1. Introduction

Background

SNOMED CT is a clinical terminology with global scope covering a wide range of clinical specialties and requirements. The use of SNOMED CT expressions in Electronic Health Records (EHRs) provides a standardized way to represent clinical meanings captured by clinicians and enables the automatic interpretation of these meanings. SNOMED CT expressions are a structured combination of one or more concept identifiers used to represent a clinical idea in a logical manner. The SNOMED CT Compositional Grammar provides a lightweight syntax for the representation of SNOMED CT expressions.

In contrast, a SNOMED CT Expression Constraint is a computable rule that can be used to define a bounded set of clinical meanings represented by either precoordinated or postcoordinated expressions. Expression constraints can be used as formal constraints on the content of a particular data element in an EHR, as the intensional definition of a concept-based reference set, as a machine processable query that identifies a set of matching precoordinated or postcoordinated expressions, or as a constraint that restricts the range of an attribute defined in the SNOMED CT concept model.

Purpose

The purpose of this document is to define and describe a formal language for representing SNOMED CT Expression Constraints. A SNOMED CT Expression Constraint is a computable rule that defines a bounded set of clinical meanings represented by either precoordinated or postcoordinated expressions. Two equivalent syntaxes are presented – a brief syntax, which is designed to be as compact as possible for interoperable communication between systems, and a long syntax, which introduces textual alternatives to the symbols from the brief syntax. This document also provides examples and guidance to assist in the implementation of this language.

Scope

This document presents the specification of an Expression Constraint Language, which can be used to represent SNOMED CT Expression Constraints. It includes a logical model of the language, two syntaxes, a set of example expression constraints and a summary of implementation considerations.

The Expression Constraint Language specified in this document is part of a consistent set of computer processable languages designed to support a variety of use cases involving the use of SNOMED CT. Other SNOMED CT computable languages include:

- Compositional Grammar: designed to represent SNOMED CT expressions; and
- Template Syntax: which allow slots to be added to expressions, expression constraints or queries that can be filled with specific values at a later time.

The compositional grammar is designed to provide a common foundation for the additional functionality added by the other languages.

This document does not include a full description of how to implement an expression constraint parser, classifier or interpreter. It does not describe how to transform an expression constraint into other languages, such as OWL, SPARQL or SQL; or how to determine whether two expression constraints are equivalent. It also does not describe how to implement an EHR which uses expression constraints to constrain or query its content, or a terminology server which uses expression constraints to query its content. Instead, it provides a specification, examples and general guidance to assist in the implementation of expression constraints in any of these applications.

This document defines and describes the current version of the Expression Constraint Language - ECL 2.1.

History

Expression constraints have been used in projects and programs around the world for a number of years – for example HL7 TermInfo, and the NHS Logical Record Architecture.

In 2013, a draft document on "SNOMED CT Expression Constraint Syntax Specification for Terminology Binding" was developed as an assignment for the SNOMED CT Implementation Advisor (SIA) scheme.

In 2014, this work was revised and extended to support a wider range of relevant use cases to produce version 1.0 of the Expression Constraint Language specification (2015). These updates included:

- Concrete values (e.g. integers, decimals and strings) are now permitted as attribute values. This is to provide alignment with the recent extensions to SNOMED CT Compositional Grammar;
- Cardinality constraints have been introduced, and as a result the optional operator (i.e. ~ ) is no longer provided;
- Attributes may now be preceded by a ‘descendantOf’ or ‘descendantOrSelfOf’ operator to indicate whether attribute descendants and/or the attribute itself should be used in the matching process;
- A reverse flag has been introduced, which allows relationships to be traversed in the reverse direction;
- Exclusion has been changed from a unary operator (‘negation’) to a binary operator (‘minus’);
- A wildcard character (‘*’) has been introduced to represent any concept in the substrate;
- A number of clarifications have been made, including the ‘memberOf’ operator and the default substrate upon which the expression constraints are executed.

An update to the Expression Constraint Language was then published in 2016 (version 1.1) to incorporate some additional features requested by implementers of the language. These updates include:
• Two new operators ‘childOf’ and ‘parentOf’ were added to support querying immediate children and immediate parents of a concept during user interface design.
• A new ‘dot notation’ was introduced (as an alternative to the Reverse flag) to refer to an attribute value for a concept or expression;
• The ability for a constraint operator (e.g. ‘descendantOf’) to be applied to a nested expression constraint was added;
• The ability to add comments within the text of an expression constraint was added;
• Additional optional brackets were allowed around subexpressions; and
• The non-normative syntax (previously named the ‘Full Syntax’) was renamed to the ‘Long Syntax’.

