SNOMED Clinical Terms®
Developer Toolkit Guide

July 2014 International Release

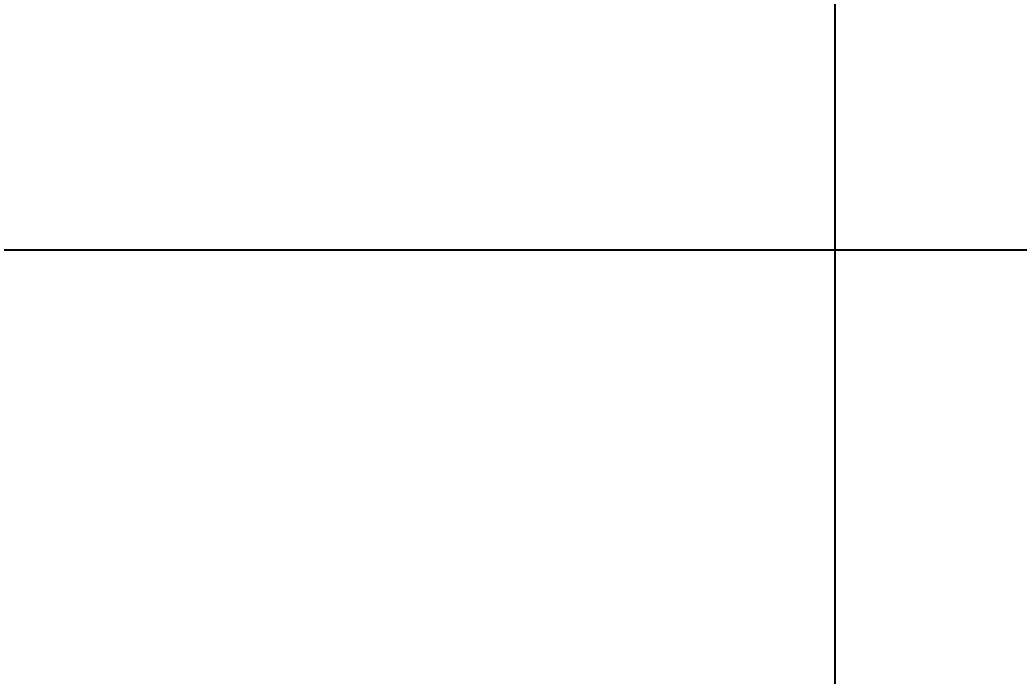


Table of Contents

Document History

Version	Notes
Version 2.1 (July 2004)	First Publication of Enhanced Developer Toolkit, which includes the Index Generator (Appendix E). The Version numbering begins with 2.1 to correspond with the existing Developer Toolkit.
Version 2.2 (January 2005)	• Updated copyright • Updated contacts
Version 2.3 (January 2007)	• Updated copyright • Updated contacts • Keyword field in Excluded Words table updated
July 2007	• Updates to reflect transfer of IP to the International Health Terminology Standards Development Organisation • Removal of references to College of American Pathologists (CAP) derivative products
January 2008	Updated for January 2008 International Release
July 2008	Updated for July 2008 International Release
January 2009	Updated for the SNOMED CT International Release
July 2009	Updated for the SNOMED CT International Release
January 2010	Updated for the SNOMED CT International Release
July 2010	Updated for the SNOMED CT International Release
January 2012	Updated for the SNOMED CT International Release
July 2012	Updated for the SNOMED CT International Release
January 2013	Updated for the SNOMED CT International Release
July 2013	Updated for the SNOMED CT International Release
January 2014	Updated for the SNOMED CT International Release
July 2014	Updated for the SNOMED CT International Release

Inventory of Documentation

The following essential SNOMED CT documentation is currently available as part of the International Release of SNOMED CT from the International Health Terminology Standards Development Organisation (IHTSDO):

SNOMED CT Technical Implementation Guide (TIG)

The TIG is intended for SNOMED CT implementers, such as software developers and designers. The TIG assumes information technology and software development experience. Clinical knowledge is not required, although some background is helpful to understand the application context and needs.

