SNOMED CT Expression Constraint Language

The SNOMED CT Expression Constraint Language is a formal language for defining bounded sets of clinical meanings represented by either precoordinated or postcoordinated expressions.

Status

SNOMED CT expression constraint language version 1.3 has been published at http://snomed.org/ecl.

Implementations

The SLPG is aware of the following implementations of the SNOMED CT Expression Constraint Language:

- SNQuery - http://snquery.veratech.es/
- Slang - http://slang.snomedic.com:8080/yats/
  - Provides API for the SNOMED International public browser (See "Expression Constraint Queries" tab) - https://browser.ihtsdotools.org/
- SNOMED International SNOMED CT Query Service - https://github.com/IHTSDO/snomed-query-service
- Snow Owl Terminology Server (via REST API) - https://github.com/IHTSDO/snow-owl
- Snow Owl MQ (form-based interface) - https://mq.b2i.sg/snow-owl/
- PHAST's FHIR Terminology server STS - https://topaze.phast.fr/resources-server/api/fhir
  - Available 7/24 for testing, through CSIRO's VS tool - https://ontoserver.csiro.au/vstool/

If you know of additional ECL implementations, please contact eps@snomed.org with details.

Future Versions

Additional requirements are being collected for consideration in a future version of the SNOMED CT expression constraint language.

In the following sections, we describe some of the proposed requirements, which are under consideration:

Expression Constraint Implementation and Testing Artifacts

A number of artifacts have been proposed to assist with the implementation and testing of the Expression Constraint Language, including:

- An ANTLR grammar of the ECL syntaxes
- A set of invalid expression constraints that should fail in testing an expression constraint parser
- A set of valid results for each example expression constraint, executed against a specific version of the SNOMED international edition
- Track changes, change bars or similar to make it easier for implementers to identify the areas of change. In addition, changes should be localized and minimized.

Selecting Concepts Recorded in Additional Reference Set Attributes

We are going to develop the refsets for anatomy. The plan is to have two association refsets for Structure/Entire and Structure/Part. Please see more information at SEP maps and refset.

We need some way of selecting the 'targetComponentId' column of these reference sets. This column is represented by the refset attribute concept 900000000000533001 [Association target component].

This may be needed, for example, to define a template slot rule that restricts possible values of the slot to only 'entire' concepts, or 'part concepts'.

Possible syntaxes:

- ^[1000005]Anatomy structure and entire association reference set$;900000000000533001 [Association target component]
- ^[1000005]Anatomy structure and entire association reference set$;900000000000533001 [Association target component]
- ^1000005[Anatomy structure and entire association reference set];900000000000533001 [Association target component]

Using Any for any Concrete Value
The Any wildcard (i.e. `*`) should be able to be used to represent any concrete value (i.e. number or string), rather than just for any concept. In doing so, consideration must be given to ensure that the grammar is able to be parsed unambiguously.

Matching Attribute Names

How can I write an ECL expression to match attribute names? Currently ECL expressions can match (return) concepts that are either the source or the target of a relationship triple (target is accessed via the ‘reverse’ notation or ‘dot notation’, but not the relationship type (i.e attribute name) itself. For example, I can write:

```
<< 404684003|Clinical finding| : 363698007|Finding site| = <<66019005|Limb structure|
```

and

```
<< 404684003|Clinical finding| . 363698007|Finding site|
```

But how do I get all the attribute names that are used by `<< 404684003|Clinical finding|`?