Materials data platform DICE

MDR Schemaの紹介と、その作成過程で得た教訓

田辺浩介 TANABE.Kosuke@nims.go.jp

物質・材料研究機構 統合型材料開発・研究基盤部門
材料データプラットフォームセンター

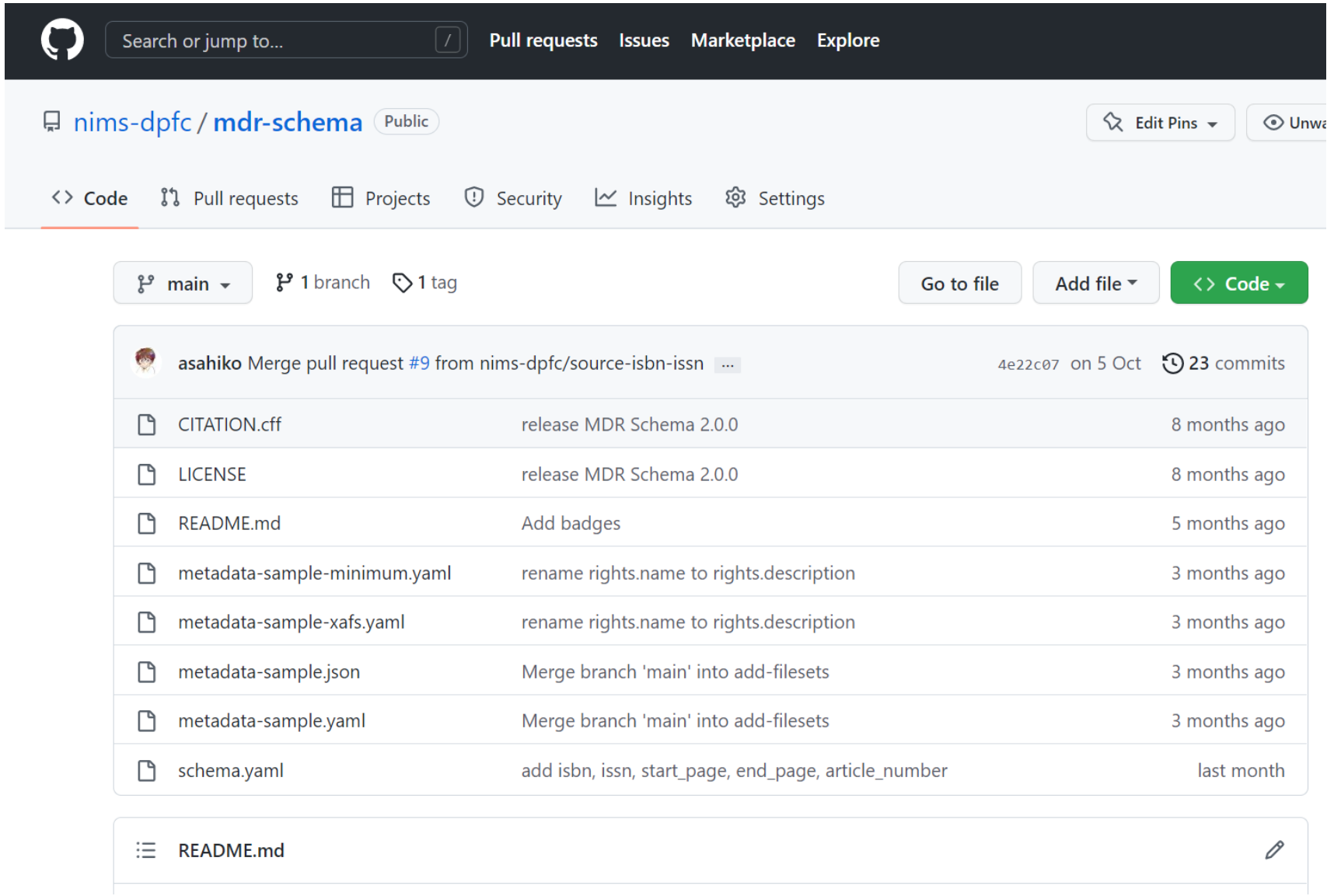
2022年11月11日 RDUF総会・公開シンポジウム

MaDIS NIMS

MDR Schemaとは

- 物質・材料研究機構のデータリポジトリ「Materials Data Repository」(MDR, https://mdr.nims.go.jp)のデータ登録に用いる簡易なメタデータスキーマ、ならびにメタデータフォーマット
- 以下の特徴を持つ
  - YAMLでの記述
  - IDの記述を活用し、メタデータの階層構造を**2階層**に限定

GitHubで公開 https://github.com/nims-dpfc/mdr-schema



MDR Schemaのスキーマ定義の例

```
159   # 人物
160   person:
161     name: str()
162     orcid: str(required=False)
163     e_rad: str(required=False)
164     organization: str(required=False)
165     department: str(required=False)
166     ror: str(required=False)
167     role: enum('author', 'editor', 'translator', 'depositor',
168
```

name, orcidなどの各項目の
データ型や必須かどうかを指定

roleに指定できる内容を指定

MDR Schemaのスキーマ定義の例

材料メタデータの指定

```
228   # 試料
229   specimen:
230     name: str()
231     description: str(required=False)
232     identifier: str(required=False)
233     material_type_vocabulary: str(required=False)
234     material_type_description: str(required=False)
235
236   ---
237   # 試料の化学組成
238   chemical_composition:
239     specimen_identifier: str(required=False)
240     identifier: str(required=False)
241     category_vocabulary: str(required=False)
242     category_description: str(required=False)
243     description: str(required=False)
```

MDR Schemaによるメタデータの記述の例

```
159  # 試料（複数記述可）
160  specimens:
161  - name: A002 #試料の名前
162    description: NiO polycrystal film サンプル8 #試料の説明
163    identifier: DCStagid-47910155 #試料のID
164
165  chemical_compositions:
166  - identifier: 'PubChem: 23954' #試料の化学組成に関するID
167    description: TiCoooo #試料の化学組成の説明・記述
168
169  crystallographic_structures:
170  - category_vocabulary: https://matvoc.nims.go.jp/wiki/Item:Q556 #試料の結晶構造に関するID
171    description: （試料の結晶構造の説明・記述）
```

メタデータのバリデーション

- Pythonで書かれたYAMLバリデーターの
  Yamale (https://github.com/23andMe/Yamale) を使って
  メタデータのバリデーションが可能