The TIG contains reference material related to the current release of SNOMED CT and includes file layouts, field sizes, required values and their meanings, and high-level data diagrams. In addition, it contains guidelines and advice about the design of applications using SNOMED CT, and covers topics such as terminology services, entering and storing information, and migration of legacy information.

SNOMED CT User Guide

The User Guide is intended for clinical personnel, business directors, software product managers, and project leaders; information technology experience, though not necessary, can be helpful.

The User Guide is intended to explain SNOMED CT's capabilities and uses from a content perspective. It explains the content and the principles used to model the terminology.

Additional Documentation

The following supplementary documentation is also included as part of the International Release of SNOMED CT:

- SNOMED CT Editorial Guide
- SNOMED CT Canonical Table Guide
- SNOMED CT Developer Toolkit Guide
- SNOMED CT Namespace Registry

1 Introduction

1.1 Purpose

This document describes the existing file structure of the files contained in the SNOMED Clinical Terms® Developer Toolkit.

- For more information about the SNOMED CT release files, please see the SNOMED CT Technical Reference Guide.
- For implementation guidance, please see the SNOMED CT Technical Implementation Guide.
- For more information about terminology content, see the SNOMED CT User Guide.
- For more information about incremental changes in each new release, see the Scope Memo and Release Notes for that release. For a list of files and file sizes, see the "readme" file for that release.

1.2 Who should read this guide?

The intended audience for this document is any individual or any organization that wishes to develop or use systems that will use SNOMED Clinical Terms and use the Developer Toolkit. This document is to provide a reference about the SNOMED CT technical structure for:

Software developers

- Developers of fully integrated applications
- Developers of terminology servers
- Developers of applications that use terminology

Health informatics specialists, analysts, purchasers, and integrators

- Health informatics specialists analyzing the needs of users and organizations
- Purchasers of healthcare information systems
- Healthcare information systems implementers and integrators
- Standards developers

1.3 Scope and Format

This document uses material from the SNOMED CT technical specifications that were used to create the work. Additional functions will be added to this document as they are delivered in SNOMED CT. Any functions marked "For Future Use" are not implemented in this release.

1.4 Feedback

Further information about SNOMED CT is available on the Internet at

www.ihtsdo.org

Please send feedback by email to:

support@ihtsdo.org

or contact:

IHTSDO
Gammeltorv 4, 1. 1457
Copenhagen K
Denmark

Tel: +45 3644 8736
Fax: +45 4444 8736

2 Introducing Index and Search Support Tables

2.1 Overview

Effective implementation of SNOMED CT depends on the ease and speed with which users can locate the terms and Concepts that they wish to use. An essential contribution to meeting this requirement is the ability to perform rapid and flexible text searches.

A set of word search tables (indexes) is included in the Developer Toolkit. These tables are designed to facilitate development of effective search facilities while reducing duplication of effort. However, neither these tables, nor indices derived from them, are sufficient to meet the full range of search requirements. Meeting the needs of different users for appropriate methods of locating particular Concepts is an area in which competitive development is expected and welcomed. Developers may choose to use some or all of the word search tables distributed with SNOMED CT or may develop their own solutions independent of these tables.

The intention of the word search tables is to identify candidate matches among the Descriptions (or Concepts) of SNOMED CT. An application or coding engine will apply further filtering to these candidate matches to identify the matches to be selected or displayed. A balance must be made between specificity and completeness of a search. The keyword algorithm is intended to maximize the likelihood that the required Concept will be included in the candidate matches rather than to achieve precision.

Applications may filter candidate matches using techniques that are many and varied. Some may take account of non-textual characteristics (e.g. Subsets, subtype Relationships or Relationships) while others use more complex textual techniques (e.g. word order dependence, case dependence, complete phrase matching, regular pattern recognition, Soundex). These extended text search techniques are beyond the scope of the keyword generation algorithm.

The algorithm for keyword generation is only applicable for English and other western European languages. It is not intended to apply to Russian, Greek, Slavic or to any non-European languages.

Please refer to the Technical Implementation Guide for additional search implementation guidance.

