図書館目録のリンクトデータにおけるデータ由来情報と管理情報
RDA in RDF、BIBFRAME、Wikidata の再検討

谷口 祥一（慶應義塾大学文学部）
taniguchi@z2.keio.jp

図書館目録のリンクトデータにおいて、データ要素値の由来情報・付加情報、Nomen の付加情報、データ管理情報をどのように RDF モデル化し表現すべきか検討した。先ず、対象となる情報の種別を 3 つに区分した上で、単一ステートメント単位か記述セット単位かという記録の単位との組み合わせを整理した。次に、記録の単位ごとに妥当な RDF 表現方式を列挙した。続けて、RDA in RDF、BIBFRAME、Wikidata における、こうした情報の扱いを確認し、併せて課題を明らかにした。

１．はじめに

図書館目録として形成されているメタデータをリンクトデータにして公開し提供することが、部分的にせよ行われている。そのためには、情報資源とそれらが形成する事象を捉えるための概念モデルや、それに依拠した RDF による必要な語彙（クラスとプロパティ）定義などが前提となる。これらの実施例や検討例は数多く存在し、研究の蓄積もある。例えば、わが国では、NDL による DC-NDL、NII による CiNii RDF 語彙、JLA 目録委員会による「日本目録規則 2018 年版」の RDF 語彙などを挙げることができる。

他方、そうした図書館目録のメタデータの中核部分ではないが、それに包含されるデータ要素に、記録されたデータ要素値の由来情報・付加情報、Nomen の付加情報、データ管理情報など、メタレベルの情報（捉え方によってはメタメタレベルの情報）がある。これらの扱いは現時点では多様であり、また十分には記録されてはいないという状況がある。なお、ここには対象リソース自体の所有・管理の経歴などの由来情報、保管形態や保管処理状況などの管理情報は含めていない。

本発表は、データ値の由来情報・付加情報やデータ管理情報などがどのように RDF によってモデル化され表現されるのか、妥当な方式（モデルパターン）を列挙し、併せて既存モデルや語彙における扱いを検討する。

図書館目録のリンクトデータ化において、こうした問題に焦点を当てた研究はない。RDA では 3R プロジェクトの一環として data provenance の記録法について果敢な取り組みがなされた結果、現行 RDA ではそうした記録法が広範に規定化されている [1]。しかし、RDF によるモデル化と表現という観点から十分に検討されているのか不明であり、本発表はそうした点からの再検討でもある。

２．対象となる情報の種別整理と記録の単位

対象とするメタレベル情報は、以下の 3 種に区分することができる。

a) データ要素値の由来情報・付加情報：記録された値の情報源、値の言語や文字種、値の有効期間や範囲、値の編成規則や転記規則、加えてこれら以外の注記など。これらは個別の RDF トリプルである記述ステートメント」（description statement）ごと、あるいはその集合体である「記述セット」（description set）を対象にして記録される。

b) メタデータの管理情報：記述セットとしてのメタデータの作成日付・更新日付、作成者・更新者・公開者、そして注記など。メタデータ全体にわたる編成規則や基準などを、管理情報に位置づけることもある。

c) Nomen の付加情報：Nomen とは実体インスタンスを指し示す呼称（appellation）である。その Nomen の情報源、言語や文字種、有効期間や範囲、編成規則や転記規則、付与者、そしてこれら以外の注記などがメタレベル情報となる。これは個別の Nomen ごとに記録されることになるため、記述ステートメント単位の記録となる。

３．適用可能な RDF データ表現の方式

RDF reification（具体化）、Named graph（名前付きグラフ）、n-ary relation（多項関連）など、RDF の基本 3 要素によるトリプル表現を超えた表現法が基本的に適用可能である。

３．１　記述ステートメントレベルの方式

記述ステートメントに対するメタレベル

情報を RDF データとして表現する方式には、以下が妥当な方式として挙げられる。

方式１：データ値を構造化して表現する。すなわち、通常の記述ステートメントを構成するプロパティの値を、構造化した値の表現に展開する。そのためには、リテラル値の場合にも、値のクラス（例：DescriptionValue、Nomen など）を想定した上で、それを明示的に使用する場合と使用しない場合を許容する。

```
ex:対象リソース URI_1
  rdf:type　:対象リソースクラス ;
  :記述プロパティ１ ［
    rdf:type　:値クラス ;
    rdf:value　"リテラル値" ;
    :メタレベルプロパティ１　"リテラル値" ;
    :メタレベルプロパティ２　ex:URI_2 ］.
```