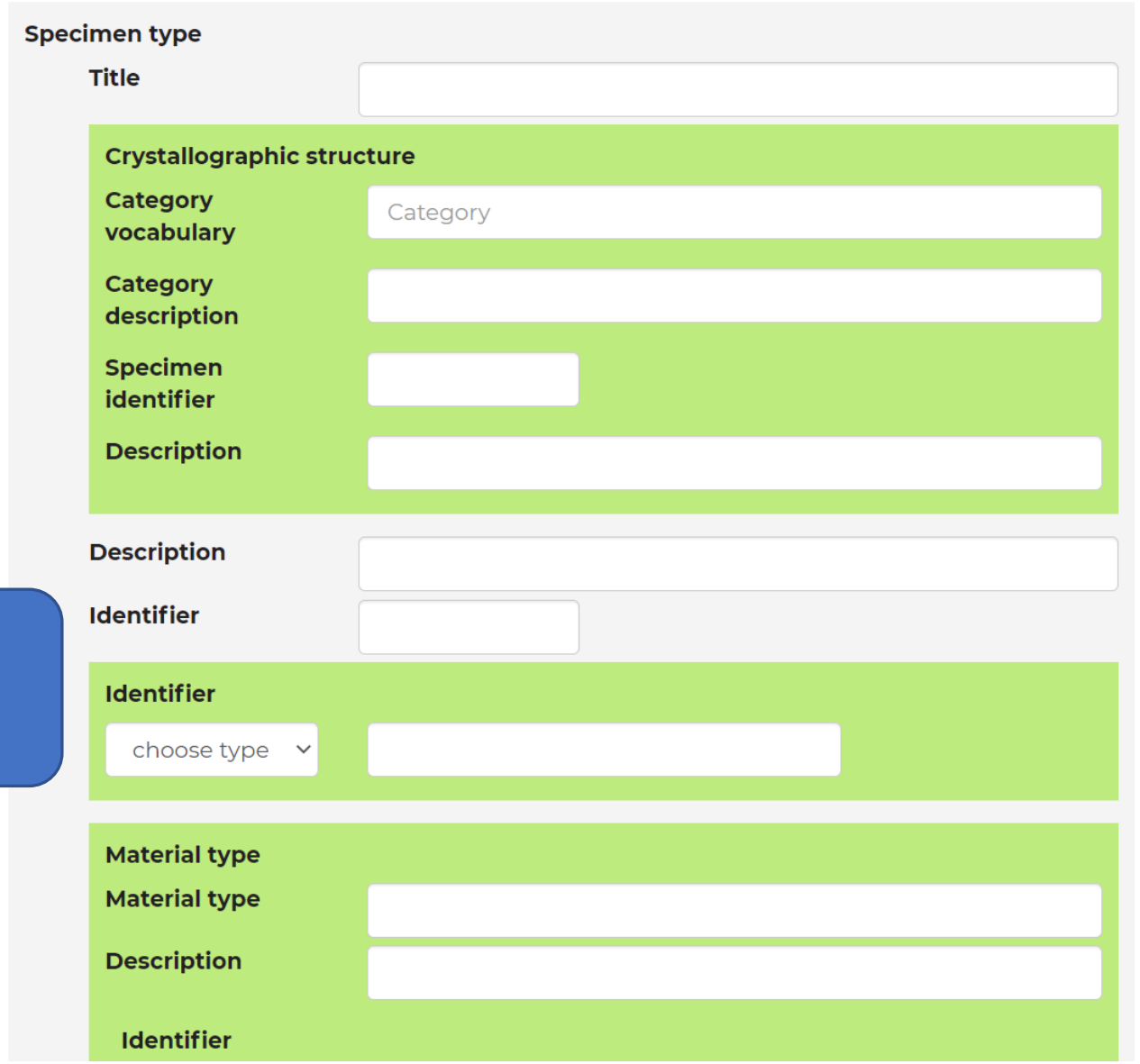
```
$ yamale metadata-sample.yaml
Validating /home/kosuke/mdr-schema/metadata-sample.yaml...
Validation failed!
Error validating data '/home/kosuke/mdr-schema/metadata-sample.yaml'
        titles.0.title: Required field missing
$
```

titleがないという
エラーになっている

なぜYAMLで記述？

- メタデータのWebフォームへの手入力はとても手間がかかる
- 機械可読な形でメタデータを書き、それを一括でインポートしたい

MDRのメタデータ入力画面



機械可読フォーマットの比較

- **CSV・Excel**
  - 表形式では、**1対多の関係を1枚のシートで記入するのが難しい**
    - データとその著者・ORCID番号の関係など
- **JSON**
  - **メタデータのファイル内にコメント文が書けない**ため、他人に渡すテンプレートファイルとして使えない
    - 「ここだけ埋めて」と言って書かせるのが難しい
- **YAML**
  - **ファイル内にコメント文が書けるので、テンプレートとして使える**
    - **「ここだけ埋めて」と言って書かせることがぎりぎり可能**

なぜ2階層に限定？

- 当初深い階層を持つメタデータスキーマを定義したが、メタデータを作る研究者も、それを扱うシステム開発者も理解や実装が難しかった
- そもそも各メタデータ項目にIDがついていれば、階層構造をとる必要性がほとんどなかった

