LinkML in Reactome

Progress so far

Robert Petryszak
11 Jan 2024

Why LinkML?

- Declarative representation of Reactome schema in a single place
- Automatic generation of:
  - model java classes
  - GraphQL schema from → GraphQL API Dev

  from LinkML
- More transparent schema change protocol:
  - Automated via Jenkins:
    - LinkML syntax validation after changes to the schema
    - Generation of model java model classes and PR creation for graph-core(-curator) changes

reactome / **reactome-schemas**

- Schema.yaml
  - production
- Schema.web.diff.yaml
  - difference between production and web

Slots:

```yaml
name:
  multivalued: true
  annotations:
    constraint: MANDATORY
    addedField: false
```

Slots can be used in many classes; can be specialised within a class.

```yaml
#### Affiliation ####
Affiliation:
  is_a: DatabaseObject
  annotations:
    node: true
    suppress_warnings: "unused"
  slots:
    - name
  slot_usage:
    name:
      annotations:
        category: all
  attributes:
    address:
      annotations:
        constraint: REQUIRED
        addedField: false
```

reactome / **reactome-schemas**

- Certain content of Reactome model java classes could not be captured in LinkML, e.g.

reactome-schemas / additional_class_content / curator-graph-core-classes / **LiteratureReference.java**

```
public String getUrl() {
    if (pubMedIdentifier != null) {
        return PUBMED_URL + pubMedIdentifier;
    }
    return null;
}
```

reactome-schemas / additional_class_content / curator-graph-core-classes / **DatabaseObject.java**

```
@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;

    DatabaseObject that = (DatabaseObject) o;
    return DB_ID != null ? DB_ID.equals(that.DB_ID) : that.DB_ID == null &&
           Objects.equals(_displayName, that._displayName);
}
```

reactome / **reactome-schemas**

- More complex java methods are generated via templates populated from LinkML

reactome-schemas / code_templates / RelationshipFetch.java

```
@JsonIgnore
public List<StoichiometryObject> fetch@Attribute@() {
    List<StoichiometryObject> objects = new ArrayList<>();
    if(@attribute@!=null) {
        for (@RelationshipClass@ aux : @attribute@) {
            objects.add(new StoichiometryObject(aux.getStoichiometry(), aux.getPhysicalEntity()));
        }
        Collections.sort(objects);
    }
    return objects;
}
```

Variables in template code replaced by values from LinkML during java class generation

reactome-schemas / code_templates / SetWithAllowedClasses.java

```
public void set@Attribute@(@Clazz@ @attribute@) {
    if(@attribute@ == null) return;

    if (@allowed_test_clauses@) {
        this.@attribute@ = @attribute@;
    } else {
        throw new RuntimeException(@attribute@ + @error_msg@);
    }
}
```

reactome / **reactome-schemas**

reactome-schemas / schema.web.diff.yaml

```
#### CellDevelopmentStep ####
CellDevelopmentStep:
  is_a: ReactionLikeEvent
  annotations:
    removed_slots: "catalystActivity"
    removed_attributes: "requiredInputComponent"
```

ReactionLikeEvent overrides ReactionlikeEvent

Diff file contains what's overridden, added to or removed from schema.yaml

schema.yaml:

```
#### CellDevelopmentStep ####
CellDevelopmentStep:
  is_a: ReactionlikeEvent
  annotations:
    node: true
  slots:
    - catalystActivity
    - tissue
  attributes:
    requiredInputComponent:
      multivalued: true
      range: PhysicalEntity
      annotations:
        constraint: REQUIRED
        type: "requiredInputComponent"
        set: true
```

reactome / **reactome-schemas**

reactome-schemas / **schema.web.diff.yaml**

```
#### CellDevelopmentStep ####
CellDevelopmentStep:
  is_a: ReactionLikeEvent
  annotations:
    removed_slots: "catalystActivity"
    removed_attributes: "requiredInputComponent"
```