2.1.1 Search Requirements

Development of user-interfaces that facilitate rapid and appropriate access to SNOMED CT is a legitimate area for competition between application suppliers. However, user perceptions of the quality, usability and value of a terminology depend in a large measure upon the nature and performance of the user interface. Therefore, the International Health Terminology Standards Organisation has an interest in guiding and facilitating this development.

Rapid and appropriate access to SNOMED CT is not simply a matter of providing indices of the type specified in this document. The following aspects of functionality also need to be addressed in ways that are appropriate to the anticipated users of a SCT Enabled Application.

- Support for searches by keywords, phrases and patterns including:
  - Word-form and word-order variants
  - Abbreviations
  - Word equivalents
  - User acronyms for commonly used Terms
- Integration of word and phrase searches with sub-type and part-of hierarchies:
  - Hierarchy navigation to narrow or broaden a Concept found by a search
  - Restricting searches to particular descendants of chosen Concepts

- ✧ Limiting “double hits” caused by synonyms and multiple siblings with identical word matches
- ❖ Integration of word and phrase searches with Defining characteristic and Qualifying characteristics
  - ✧ Generating a post-coordinated SNOMED CT representation from a phrase
  - ✧ Restricting searches to appropriate qualifiers of a selected Concept
- ❖ Integration of word and phrase searches with Subsets
  - ✧ Restricting searches to specified Subsets of Descriptions or Concepts used in:
    - A language or dialect
    - A country, organization, specialty or user
    - A particular context in a record or protocol
  - ✧ Prioritizing matches according to memberships of one or more Subsets

The Technical Implementation Guide offers detailed advice on searching and related user-interface issues.

2.1.2 Word Search (Keyword) Generation Algorithm

2.1.2.1 Introduction

This section describes the algorithms used to generate the keywords and dualkeys that populate word search tables. These algorithms are relevant to application developers because similar algorithms must be applied to search phrases to enable effective searches using indices based on these tables.

The algorithm for keyword generation is only applicable for English and other western European languages. It is not intended to apply to Russian, Greek, Slavic or to any non-European languages.

The keyword generation process consists of the following steps:

- ❖ Single keyword generation
  - ✧ Obtain source text
  - ✧ Break source text into separate words
  - ✧ Remove excluded words
  - ✧ Process special characters
  - ✧ Limit keyword length
  - ✧ Remove duplicates
  - ✧ Result ➔ keyword list
- ❖ Dualkey generation
  - ✧ Obtain short keyword list
  - ✧ Remove duplicates
  - ✧ Create word pair permutations
  - ✧ Result➔ dual key list

2.1.2.2 Single keyword generation

Obtain Source Text

For the DescWordKey Table and DescDualKey Table the source text is derived from the Term of a single Description. Any lowercase characters are translated so that the source text consists only of upper case characters.

For the ConcWordKey Table and ConcDualKey Table the source text is derived from the Terms associated with several Descriptions using the following process:

- Locate all Descriptions for the same concept using the ConceptId
  - Choose only descriptions with DescriptionStatus = 0 (current)
    - Both US and UK dialect descriptions are used to generate the English Language tables
- Concatenate the Terms for each of these Descriptions including a single space between each Term
- Translate any lowercase characters so that the source text consists of only upper case characters

Break Source Text into Separate Words

The keywords in a term are derived by breaking the term at any of a set of identified characters. The separator characters are discarded. There are two types of separators:

- Simple separators
  - These are always treated as a break between words
  - The simple separators are listed in Table 1
- Context-dependent separators
  - These require a rule to be tested to determine if they are to be treated as a break between words or require specialized processing
  - The context dependent separators and associated rules are listed in Table 7

Table 1 – Simple separators

Character Name	Characters
Space	
Punctuation (excluding full-stop (period) – see below)	**, ; : ! ?**
Brackets (including both opening and closing brackets – all shapes)	**( ) [ ] { } < >**
Double Quotes	**“ ”**

