The University of Manchester
Jodrell Bank
Observatory

VO-DML Tooling Update

Paul Harrison (JBO)
IVOA Interop Autumn 2022

Contents

- VO-DML Tooling Updates
- Serialization
- Python generation

Gradle based VO-DML Tooling

- Introduced the new effort to update the VO-DML tooling at the last two Interops
  - Focus on text-represented, model-first development with code generation
    - improve the tooling setup (see PhotDM_setup example fork)
    - generated code can then implement serialization.
  - Make collaborative development of DMs less painful.
    - easy “rigorous” modular reuse of existing models.
    - the generated code is another way to judge “quality” of data model.
- Gradle-based tooling now ‘standard’ (plug-in at version 0.3.10)
  - https://github.com/ivoa/vo-dml/blob/master/tools/ReadMe.md
  - Additional functionality and improvements have been driven by use on ProposalDM

Serialization

- "natural" (hierarchical objects) serialization vs. mapping to VOTable.
  - compact and easy to read - vs dealing with the "meta model"
  - different use cases (e.g. service api, config file)
    - should be possible to translate automatically between the MIVOT and this serialization
  - Generated code achieves round-trip serialization (naturally!)
- does UType = VODML-REF (or not)?
  - still not sure the there is a rigorous definition of UType anywhere

XML serialization

- Whole model instance serialization - all references are included first
- This is different from previous VO-DML XML serializations.
- Would be good for VO-DML to have distinction between internal and external references.

```
<MyModel>
    <refs>
        <aref id="1000"> ... </aref>
        <bref name="bid"> ... </bref>
    </refs>
    <contentObjectType1>
        <bref>bid</bref>
        ...
    </contentObjectType1>
    <contentObjectType1>
        <aref>1000</aref>
        ...
    </contentObjectType1>
    ...
</MyModel>
```

JSON Serialisation (new)

```
{
  "MyModel": {
    "refs": {
      "mymodel:package.refa" : [
        {"name": "a1", "val" : "aval"},
        {"name": "a2", "val" : "aval2"}
      ],
      "mymodel:package.refb" : [
        {"_id" : 1000, "val" : "aval"},
        {"_id" : 1001, "val" : "aval2"}
      ]
    },
    "content" : [ {
      "mymodel:package.content1" : {"zval" : "aval", "refa" : "a1"}
    },
      {
      "mymodel:package.another1" : {
        "nval" : "aval", "refb" : 1001,
        "enc": {"foo": 23, "bar": "value"}
      }
     }
    ]
  }
}
```

referenced first

_id is conventional name if no natural key

uType used as type identifier

Reference to a "natural" key

Anonymous object if the type can be inferred unambiguously from model

RDB serialization

- Use object ⇒ relational facilities provided by Java JPA tools
  - Implemented using Hibernate
- Main design decisions
  - Using the “Joined Table” default methodology for inheritance - SingleTable as new optional mapping
    - DataTypes become embedded within parent table as extra rows
    - “NOT NULL” constraints difficult to be comprehensive with (especially for the embeddable and SingleTable cases)
    - There are some “edge cases” still to be determined
      - what to do about arrays? - solution will have to be RDB specific
- Details of this are not yet documented anywhere except by the generated code and DDL
  - However, round trips with instances are being done frequently with real models - e.g. ProposalDM

Python generation (new)

- Have put in place the “scaffolding” to complete the task of proper Python code generation
  - vodmlPythonGenerate gradle task
  - vo-dml2python.xsl
    - does basic @dataclass generation
      - need to add XML,JSON & RDB support
- ⇒ Python code generation not yet “production ready”
  - when the full serialization interoperability with Java code achieved then the tooling will be deemed to have reached v1.0 status.

TODO

- Finish Python code generation (volunteers?)
- Add “MIVOT VOTable mapping” serialisation code.
- Formal changes to the VO-DML standard and schema (v1.1)
  - Making optional some of the repeated information in VO-DML
    - the “Natural Keys” extension..
- Would be good to have an updated DM Designers’ Cookbook.
- C++ code generation?
- TODOs actually managed as usual with GitHub issues