値クラスが Nomen の場合には、rdf:value ではなく、rdfs:label がより適切となる。当該方式では、記述プロパティの値がリテラル値または URI ではなく、空白ノードを用いた構造化表現であるため、検索などの操作における手間が増えるという副作用がある。

方式 2：プロパティ「prove/記述プロパティ」と provenanceFor、クラス Provenance を導入し、対象リソースの記述データとメタレベルデータとを切り分けて表現する。リテラル値の場合には、値が重複して記録される結果となる。

```
ex:対象リソース URI_1
  rdf:type　:対象リソースクラス ;
  :記述プロパティ１　"リテラル値" ;
  :prove/記述プロパティ１ ［
    rdf:type　:Provenance ;
    rdf:value　"リテラル値" ;
    :メタレベルプロパティ１　"リテラル値" ;
    :メタレベルプロパティ２　ex:URI_2 ］.
```

方式 3:方式 2 の変形であり、値の 2 層構造表現とも捉えることができる。

```
ex:対象リソース URI_1
  rdf:type　:対象リソースクラス ;
  :記述プロパティ１　_:b1 ;
  :prove/記述プロパティ１ ［
    rdf:type　:Provenance ;
    :provenanceFor　_:b1 ;
    :メタレベルプロパティ１　"リテラル値" ;
    :メタレベルプロパティ２　ex:URI_2 ］.
_:b1
  rdf:type　:値クラス ;
  rdf:value　"リテラル値" .
```

方式 4:RDF が規定する Reification（具体化）の方式を適用する。プロパティ rdf:subject、rdf:predicate 、 rdf:object と クラス rdf:Satement を用いてメタレベルの情報を表現する。

```
[]　rdf:type　rdf:Statement ;
  rdf:subject　ex:対象リソース URI_1 ;
  rdf:predicate　:記述プロパティ１ ;
  rdf:object　"リテラル値" ;
  :メタレベルプロパティ１　"リテラル値" ;
  :メタレベルプロパティ２　ex:URI_2 .
```

３．２　記述セットレベルの方式

記述セットに対するメタレベル情報を表現する方式には、以下が挙げられる。

方式 5：由来情報・付加情報の場合には、プロパティ provenance と provenanceFor、クラス Provenance を導入し、管理情報の場合には adminMetadata 、adminMetadataFor と 、AdminMetadata を導入する。provenance と adminMetadata の主語は、メタデータ URI ではなく、対象リソースの URI とする。クラス Provenance、AdminMetadata のインスタンスの下にメタレベルの情報を配置する。

```
ex:対象リソース URI_1
  rdf:type　:対象リソースクラス ;
  :記述プロパティ１　"リテラル値" ;
  :記述プロパティ２　ex:URI_2 ;
  :provenance　［
    rdf:type　:Provenance ;
    :provenanceFor　ex:対象リソース URI_1 ;
    :メタレベルプロパティ１　"リテラル値" ;
    :メタレベルプロパティ２　ex:URI_3 ］;
  :adminMetadata　［
    rdf:type　:AdminMetadata ;
    :adminMetadataFor　ex:対象リソース URI_1 ;
    :メタレベルプロパティ３　"リテラル値" ;
    :メタレベルプロパティ４　ex:URI_4 ］.
```

方式 6：名前付きグラフ（Named graph）を導入し、それを主語にしてメタレベルのプロパティを適用する。由来情報・付加情報と管理情報のいずれも同じ方式で表現可能である。また、グラフ URI は空白ノードとすることもできる。TriG によるシリアライゼーションを適用して表現すると下記のようになる。

```
ex:グラフ URI_1
{ex:対象リソース URI_1
   rdf:type　:対象リソースクラス ;
   :記述プロパティ１　"リテラル値" ;
   :記述プロパティ２　ex:URI_2 .}
ex:グラフ URI_1
  :メタレベルプロパティ１　"リテラル値" ;
  :メタレベルプロパティ２　ex:URI_3 ;
  :メタレベルプロパティ３　"リテラル値" ;
  :メタレベルプロパティ４　ex:URI_4 .
```

現在審議中の RDF-star などを、上記に代えて適用することもできる。

方式 7：方式 5 と 6 の組み合わせであり、名前付きグラフに加えて、プロパティ provenance と provenanceFor、クラス Provenance、ある