Table 2 – Context-dependent separators

Character Name	Character	Rules
Period (full-stop)	**.**	If immediately preceded and followed by single character or if the ending period of an abbreviation which has other embedded periods: ❖ Period character deleted Examples: "M.I." becomes "MI" "C.H.D." becomes "CHD" Otherwise ❖ Treated as a separator
Hyphen (dash or minus)	**-**	If neither the immediate preceding nor following characters are separators ❖ Generates two keys ✧ Joining preceding and following words ✧ The word following the hyphen Example: "BETA-BLOCKER" creates two keywords "BETABLOC" "BLOCKER" Otherwise ❖ Treat as a separator
Slash (oblique)	**/**	If neither the immediate preceding nor following characters are separators: ❖ Generates two keys ✧ Treating the slash as a regular character ✧ The word following the dash Example: "MMOL/LITRE" creates two keywords "MMOL/LIT" "LITRE" Note: If the word before the slash is hyphenated, start the keyword at the beginning word -- for example, BETABLOC not BLOCKER in the hyphen example above. Otherwise ❖ Treat as a separator.
Ampersand	**&**	Replaced by Plus "+" and treated as described below.
Plus	**+**	If immediately preceded and followed by single character word: ❖ Treated as single word with intervening spaces deleted: Examples: "D+V" "D +V" "D & V" and "D&V" all generate the key "D+V" Otherwise ❖ Treated as a separator

Additional Algorithm Details

Remove Excluded Words

- Potential keywords are removed from consideration if they:
  - Consist of a single character
  - Begin with a numeral
  - Are present in the Excluded Words Table (with a relevant LanguageCode).

Process Special Characters

General purpose symbols

All symbols in the standard ASCII 7-bit set other than letters and numerals and the symbols referred to above as separators are deleted without creating a word separation. The symbols treated in this way are **# $ % ' * = @ \ ^ ` | ~**

Thus "doctor's" generates the Keyword "DOCTORS".

Accented character and umlauts

Accented characters are replaced by their unaccented equivalents.

This rule also applies to umlauts (i.e. "Köhler" produces the keyword "KOHLER" **not** the technically more correct keyword "KOEHLER").

Diphthongs

Diphthongs are converted to the equivalent letter pair "æ" to "ae".

Greek characters

Greek characters such as *ß* generate keywords based on the full English spelling of the character name.

For example, "*ß*-carotene" generates keys as though it were spelled in full as "beta-carotene". Due to the rules for processing the hyphen (see Table 2) this generates two keys "BETACARO" and "CAROTENE".

All other symbols

All other characters are deleted for keywords related to western alphabet versions of SNOMED CT. Additional characters may be included in Keywords for non-western alphabets.

Limit Keyword Length

Any keywords that are more than eight characters in length are truncated to the first eight characters.

Removal of Duplicate word

Any duplicate keywords are removed from the list.

For example, if a Concept with the Preferred Term "Renal stone" and a Synonym "Kidney stone" will have keys for "RENAL", "KIDNEY" and "STONE." The repetition of "STONE" will not result in an additional keyword entry for that Concept.

Dualkey generation

Obtain short keyword list

The short keyword list for a particular Concept or Description is obtained by following the algorithm specified for single keywords. Each keyword is then shortened to its first three characters. Any two-character keywords are extended to three characters in length by addition of a trailing space.

Remove duplicates

Duplicates are removed in the generation of single keywords but new duplicates may arise as a result of shortening the keywords. These additional duplicates are removed at this stage.

For example, "Meningococcal meningitis" would result in the two keywords "MENINGOC" and "MENINGIT" but only one short key "MEN."

Create word pair permutations

- Dualkeys are produced by concatenating two short keywords according to the following rules:
  - Every possible pair of short keywords for a Concept or Description is used to generate a dualkey.
  - The order or proximity of the words in the source text has no effect on the dualkey.
  - The short keyword with the lowest ASCII sort order always appears first in a dualkey.

Examples

"Lower abdominal pain" creates the following three dualkeys:

- ABDLOW
- ABDPAI
- LOWPAI

"Severe MI" creates the following dualkey:

- MI SEV

Because of the sort order rule the following **cannot** exist as dualkeys

- LOWABD
- PAIABD
- PAILOW
- SEVMI

2.2 Word Search Tables and Index Generator – Summary

The following five tables are included in the Developer Toolkit of SNOMED CT. These tables are derived from the SNOMED CT Descriptions Table. The LanguageCode of the Descriptions Table is used to choose only descriptions for a language.

