2019-10-29 - SLPG Meeting

Date & Time

10:30am (local time) Tuesday 29th October 2019

Location

Grand Residence 103, Grand Hyatt Hotel Kuala Lumpur, Malaysia

Goals

- To review current priorities and plan future work based on these priorities
- To progress work on
  - URIs
  - Templates
  - ECL
  - Query language

Attendees

- Chair: Linda Bird
- Project Group: Michael Lawley, Daniel Karlsson, Anne Randorff Højen, Peter G. Williams, Kai Kewley
- Observer: Fuad Rahman (Goldblatt)

Apologies

- Ed Cheetham

Agenda and Meeting Notes

Description	Owner	Notes
Welcome and agenda	Linda Bird	
Priorities	Linda Bird	• Discuss current priorities and new requirements ○ Add 'boolean' to Compositional Grammar, ECL and Templates ○ Use abbreviation "bool".
URIs	Peter G. Williams	• Overview of current status: ○ Published version: https://confluence.ihtsdotools.org/display/DOCURI ○ Work in progress: https://confluence.ihtsdotools.org/display/WIPURI/URI+Standard ■ Consistent structure to each URI page ■ New proposal on URIs for languages and language instances • Proposed update to support FHIR Resource instances ○ 2.7 URIs for SNOMED Resources • Agreement ○ http://snomed.info/valueSet/gps ○ http://snomed.info/fhirStructureDefinition/conditionWithSnomedBinding http://snomed.info/openEhrArchetype/conditionWithSnomedBinding http://snomed.info/<resourcetype>/<resourcename> ResourceTypes: ■ valueSet ■ fhirStructureDefinition ■ openEhrArchetype ○

Expression Templates	Peter G. Williams	• Overview of current status ◦ Published version: https://confluence.ihtsdotools.org/display/DOCSTS/Template+Syntax+Specification ◦ Work in progress: https://confluence.ihtsdotools.org/display/WIPSTS/Template+Syntax+Specification ▪ Added a 'default' constraint to each replacement slot - e.g. default (72673000 |Bone structure (body structure)|) ▪ Enabling 'slot references' to be used within the value constraint of a replacement slot - e.g. [[+id (<< 123037004 |Body structure| MINUS << $findingSite2) @findingSite1]] ▪ Allowing repeating role groups to be referenced using an array - e.g. $rolegroup[1] or $rolegroup[!=SELF] ▪ Adding 'sameValue' and 'allOrNone' constraints to information slots - e.g. sameValue ($site), allOrNone ($occurrence) ▪ Self ▪ See changes in red here: 5.1. Normative Specification Examples: [[+id]]: [[1..*] @my_group sameValue(morphology)] { |Finding site| = [[+id (<<123037004 |Body structure (body structure)| MINUS << $site[! SELF]) @site]] , |Associated morphology| = [[+id @my_morphology]] } • Implementation feedback on draft updates to Expression Template Language syntax ◦ Use cases from the Quality Improvement Project: ▪ Multiple instances of the same role group, with some attributes the same and others different. Eg same morphology, potentially different finding sites. Note that QI Project is coming from a radically different use case. Instead of *filling* template slots, we're looking at existing content and asking "exactly *how* does this concept fail to comply to this template?" For discussion: [[0..1]] { [[0..1]] 246075003 |Causative agent| = [[+id (< 410607006 |Organism|) @Organism]] } Is it correct to say either one of the cardinality blocks is redundant? What are the implications of 1..1 on either side? This is less obvious for the self grouped case. **Road Forward for SI** 1. Generate the parser from the ABNF and implement in the Template Service 2. User Interface to a) allow users to specify template at runtime b) tabular (auto-completion) lookup STL 3. Template Service to allow multiple templates to be specified for alignment check (aligns to none-off) 4. Output must clearly indicate exactly what feature of concept caused misalignment, and what condition was not met. Additional note: QI project is no longer working in subhierarchies. Every 'set' of concepts is selected via ECL. In fact most reports should now move to this way of working since a subhierarchy is the trivial case. For a given template, we additionally specify the "domain" to which it should be applied via ECL. This is much more specific than using the focus concept which is usually the PPP eg Disease. FYI Michael Chu
Description Templates	Kai Kewley	• Overview of current use • Review of General rules for generating descriptions ◦ Removing tags, words ◦ Conditional removal of words ◦ Automatic case significance ◦ Generating PTs from target PTs ◦ Reordering terms • Mechanism for sharing general rules - inheritance? include? • Description Templates for translation • Status of planned specification
Expression Constraint Language	Linda Bird	• Review scope and syntax of previous internal implementation (Kai Kewley) • Recap where we were up to with adding term-searches to ECL (see below) **Previous discussions** **Syntax** {{ term = [termSearchType :] "String", languageCode = [langCode] }} **Term Search Type**

a. Wild Card Match (collation) - e.g.
   - {{ term = **wild**:"*heart*" }}
   - {{ term = wild (sv):"*hjärta*" }}

a. ~~Regex - e.g.~~
   - ~~{{ term = regex:".*heart.*" }}~~

a. Word Prefix Any Order - e.g.
   - {{ term = **match**:"hear att" }}

a. Default (word prefix any order) - e.g.
   - 2. {{ term = "hear att" }}
   - {{ term = "*heart*" }}