ReactionLikeEvent overrides ReactionlikeEvent

Diff file contains what's overridden, added to or removed from schema.yaml

schema.yaml:

```
#### CellDevelopmentStep ####
CellDevelopmentStep:
  is_a: ReactionlikeEvent
  annotations:
    node: true
  slots:
    - catalystActivity
    - tissue
  attributes:
    requiredInputComponent:
      multivalued: true
      range: PhysicalEntity
      annotations:
        constraint: REQUIRED
        type: "requiredInputComponent"
        set: true
```

Note: any changes to (production) schema.yaml will end up in generated model java classes for **both production and web** unless specifically turned off in schema.web.diff.yaml

reactome-schemas / help.py

The following functionality is available:

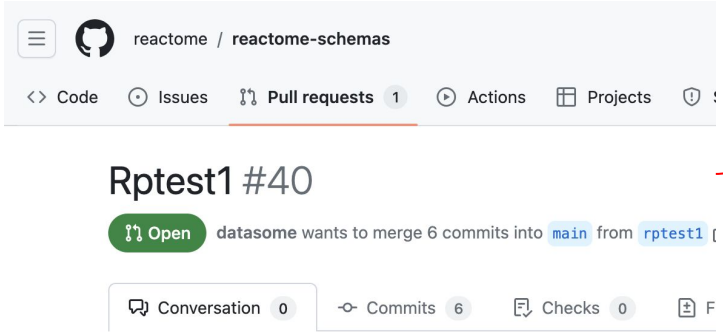
- **./generate.py java**
  Generate graph-core-curator java classes into curator-graph-core-classes directory
- **./generate.py java schema.web.diff.yaml**
  Generate graph-core java classes into graph-core-classes directory
- **./generate.py graphql**
  Generate curator-graph-core-classes/schema.graphql from schema.yaml
- **./generate.py graphql schema.web.diff.yaml**
  Generate graph-core-classes/schema.web.graphql from schema.yaml merged with schema.web.diff.yaml
- **./generate.py yaml schema.web.diff.yaml**
  Generate schema.web.yaml by merging schema.yaml and schema.web.diff.yaml
  - so that schema.web.yaml's syntax can be validated
- **gen-json-schema schema.yaml**
  Generate json schema from schema.yaml - can be used to validate linkml syntax of schema.yaml
- **gen-json-schema schema.web.yaml**
  Generate json schema from schema.web.yaml - can be used to validate linkml syntax of schema.web.yaml
- **./compare_java.py curator-graph-core-classes <graph-core-curator clone dir>/src/main/java/org/reactome/server/graph/curator/**
  Compare generated classes to those in graph-core-curator repo
- **./compare_java.py graph-core-classes <graph-core clone dir>/src/main/java/org/reactome/server/graph/curator/**
  Compare generated classes to those in graph-core repo

reactome-schemas / compare_java.py

- Less strict than unix/git diff
  - Proves that all model data in java classes was captured in LinkML
  - Highlights non-standard variable naming or method implementations
    - Work on ‘regularizing’ graph-core(-curator) completed (in a branch)
  - When LinkML schema is changed, git diff can be used to compare the newly generated model java classes with the original ones

Jenkins

PR created



hook

https://release.reactome.org/jenkins/ghprbhook/

pipeline

https://release.reactome.org/jenkins/job/continuous-integration/job/reactome-schemas-pull-request

function

reactome-schemas / Jenkins-reactome-schemas-pull-request.sh

1. Validate syntax of schema.yaml and schema.web.yaml
2. On validation failure, email:
   - **Now**: Robert
   - **Future**: **?**

Jenkins

reactome / **reactome-schemas**

- Merged PR/git push

hook

http://release.reactome.org/jenkins/github-webhook/

pipeline

https://release.reactome.org/jenkins/job/continuous-integration/job/reactome-schemas

function

reactome-schemas / Jenkins-graph-core

1. Git clone graph-core(-curator)
2. Generate java classes
3. **Now**: compare_java.py: 1 to 2
   Future:
   - git diff: 1 to 2 > diff.txt
   - If diff.txt not empty, create PR for graph-core(-curator) with diff.txt in the description
   - Email PR approvers?