階層構造の深いメタデータの例: 著者の情報

```
creators:
  name:
  - full_name: Kosuke Tanabe
    lang: en
  - full_name: 田辺浩介
    lang: ja
  identifiers:
  - type: orcid
    identifier: 0000-0002-9986-7223
  - type: e_rad
    identifier: 70409788
```

日本語のfull_nameの値を記述するのに
creators -> name -> full_name の2階層の記述が必要

orcidの値を記述するのに
creators -> identifiers -> identifier と
creators -> identifiers -> type の両方の記述が必要

階層構造を制限した記述の例:著者の情報

creators:

- name: Kosuke Tanabe

  orcid: 0000-0002-9986-7223

  e_rad: 70409788

full_name -> en, full_name -> ja
相当の情報は、IDの参照先のデータベース（この場合はORCID）に記述しておく

ORCID APIのレスポンスから
日本語の氏名を取得できる

```
<common:last-modified-date>2014-11-05T15:22:16.906Z</common:last-modified-date>
            <common:source>
                <common:source-orcid>

<common:uri>https://orcid.org/0000-0002-9986-7223</common:uri>
                    <common:path>0000-0002-9986-7223</common:path>
                    <common:host>orcid.org</common:host>
                </common:source-orcid>
                <common:source-name>Kosuke Tanabe</common:source-name>
            </common:source>
            <other-name:content>田辺 浩介</other-name:content>
        </other-name:other-name>
    </other-name:other-names>
    <person:biography visibility="public" path="/0000-0002-9986-7223/biography">
        <common:created-date>2016-04-15T23:26:38.757Z</common:created-date>
```

階層構造の深いメタデータの例: 装置の情報

```
instruments:
- name: BL14B2_XAFS
  description: SPring-8
  function:
  - category: spectroscopy
  managing_organization:
    organization: JASRI
```

category（装置の機能のカテゴリー）の値を取得するのに
instruments -> function -> category と
3階層たどる必要がある

managing_organization（装置管理者）の値を取得するのに
instruments -> managing_organization -> organization と3階層たどる必要がある

階層構造を制限した記述の例: 装置の情報

```
instruments:
- identifier: instrument_00001
  name: BL14B2_XAFS
  description: SPring-8
  function_category:
  - https://matvoc.nims.go.jp/entry/Q30
instrument_managing_organization:
  instrument_identifier: instrument_00001
  organization: JASRI
```

装置のローカルID。これを用いて instrument_managing_organizationの紐付けを行う

“spectroscopy”を示すID

instrument_managing_organization（装置管理者）をinstruments以下の項目からトップレベルの項目に移動

MDR Schema導入の効果

- メタデータの記述が簡潔になった
  - 研究者とリポジトリ担当者との間での、メタデータの修正依頼とその確認作業のやりとりが大幅に減った
  - **連携先のデータベース（DOI, ORCIDなど）に存在するメタデータ項目**を参照することで、リポジトリ自身で読み書きを行うメタデータ項目が減った
  - これらの結果、**リポジトリへのデータ登録にかかる時間が大きく短縮**された

MDR Schemaの作成で得た教訓

- メタデータ項目の決定や入力を最初からがんばりすぎない
  - リポジトリを作る人も、メタデータを使う人もたいへん
  - メタデータが書けないからデータの登録が滞るのでは本末転倒
- メタデータ項目に迷う前に、扱う対象にとにかくIDをつけよう
  - DOI, ORCID, RORが使えればそれを使う
  - 装置にもDOIをつける取り組みが始まっている
    - 「研究資料・実験機器へのPID付与検討小委員会」の取り組みを参照
  - それらが使えない場合、UUIDでもローカルの通し番号でもよい
    - ただ、ローカルでのID管理は労力が大きいので、できる限りDOIやORCIDを使おう