Potential Examples

- << 64572001 |Disease| {{ term = "heart"}}
- << 64572001 |Disease| {{ term = "heart", languageCode = "en"}}
- << 64572001 |Disease| {{ term = "heart", languageCode = "en"}} AND << 64572001 |Disease| {{ term = "hjärta", languageCode = "sv"}}
- << 64572001 |Disease| {{ term = "heart", languageCode = "en"}} {{ term = "hjärta", languageCode = "sv"}}

- << 64572001 |Disease| {{ term = "heart", languageCode = "en"}} OR << 64572001 |Disease| {{ term = "hjärta", languageCode = "sv"}}
- << 64572001 |Disease| {{ (term = "heart", languageCode = "en") OR (term = "hjärta", languageCode = "sv")}}
- (<< 64572001 |Disease|: |Associated morphology| = *) {{ term = "heart", languageCode = "en", }} {{ term = "hjärta", languageCode = "sv"}}
- (<< 64572001 |Disease| {{ term = "*cardio*" }}) MINUS (<< 64572001 |Disease| {{ term != "*heart*" }})
- Recommendation to be made on (based on investigation of grammar):
  - << 64572001 |Disease| {{ term = "heart", languageCode = "en"}} AND {{ term = "hjärta", languageCode = "sv"}}
  - << 64572001 |Disease| ({{ term = "heart", languageCode = "en"}} OR {{ term = "hjärta", languageCode = "sv"}})
  - << 64572001 |Disease| ({{ term = "heart", languageCode = "en"}} MINUS {{ term = "hjärta", languageCode = "sv"}})

Use Cases

- Intentionally define a reference set for chronic disease. Starting point was ECL with modelling; This misses concepts modelled using the pattern you would expect. So important in building out that reference set.
- Authors quality assuring names of concepts
- Checking translations, retranslating. Queries for a concept that has one word in Swedish, another word in English
- AU use case would have at most 3 or 4 words in match
- Consistency of implementation in different terminology services
- Authoring use cases currently supported by description templates
- A set of the "*ectomy"s and "*itis"s

Questions

		○ Do we include 'typeId' - e.g. << 64572001 |Disease| {{ D.term = “*heart*”, typeId = 900000000000013009 |Synonym| }} ■ NO ○ Do we include 'type' - e.g. << 64572001 |Disease| {{ D.term = “*heart*”, D.type = synonym }} ■ NO ○ Do we include 'languageCode' - e.g. << 64572001 |Disease| {{ D.term = “*heart*”, D.type = synonym, D. languageCode = “en” }} ■ YES ○ Do we include 'caseSignificanceId' - e.g. << 64572001 |Disease| {{ D.term = “*Heart*”, D. caseSignificanceId = 900000000000017005 |case sensitive|}} ■ NO ○ Do we include 'caseSignificance' - e.g. << 64572001 |Disease| {{ D.term = “*Heart*”, D.caseSignificance = sensitive }} ■ NO ○ Do we include 'language' and 'version' - e.g. << 64572001 |Disease| {{ term = “*heart*” }} VERSION = http://…, LANGUAGE = (999001881000000108|Gastro LRS|, |GB English|) ■ NO ○ Do we include syntactic sugar - e.g. ■ << 64572001 |Disease| {{ preferredTerm = “*heart*”, languageRefSet = en-gb}} ■ << 64572001 |Disease| {{ fullySpecifiedTerm = “*heart*”, languageRefSet=en-gb}} ■ << 64572001 |Disease| {{ acceptableTerm = “*heart*”, languageRefSet = en-gb}} ■ << 64572001 |Disease| {{ preferredTerm = “*heart*”}} FROM version = X, language = Y ■ NO ○ Do we use/require the "D" at the start of "term"? ■ NO ○ Packaging - How do we package this extension to ECL ■ A new version of ECL - e.g. 2.0 (or 1.4?) same specification document ■ An optional extension to ECL for SNOMED authors/content developers - e.g. ECL++ / ECLv1.3++ An appendix on the ECL document? (with Filter Language) ■ A subset/profile of the Query Language A separate document that defines SNOMED filters, which can be added to any version of ECL ■ Filter Language
Executing maps	Linda Bird	Reverse memberOf function • What refsets is a concept a member of? Proposed syntax to support execution of maps (Outstanding question: ECL or Query Language? Scope and packaging needs further discussion) • Example use cases ○ Mapping from international substance concepts to AMT substance concepts ○ Anatomy structure and part association reference set - e.g. find the anatomical parts of a given structure • Potential syntax to consider ○ Functional ■ mapTarget (|Anatomy structure and part association refset|, << |Upper abdomen structure|) • Return the map targets from the given map refset. where the referencedComponent matches the condition ■ mapSource (|Anatomy structure and part association refset|, << |Liver part|) • Return the referencedComponent from the given map refset, where the targetId matches the condition. ○ Dot notation + Attribute refinement ■ |Anatomy structure and part association refset| . |mapTarget| ■ |Anatomy structure and part association refset| . |referencedComponent| (Same as ^ |Anatomy structure and part association refset|) ■ (|Anatomy structure and part association refset|: |referencedComponent| = << |Upper abdomen structure) . |mapTarget| ■ (|Anatomy structure and part association refset|: |mapTarget| = << |Upper abdomen structure) . |referencedComponent| ○ Dot notation + Filters ■ (|Anatomy structure and part association refset| {{ |referencedComponent| = << |Upper abdomen structure| }}). |mapTarget| ■ (|Anatomy structure and part association refset| {{ mapTarget = << |Upper abdomen structure| }}) . |referencedComponent| • ^ (|Anatomy structure and part association refset| {{ mapTarget = << |Upper abdomen structure| }}) ○ Specify value to be returned ■ ?|mapTarget|? |Anatomy structure and part association refset| ■ ?|mapTarget|? |Anatomy structure and part association refset| {{ |referencedComponent| = << |Upper abdomen structure| }} ■ ?|mapTarget|? |Anatomy structure and part association refset| : |referencedComponent| = << |Upper abdomen structure|