いはプロパティ adminMetadata と adminMetadataFor、クラス AdminMetadata を導入する。

```
ex:グラフ URI_1
{ex:対象リソース URI_1
   rdf:type   :対象リソースクラス ;
   :記述プロパティ 1   "リテラル値" ;
   :記述プロパティ 2   ex:URI_2 .}
ex:グラフ URI_1
  :provenance [
     rdf:tye   :Provenance ;
     :provenanceFor   ex:グラフ URI_1 ;
     :メタレベルプロパティ 1   "リテラル値" ;
     :メタレベルプロパティ 2   ex:URI_3 ] ;
  :adminMetadata   [
     rdf:type   :AdminMetadata ;
     :adminMetadataFor   ex:グラフ URI_1 ;
     :メタレベルプロパティ 3   "リテラル値" ;
     :メタレベルプロパティ 4   ex:URI_4 ] .
```

４．既存モデル・語彙の検討

４．１　RDA in RDF

RDA は 3R プロジェクトの結果、data provenance の記録法が広範に規定化されている。ガイダンス Data provenance、そして Nomen についてはガイダンス Nomens and appellations のセクション Data provenance of nomens が整備され、さらに個別のエレメントのインストラクション内において規定化が図られている [1]。ここには由来情報・付加情報のみではなく、管理情報も含まれている。なお、RDA は RDF によるエンコーディング、そしてリンクトデータ化のみをその実装シナリオとしているわけではないが、本発表では RDF 適用を前提とする。なおかつ、canonical プロパティ（定義域の指定あり、値域の指定なし）の採用を前提とする。

RDA では対象リソースに対するメタデータを metadata work と名付け、著作の一種に位置づけている。定義によると、metadata work は、記述ステートメントの場合と記述セットの場合の両者を含む。ここでの記述ステートメントの主語は、実体インスタンス URI である。

こうした metadata work に関わるプロパティは 4 種に区分することができる。

a) data provenance の記録のみに使用されるプロパティ（定義域はいずれも著作）。note on metadata（rdaw:P10402 ; メタデータに関する注記）、recording source（rdaw:P10404 ; 記録の情報源）、scope of validity（rdaw:P10403 ; 有効範囲）、source consulted（rdaw:P10406 ; 参照情報源。値域は体現形）。

b) 汎用的なプロパティであり、data provenance の記録にも使用可能なプロパティ。①author agent（rdaw:P10061 ; メタデータの作成者。定義域は著作）、publisher agent（rdam:P30083 ; メタデータの公開者。定義域は体現形）、date of publication（rdam:P30011 ; メタデータの公開日付。定義域は体現形）。②related manifestation of work（rdaw:P10309 ; 準拠した記述規定・転記規定。定義域は著作）、language of expression（rdae:P20006 ; 言語。定義域は表現形）、script（rdae:P20065 ; 文字種。定義域は表現形）など。

c) 対象リソースと metadata work とを結びつけるプロパティ。①metadata description of RDA entity（rdaw:P10622）と、RDA entity をそのサブクラスである work、agent、nomen などに置き換えたプロパティ群。定義域は metadata work である著作、値域は RDA entity など。②RDA entity described with metadata by（rdax:P00030）と、RDA entity をそのサブクラス work、agent などに置き換えたプロパティ群。定義域は RDA entity など、値域は metadata work である著作。

d) Nomen の記録において付加情報を記すプロパティ（定義域は Nomen）。note on nomen（rdan:P80071 ; nomen に関する注記）、assigned by agent（rdan:P80073 ; nomen の付与者）、scheme of nomen（rdan:P80069 ; nomen のスキーム）など、多数のプロパティが定義されている。

次に、こうした RDF 表現法について、問題点の整理などを試みる。

・metadata work を定義域とするプロパティ群 a)と一部の b)は、個別ステートメント単位では上記の方式 4、記述セット単位では方式 6 のみ適用可能である。他方、個別ステートメントごとにグラフ URI を付与し、そのメタレベル情報を記録していくことは可能とはいえ、相当程度の煩雑さを伴う。さらには、Nomen に関わるプロパティ群 d)は、方式 1 となり、metadata work とは異なる表現方式となる。

・トリプル「対象リソース URI － RDA entity described with metadata by – metadata work URI」とその逆向きのトリプルがあり、かつ metadata work 内では対象リソース URI を主語とするトリプルから構成されるという、言わば再帰に似た構成となる。

```
ex:metadataWorkURI_1
{ex:対象リソース URI_1
   rdf:type   :対象リソースクラス ;
   rdax:P00030   ex:metadataWorkURI_1 .}
```

ex:metadataWorkURI_1
  rdf:type　rdac:C10001 ;
  rdaw:P10622　ex:対象リソース URI_1 .

