Sheets）に記載する
* h1,h2,h3などのヘッダータグは適切に用いる
  + 文字を大きくするためだけにh1を使うなどはしない
* タグにはid、classを指定し、そのタグがどのような意味を持つかを客観的に述べることができるのが望ましい
  + 発車時刻が記載されているタグにはid="depTime"、駅名が記載されているタグにはid="stationName"など
  + これらのタグの対応が仕様として記載されているだけで、作業効率は大幅に上がる
* 特殊文字に意味を持たせない・使用しない
  + ①、■、▲など（列車種別を示す際に用いられる例が散見される）
* 装飾に意味を持たせない
  + 例）赤文字は特急、など
  + タグに対してid、classを用いてメタデータとして意味を持たせるのが望ましい

## 元データにおける望ましい記述例

先に挙げた２つの例においては、記述形式が異なるだけで記述方法（ルール）は同じものといえる。以下に望ましくない例、望ましい例を同じデータを用いて示す。



図 1 望まくない記述例

### 

図 2 望ましい記述例

# 地物情報

地物情報は、運営事業者様よりご提供頂いた店舗、トイレ、ロッカーといった地物を地図上にプロットし、メタデータを入力することで  
GIS（Geographic Information System）におけるPOI（Point Of Interest）としてデータを作成した。以下にその手順を挙げる。

* 店舗、トイレ、ロッカーなどの情報を集める
* GISソフトウェア上で実際の位置に合わせてPOIを作成する
* 作成したPOIに対して名称、注記などのメタデータを入力する
* 作成した情報を情報流通連携基盤へ登録する

この他に構内地図を作成するにあたり、図面・案内図を提供していただくケースがある。この場合、地理座標系が埋め込まれたベクターデータ（例:Shapeデータ）であれば変換を行う必要はないが、設計図面、パンフレット等で用いる案内図には地理座標系が埋め込まれていないため実際の地理座標系に合わせる必要が生じる。この変換には図中で正確に対応する地理座標情報（緯度経度）が最低3点、可能であれば8点必要となる。

## 元データ作成時における注意点

* 地理座標系を合わせる
  + 基本的にWGS84を用いる。しかしWGS84と異なった座標系であってもほぼ正確な変換は可能なため、仕様書に使用している地理座標系を記載する
* CSV、Excel等の形式の場合、1つの列に対して1つの情報を割り当てる
  + 以下に挙げる例のように、二次元に収まる形で情報を作成する
* 機種依存文字・外字は使用しない
  + 文字コードには国際標準となっているUTF-8を使用することが望ましい

表 2 CSV、Excelによる地物情報の望ましい記述例

