Event Report

CDM CODE GENERATORS WORKSHOP

CHRIS.RAYNER@ISLAEMEA.ORG

1 Introduction

This document is a detailed overview of the CDM Code Generator Workshop organised and sponsored by The International Securities Lending Association (ISLA).

2 Overview

Over two days at the end of January/beginning of February 2024, the International Securities Lending Association (ISLA) in conjunction with REGnosys and the Fintech Open Source Foundation (FINOS) ran a workshop focused on the Common Domain Model (CDM) Code Generators.

The event was hosted by etc.venues at their Monument site on January 31st and February 1st. Organisation, logistics and comms were provided by Chris Rayner and Harry Kingham from ISLA, with subject matter expertise being provided by Minesh Patel and Hugo Hills from REGnosys. Rob Moffat attended the retrospective session on behalf of FINOS.

3 Participants

The workshop was opened to a select group who had expressed an interest in getting a better understanding of how the CDM Code Generators worked. Having a smaller group allowed a more hands-on approach to the workshop and promoted a collaborative team working environment.

Representatives from the following organisations attended:

Pirum were also due to attend but sent apologies as they unfortunately had to pull out.

Firms who were not able to attend the workshop as participants were offered the opportunity to drop in and observe. Representatives from Broadridge, FINOS, FIS, ISLA, London Stock Exchange Group and REGnosys dropped in over the course of the two days.

4 The Challenge

Participants were set the task of creating a new Code Generator that could translate the CDM Rosetta format into valid JSON Schema. The resultant code and associated documentation would then be contributed back to the CDM github repository for use by the community.

The participants decided to work to draft 4 of the JSON Schema specification as defined by json-schema.org. The main driver for this was that there is a real-world use case for a JSON Schema representation of the CDM that conforms to the version of JSON Schema used by OpenAPI 3.0.3[1].

The artefacts required to meet the challenge were then defined. Discussions highlighted three important elements that needed to be scoped:

1. **Source Data – Rosetta Samples**
   The first requirement was to manually create small Rosetta samples that would demonstrate the most common constructs that are used in the CDM. These samples needed to provide examples of:

   - Basic Rosetta types i.e. string, number, integer, date, time, dateTime, Boolean
   - Enumerations
   - Complex Rosetta types including nested and cyclical calls
   - Various cardinalities including optional (0..1), mandatory(1..1) and multiple (n..*), (n..n)
   - References using metadata id, key, reference, location and address

   The added benefit of creating these simple source examples is that they can be reused for existing and future code generation projects.

   The samples were created in separate files which better replicates the structure of the CDM. This allowed the concept of namespaces to be considered and tested as part of the deliverable.

2. **Test Data – JSON Samples**
   JSON samples that conform to the structures defined in the example Rosetta files were also required. These could then be used in test cases to confirm that the JSON schema generated exactly matches the corresponding Rosetta definitions.

   Like the Rosetta samples, these JSON samples were handwritten and peer reviewed by the workshop participants.

[1] Note: OpenAPI 3.1 conforms to the latest JSON Schema specification, 2020-12. However, this is only partially backwardly compatible with draft 4. As part of the workshop the JSON Schema that was generated was tested against both the draft 4 and 2020-12 specifications, and was successfully validated against both.

3. Prototyping Code Generators

CDM Code Generators require the use of Xtend which is an extension to the Eclipse IDE. The actual coding required to create a new code generator was core to the workshop and relied upon the source and test data samples for acceptance testing.

To facilitate the testing of the JSON schema created by the new code generator new functions were proposed. These had to be able to:

- Confirm that the JSON Schema generated conforms to the draft 4 specification
- Confirm that the JSON samples validate against the generated JSON Schema