<table>
<tr><td>Returning attributes</td><td>Michael Lawley</td><td>Proposal from Michael:
• 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 (ie attribute name) itself.
For example, I can write:
<< 404684003|Clinical finding| : 363698007|Finding site| = <<66019005|Limb structure|
<< 404684003|Clinical finding| . 363698007|Finding site|
But I can't get all the attribute names that are used by << 404684003|Clinical finding|
○ Perhaps something like:
▪ ? R.type ? (<< 404684003 |Clinical finding|)
○ This could be extended to, for example, return different values - e.g.
▪ ? |Simple map refset|.|maptarget| ? (^|Simple map refset| AND < |Fracture|)</td></tr>
<tr><td>Query Language - Summary from previous meetings</td><td>Linda Bird</td><td>**Examples: version and language**
○ << 64572001 |Disease| {{ term = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131
○ << 64572001 |Disease| {{ synonym = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131
○ << 64572001 |Disease| {{ FSN = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131
○ << 64572001 |Disease| {{ FSN = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** W
○ << 64572001 |Disease| {{ preferredTerm = “*heart*”}} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** Y
○ << 64572001 |Disease| {{ acceptableTerm = “*heart*”}} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** Y
○ (* {{ term = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** Z) MINUS
(* {{ term = "*heart*" }} **VERSION** http://snomed.info/sct/900000000000207008/version/20170731, **LANGUAGE** W)
○ X MINUS Y **WHERE** X = * , Y = (* {{ term = "*heart*" }}) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** W
Notes
○ Allow nested where, version, language
○ Scope of variables is inner query</td></tr>
<tr><td></td><td></td><td>**Examples: where**
○ X MINUS >! X **WHERE** X = (<< 1234 : 5678 = << 6547)
○ X MINUS >! X **WHERE** X = (<< 1234 : 5678 = << 6547) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131
○ X MINUS >! Y **WHERE** X = (<< 1234 : 5678 = << 6547), Y = (<< 1456) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131
○ X MINUS >! X **WHERE** X = (<< 1234 : 5678 = << 6547) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131 , **LANGUAGE** 900000000000508004 |GB English|
○ X MINUS >! X **WHERE** X = (<< 1234 : 5678 = << 6547) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** 999001881000000108|GB clinical extension LRS|, 900000000000508004 |GB English|
○ X minus >! X **WHERE** X = (< M **WHERE** M = (< 1234))) **VERSION** http://snomed.info/sct/900000000000207008/version/20180131, **LANGUAGE** 999001881000000108|GB clinical extension LRS|, 900000000000508004 |GB English|
Notes
▪ Allow nested variable definitions, but recommend that people don't due to readability
▪ Scope of variables is the inner query
▪ No recursion e.g X **WHERE** X = 1234 MINUS X
• ie can't use a variable in its own definition
• ie X is only known on the left of the corresponding **WHERE**, and not on the right of the **WHERE**</td></tr>
</table>