単一 metadata work 内に、異なる複数の主語リソースのトリプルデータがあるとき、metadata work の URI と個々の主語リソースとが関連づけられることになる。こうした構成がどのような副作用を伴うのか不明である。

・metadata work につらなる metadata expression, metadata manifestation がモデルとして想定されることになる。上記 b)に挙げられたプロパティ language of expression と script は定義域が表現形、publisher agent と date of publication の定義域は体現形である。すなわち、これらのプロパティを適用するためには、その主語リソースとして metadata expression や metadata manifestation が要請される。他方で、こうしたインスタンスとその URI を実際に設けることは煩雑過ぎる。

４．２　BIBFRAME

BIBFRAME（version 2.1.0）[2]では、RDF の適用が前提とされている。

a) 管理情報は本研究による方式 5 によって表現される。トリプル「対象リソース URI －bf:adminMetadata/bf:adminMetadataFor －クラス bf:AdminMetadata」となり、クラス bf:AdminMetadata の箇所が構造化表現される。bf:AdminMetadata を主語リソースとするプロパティには、bf:derivedFrom 、bf:descriptionConventions （値域は bf:DescriptionConventions）、bf:description Language 、bf:generationProcess （値域は bf:GenerationProcess ）、bf:description Modifier、bf:descriptionAuthentication（値域は bf:DescriptionAuthentication）がある。加えて、bf:AdminMetadata を主語リソースとする bf:creationDate 、bf:generationDate 、bf:changeDate がある。

b) 加えて、汎用的なプロパティを組み合わせて使用することができる。これらは Nomen に属するデータにも適用することができる。bf:assigner（識別子、名称、分類記号などの付与者）、bf:source（情報源。値域は bf:Source）、bf:qualifier（限定子）、bf:status（確定状況。値域は bf:Status）、bf:note（注記。値域は bf:Note）、bf:Note を定義域とする bf:noteFor、bf:noteType が該当する。bf:language も Nomen の付加情報として適用可能である。

BIBFRAME での扱いの問題点をまとめると下記のようになる。

・Nomen を含めてデータ値の由来情報・付加情報については、個別ステートメントに対する上記 b)のみ適用可能である。記述セット単位に記録することはできない。

・他方、管理情報は対象リソース URI と結びつくことになり、対象リソースのメタデータに対するものではない。BIBFRAME では、グラフ URI の採用は現時点では想定されていない。

４．３　Wikidata

Wikidata は、IFLA ワーキンググループを始め、いくつかのプロジェクト等において、図書館目録のメタデータのスキーマもしくは典拠データ管理システムとして注目されている。

Wikidata では、内部的には特有の JSON 形式で保持しているデータを、RDF 形式に変換して出力している [3]。現行方式は、RDF データの Sparql による検索が容易となるよう、主語となる wd:アイテム ID に対して、プロパティとその値をそのまま表現する「直接記述」と、各種の付加情報などメタレベルの情報を表現する「ステートメント記述」の両方を採用しており、方式 2 に該当する。

wd:アイテム ID
  wdt:プロパティ ID_1　"リテラル値"/URI ;
  p:プロパティ ID_1　wds:ステートメント ID_2 .
wds:ステートメント ID_2
  ps:プロパティ ID_1　"リテラル値"/URI ;
  psv:プロパティ ID_1　wdv:値ノード ID_3 ;
  pq:限定子 ID_4　"リテラル値"/URI ;
  pqv:限定子 ID_4　wdv:値ノード ID_4 ;
  prov:wasDerivedFrom　ref:参照ノード ID_5 .

上記のようにステートメント記述の場合には、「wds:ステートメント ID」を主語とするトリプル、さらには「wdv:値ノード ID」、「wdref:参照ノード ID」を主語とするトリプルが加わることによって、由来情報や付加情報など、メタレベルの情報が表現される。

こうした表現法により、Wikidata では広範かつ柔軟に由来情報・付加情報が記録されているが、その反面、複雑度の増加を招いている。

注・引用文献

1) RDA Toolkit. https://www.rdatoolkit.org/
2) Library of Congress. BIBFRAME Model, Vocabulary, Guidelines, Examples, Notes, Analyses. https://www.loc.gov/bibframe/docs/index.html
3) Wikibase/Indexing/RDF Dump Format. https://www.mediawiki.org/wiki/Wikibase/Indexing/RDF_Dump_Format