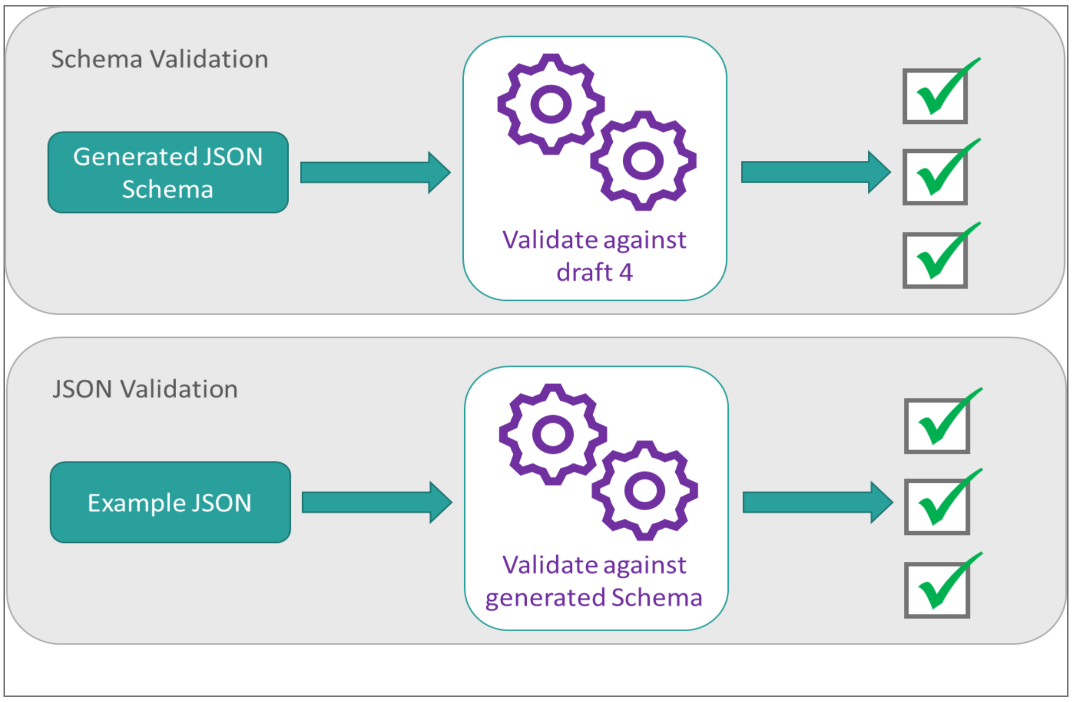
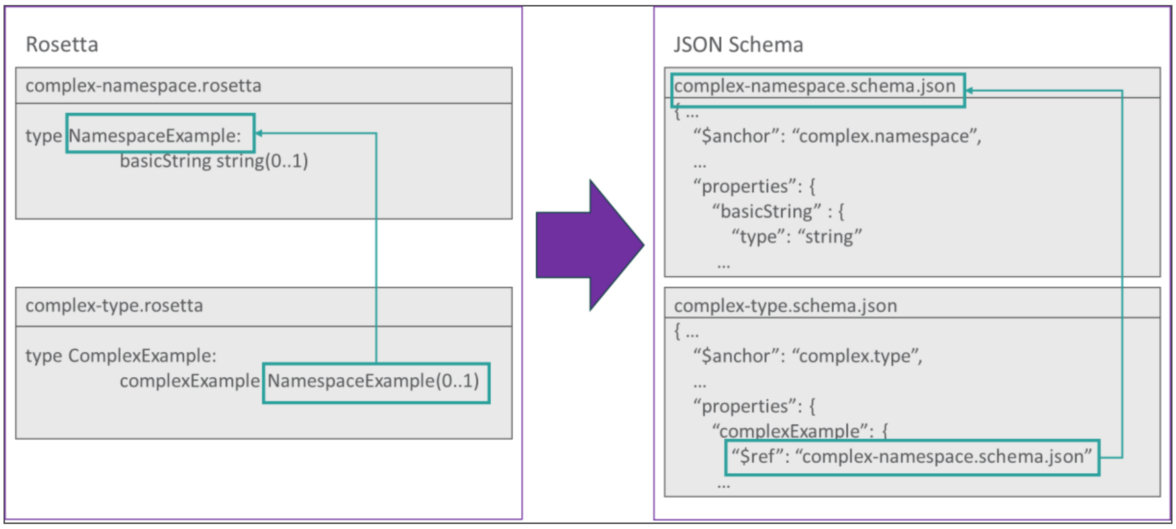


Diagram 1: Two functions are required to confirm that the generated JSON Schema is as expected.

The decision was made to write the code generator so that it would create multiple JSON Schema files, one for each Rosetta file processed, rather than just one schema. This is more consistent as it replicates the core structure of the CDM.

Writing the code generator to generate JSON Schema files for each Rosetta file meant that the schema files had to have a way to reference each other. In order to support the references to dependent or associated types and namespaces the JSON Schema files used the “$anchor” and “$ref” keywords.

The namespace for a schema file was taken as the Rosetta file name and put into a “$anchor” type; when a different namespace needed to be referenced from within a type then the JSON Schema file holding that type would be put into a “$ref” type.



complex-namespace.rosetta

```
type NamespaceExample:
        basicString string(0..1)
```

complex-type.rosetta

```
type ComplexExample:
        complexExample NamespaceExample(0..1)
```

complex-namespace.schema.json

```
{ ...
  "$anchor": "complex.namespace",
  ...
  "properties": {
    "basicString" : {
      "type": "string"
    ...
```

complex-type.schema.json

```
{ ...
  "$anchor": "complex.type",
  ...
  "properties": {
    "complexExample": {
      "$ref": "complex-namespace.schema.json"
    ...
```

Diagram 2: In the Rosetta examples above "complexType" is of type "NamespaceExample" which is defined in a different namespace "complex-namespace". The JSON Schema generated for these Rosetta examples uses a "$ref" for "complexType" that points to the schema file that holds the structure of "NamespaceExample", being "complex-namespace.schema.json".

In order to maximise the time and skillsets available for the workshop the participants split into two groups – one group worked on the Rosetta, JSON and schema examples, while the other worked on the code required for the new code generator and test functions.

5 Achievements

A new code generator was successfully created that can translate constructs in Rosetta format into JSON Schema that conform to the required schema version, being the draft 4 specification.

To support and validate the code generator the following data and files were created:

- 7 Rosetta files, containing 25+ constructs including basic types and complex types with varying cardinalities, enumerations, metadata and references.
- 30+ JSON examples that correspond to the Rosetta constructs in the Rosetta files
- Expected JSON Schemas for each of the Rosetta and JSON examples
- 100+ test cases to confirm the validity of the JSON Schema
- A java function that takes a Rosetta sample and invokes the new code generator to create the JSON Schema from it and then confirm that the schema created is valid against the JSON Schema draft 4 specification
- A java function that takes a JSON sample file and confirms that it is valid against the JSON Schema that was generated by the code generator

With over 100 test cases defined and successfully executed, the group were confident that the JSON Schema being generated was both valid against the JSON Schema draft 4 specification, as well as being an accurate representation of the Rosetta constructs.

As a final test, the code generator was run against the entire suite of Rosetta files in the current CDM production version, being 5.4.0. This processed 756 Rosetta files and successfully created 756 valid JSON Schema files in just under 5 seconds.

All sample data, test cases and code will be contributed to the core CDM repository. The aim is for this to be completed by the end of 2024 Q1.

6 Appendix

For more information about the CDM please go to the Common Domain Model microsite hosted by FINOS here: https://cdm.finos.org

For more information about the work being done by ISLA please go to the main International Securities Lending Association website here: https://www.islaemea.org/