Keywords for Term-based searching:

- **D.term**
  - D.term = "*heart*"
  - D.term = wild:"*heart*"
  - D.term = regex:".*heart.*"
  - D.term = match:"hear att"
  - D.term = (sv) wild: "*heart*"
- **D.languageCode**
  - D.languageCode = "en"
  - D.languageCode = "es"
- **D.caseSignificanceId**
  - D.caseSignificanceId = 900000000000448009 |entire term case insensitive|
  - D.caseSignificanceId = 900000000000017005 |entire term case sensitive|
  - D.caseSignificanceId = 900000000000020002 |only initial character case insensitive|
- **D.caseSignificance**
  - D.caseSignificance = "insensitive"
  - D.caseSignificance = "sensitive"
  - D.caseSignificance = "initialCharInsensitive"
- **D.typeId**
  - D.typeId = 900000000000003001 |fully specified name|
  - D.typeId = 900000000000013009 |synonym|
  - D.typeId = 900000000000550004 |definition|
- **D.type**
  - D.type = "FSN"
  - D.type = "fullySpecifiedName"
  - D.type = "synonym"
  - D.type = "textDefinition"
- **D.acceptabilityId**
  - D.acceptabilityId = 900000000000549004 |acceptable|
  - D.acceptabilityId = 900000000000548007 |preferred|
- **D.acceptability**
  - D.acceptability = "acceptable"
  - D.acceptability = "preferred"

Additional Syntactic Sugar

- **FSN**
  - FSN = "*heart"
    - D.term = "*heart", D.type = "FSN"
    - D.term = "*heart", D.typeId = 900000000000003001 |fully specified name|
  - FSN = "*heart" **LANGUAGE** X
    - D.term = "*heart", D.type = "FSN", D.acceptability = * **LANGUAGE** X
    - D.term = "*heart", D.typeId = 900000000000003001 |fully specified name|, acceptabilityId = * **LANGUAGE** X
- **synonym**
  - synonym = "*heart"
    - D.term = "*heart", D.type = "synonym"
    - D.term = "*heart", D.typeId = 900000000000013009 |synonym|
  - synonym = "*heart" **LANGUAGE** X
    - D.term = "*heart", D.type = "synonym", D.acceptability = * **LANGUAGE** X
    - D.term = "*heart", D.typeId = 900000000000013009 |synonym|, (D.acceptabilityId = 900000000000549004 |acceptable| OR D.acceptabilityId = 900000000000548007 |preferred|) **LANGUAGE** X
- **synonymOrFSN**
  - synonymOrFSN = "*heart"
    - synonym = "*heart" OR FSN = "*heart"
    - D.term = "*heart", (D.type = "synonym" OR D.type = "fullySpecifiedName")
  - synonymOrFSN = "*heart" **LANGUAGE** X
    - synonym = "*heart" OR FSN = "*heart" **LANGUAGE** X
    - D.term = "*heart", (D.type = "synonym" OR D.type = "fullySpecifiedName"), D.acceptability = * **LANGUAGE** X
- **textDefinition**
  - textDefinition = "*heart"
    - D.term = "*heart", D.type = "definition"
    - D.term = "*heart", D.typeId = 900000000000550004 |definition|
  - textDefinition = "*heart" **LANGUAGE** X
    - D.term = "*heart", D.type = "definition", D.acceptability = * **LANGUAGE** X
    - D.term = "*heart", D.typeId = 900000000000550004 |definition|, D.acceptabilityId = * **LANGUAGE** X
- **Unacceptable Terms**
  - (D.term = "*heart") MINUS (D.term = "*heart", D.acceptability = * **LANGUAGE** X)

		Language preferences using multiple language reference sets - **LRSs that use the same Language tend to use 'Addition' - i.e. child LRS only includes additional acceptable terms, but can override the preferred term** - **E.g. Regional LRS that adds local dialect to a National LRS** - **E.g. Specialty-specific LRS** - **E.g. Irish LRS that adds local preferences to the en-GB LRS** - 99999900 |Irish language reference set| **PLUS** |GB English reference set| - **LRSs that define a translation to a different language tend to use 'Replacement' - i.e. child LRS replaces set of acceptable and preferred terms for any associated concept** - **E.g. Danish LRS that does a partial translation of the International Release** - 999999 |Danish language reference set| **ELSE** |GB English reference set|
Next steps	Linda Bird	- Discuss and plan next steps
Confirm next meeting date /time	Linda Bird	

File	Modified
Microsoft Excel Spreadsheet RegexCheat.xlsx	2019-Oct-16 by Linda Bird