The University of Manchester
Jodrell Bank
Observatory

MANCHESTER
1824

VO-DML Tooling Demo

Paul Harrison (JBO)
IVOA Interop Spring 2023

Introduction

- VO-DML Tooling update introduced in previous Interop talks using gradle (v7) now quite mature.
    - refined by the needs of ProposalDM
        - automated model code generation for interfaces to Proposal Submission Tool
    - code generation based on VO-URP by Gerard Lemson & Laurent Bourgès - but extended significantly with new serialization strategies.

Demo

- Integrating CoordsDM - see GitHub fork
  - `git clone -b vodmlplug  https://github.com/pahjbo/CoordinateDM.git`
  - Don’t need to explicitly download tools
    - set up appropriate control files in model repository
      - build.gradle.kts (the most important)
      - settings.gradle.kts (just says where to find things)
  - Integrates nicely with IDEs

Demo - Familiar Operations

- Things you know still work
  - validate model
    - `gradle vodmlValidate`
    - success!
  - generate documentation
    - `gradle vodmlDoc`
    - results in build/generated/docs/vodml/
  - derive VO-DML from UML XMI
    - `gradle UmlToVodml`
    - after setting up appropriate task

The University of Manchester
Jodrell Bank
Observatory

MANCHESTER
1824

OPTICON
RadioNet
Pilot

Demo - Testing

- Go beyond simple validity of the model - evaluate the model by looking at instances written in a statically typed language (Java).
- Model code auto-generated from VO-DML
    - allows testing in place with model repository
    - `git checkout testgenerated`
        - example instance test code already hand-written in this branch!
    - `gradle test`
    - model serialisation tests - round trips - everything passes

eMERLIN

Demo - Rapid Model Evolution

- Make the model instance invalid?
  - Comment out the ***lat*** of the **LonLatPoint**
  - - it still validates, why?

- Make rapid changes to the model in VODSL (convert the VO-DML to VODSL)

  - `gradle vodmltovodsl --dml=vo-dml/Coords-v1.0.vo-dml.xml --dsl=model/Coords-v1.0.vodsl`
  - NB - need to change vodsl base model include to
    - include "../build/tmp/IVOA-v1.0.vodsl"
  - cheat - the branch already has the suitably edited vodsl file…

- Make ***lat*** a mandatory attribute of **LonLatPoint** to vodsl file then

  - `gradle test`

  - now it fails 😉

The University of Manchester
Jodrell Bank
Observatory

Summary

- The “vodmlplug” branch (link shows files added) merely replaces the old ant based tools - but you do not have to download tooling directly.
  - makes no changes to VO-DML
  - adds
    - some better schematron checking of the VO-DML
    - export of “editable” GML diagram for publishing
- The “testgenerated” branch shows how to evolve the model (or create a new one) using VODSL to create the VO-DML
  - VODSL is more concise than XMI or even VO-DML, therefore easier to see diffs in version control systems, therefore easier to collaborate.
    - VO-DML->VODSL ->VO-DML round-trip works
  - Java code generation and model serialization round trips to XML, JSON and RDB working smoothly for general models
    - Note XML schema different from the “ant” era - different strategy for references.
  - There is a lot happening automatically - perhaps the demo does not illustrate this well!

Future Steps

- Finish Python code generation (volunteers?)
- Add “MIVOT VOTable mapping” serialisation code.
- Formal changes to the VO-DML standard and schema (v1.1)
  - Make appendix C normative
  - Making optional some of the repeated information in VO-DML
  - the “Natural Keys” extension..
  - clear up constraints usage.
- Would be good to have an updated DM Designers’ Cookbook.
- These and more managed as GitHub Issues