Early in 2017 version 1.2 was published, to include a new feature requested by implementers: namely, the ability for the ‘memberOf’ function to be applied to a set of reference set concepts defined using an expression constraint. In this version, the explanation of Operator Precedence was also moved from section 6.7 to section 5.4. Version 1.3 was then published in mid 2017 to support a range of additional features - including allowing the refinement of subexpression constraints, permitting the use of subexpression constraints to represent a set of valid attribute names and simplifying the parsing of dotted expression constraints.

In mid 2020, version 1.4 was published to support boolean attribute values and to introduce the ‘childOrSelOf’ and ‘parentOrSelOf’ operators. Later that year, version 1.5 was published to support description filter constraints. These constraints filter the result set, by matching only on concepts which have a description that satisfies the filter criteria. Section 5.5 (Character Collation for Term Filters) and section 6.8 (Filter Constraints) were added in ECL version 1.5.

In 2021, version 1.6 added concept filters, which allow the result set to be filtered based on the definition status, module, effectiveTime and active status of each concept.

And then in early 2022, version 2.0 was published. Version 2.0 includes a number of significant features, including:

• History supplements, to supplement the results with relevant inactive concepts,
• Reference set member filters, to filter the rows of a reference set, based on the value of specified fields,
• Support for returning multiple fields of a reference set, including fields other than the referencedComponentId,
• Support for module, effectiveTime and active filters on descriptions, and
• Support for word-prefix-any-order and wildcard searches for string-based concrete attribute values (for consistency with term searches in a Description filter).

Most significantly, version 2.0 is the first version of ECL that is specifically designed to support querying over historical patient records, which may contain inactive codes.

In August 2022, version 2.1 was published to allow description filters to filter results using description identifiers, and to harmonise the dialect alias filter (see Appendix C) with BCP-47 (Internet Best Current Practice Specification).

In November 2023 version 2.2 was published. This version added the ability to reference concepts using alternate identifiers, and also two convenience methods for finding the top (root) or bottom (leaf) concepts within a set.

For a list of previous PDF versions, please refer to Previous Versions.

Audience

The target audiences of this document include:

• SNOMED National Release Centres;
• SNOMED CT designers and developers, including designers and developers of EHR systems, information models, data entry interfaces, storage systems, decision support systems, retrieval and analysis systems, communication standards and terminology services;
• SNOMED CT terminology developers, including concept model designers, content authors, map developers, subset and constraint developers and release process managers.

It should be noted that this document contains both technical and non-technical content. In particular, the detailed logical model and formal syntax is specifically focussed at more technical readers. Less technical readers are encouraged to read the introductory material (including the use cases and requirements) and the extensive set of examples that is presented. It should also be noted that even though complex expression constraints are possible, most expression constraints are likely to be very simple, such as those described in Simple Expression Constraints.

Document Overview

This document defines the SNOMED CT Expression Constraint Language and describes how and where it may be implemented. Chapter 2 begins by describing the use cases in which it is anticipated that SNOMED CT Expression Constraint Language will be used. Chapter 3 then describes the requirements used to guide the definition of this language. In Chapter 4, the logical model of the Expression Constraint Language is presented, while in Chapter 5 two syntaxes are defined using an ABNF serialisation of the logical model. Chapter 6 then presents some examples of expression constraints that conform to the SNOMED CT Expression Constraint syntaxes, and Chapter 7 discusses some implementation considerations. Appendix A – Examples Of Valid Expressions provides some examples of pre-coordinated and post-coordinated expressions that satisfy each of the expression constraints presented earlier in the document. Appendix B – Examples Of Invalid Expressions then provides some examples that do not satisfy these expression constraints. Appendix C - Dialect Aliases provides a list of example aliases that may be used to specify a particular dialect in an ECL filter constraint. Appendix D - ECL Quick Reference provides a quick reference to the key syntax features of the Expression Constraint Language. And finally, Appendix E - Reference Set Fields explains how reference set field names are used in ECL 2.0+. 