Excluded Words Table	Each row in this table is a word excluded from the list of possible keywords and dualkeys. Words are excluded if they are frequently used and are so limited in semantic specificity that they impair rather than enhance searches.
DescWordKey Table	Each row in this table is a word followed by a reference to a Description in which this word appears.
ConcWordKey Table	Each row in this table is a word followed by a reference to a Concept. A Concept is referenced if the word appears anywhere in the combination of the Fully Specified Name with any Preferred Term or Synonym.
DescDualKey Table	Each row in this table is a six-character string representing the first three letters of a pair of words followed by a reference to a Description in which these two words appear.
ConcDualKey Table	Each row in this table is a six-character string representing the first three letters of a pair of words followed by a reference to a Concept. A Concept is referenced if both words appear anywhere in the combination of the Fully Specified Name with any Preferred Term or Synonym.

All keywords are regarded as case independent and are presented in the word search tables in upper case. Case dependent searching can be applied by appropriately filtering the candidate matches.

The Index Generator creates the word key and dual key indexes. It uses as input a Descriptions Table in standard SNOMED CT format and an Excluded Words Table. See Appendix E for details.

2.3 Word Equivalents

2.3.1 Introduction

The Word Equivalent Table is included in the Developer Toolkit of SNOMED CT. It supports enhanced searches that take into account semantically similar words such as KIDNEY and RENAL. It also provides commonly used abbreviations.

This table can be used by implementers to offer additional search capability in applications without greatly increasing the volume of synonyms. It is not intended as a comprehensive dictionary of words. Many searches can be completed without using this table; like other word search tables, it is completely optional and can be used as an example of a capability that may be customized and extended by SNOMED implementers.

2.3.2 Word Equivalents Tables – Summary

Key Fields	
WordBlockNumber	A 32-bit integer shared by a set of equivalent words or phrases. The WordBlockNumber links together several rows that have an identical or similar meaning.
WordText	A word, phrase, acronym or abbreviation that is equivalent to the WordText of other rows that share the same WordBlockId.
Data Fields	
WordType	An integer indicating the type of equivalence.
WordRole	An integer indicating the usual role of this word. This should be considered if attempting to find a post-coordinated combination of Concepts that matches a phrase.

2.4 Duplicate Terms Table

The Duplicate Terms Table supports enhanced searches that take into account terms that are lexically similar for more than one SNOMED concept. This table can be used to prioritize the display or use of concepts when a text-based search (or other means of access) results in multiple SNOMED concepts. The table indicates which of the concepts is favored for that term. As an example, for the term "liver," the concept "liver structure" is favored over "entire liver." This information can then be used within the software application to choose the favored concept, or at least present it as a recommendation. It is expected that the content of this table will need to be modified since what is preferred in one context may not be preferred in another.

The Duplicate Terms Table contains a list of duplicate terms (text strings) sourced from SNOMED's current synonym and preferred terms. Some of the heuristics used to prioritize the terms is as follows. In addition, priorities were assigned manually.

Only current descriptions were used to create the Duplicate Terms Table.

Preferred Term Priority

If two concepts have the same term and one is a Preferred Term while the other is a Synonym, then the Preferred Term has the higher priority.

Supertype Priority

If two concepts have the same term and one is a supertype of another, then the supertype is given priority.

Finding vs. Disorder

If one concept is a Finding that is the definitional-manifestation of the Disorder, give priority to the Disorder.

Morphology vs. Disorder

Disorder is given priority.

Observable vs. Finding

Observable is given priority.

Finding vs. Morphology

Finding is given priority.

Finding vs. Body Structure or Cell

Finding is given priority.

Dietary Substance vs. Organism

Dietary Substance is given priority.

3 Introducing Navigation Subsets

Navigation subsets allow SNOMED CT concepts to be organized and displayed in alternative hierarchies. The use and implementation of navigation subsets is explored in the Technical Implementation Guide. See Appendix B for the file description, and Technical Reference Guide for details about the subset structure.

3.1 Navigation Subsets

Two example navigation subsets are included in the Developer Toolkit to illustrate how these subsets are defined.

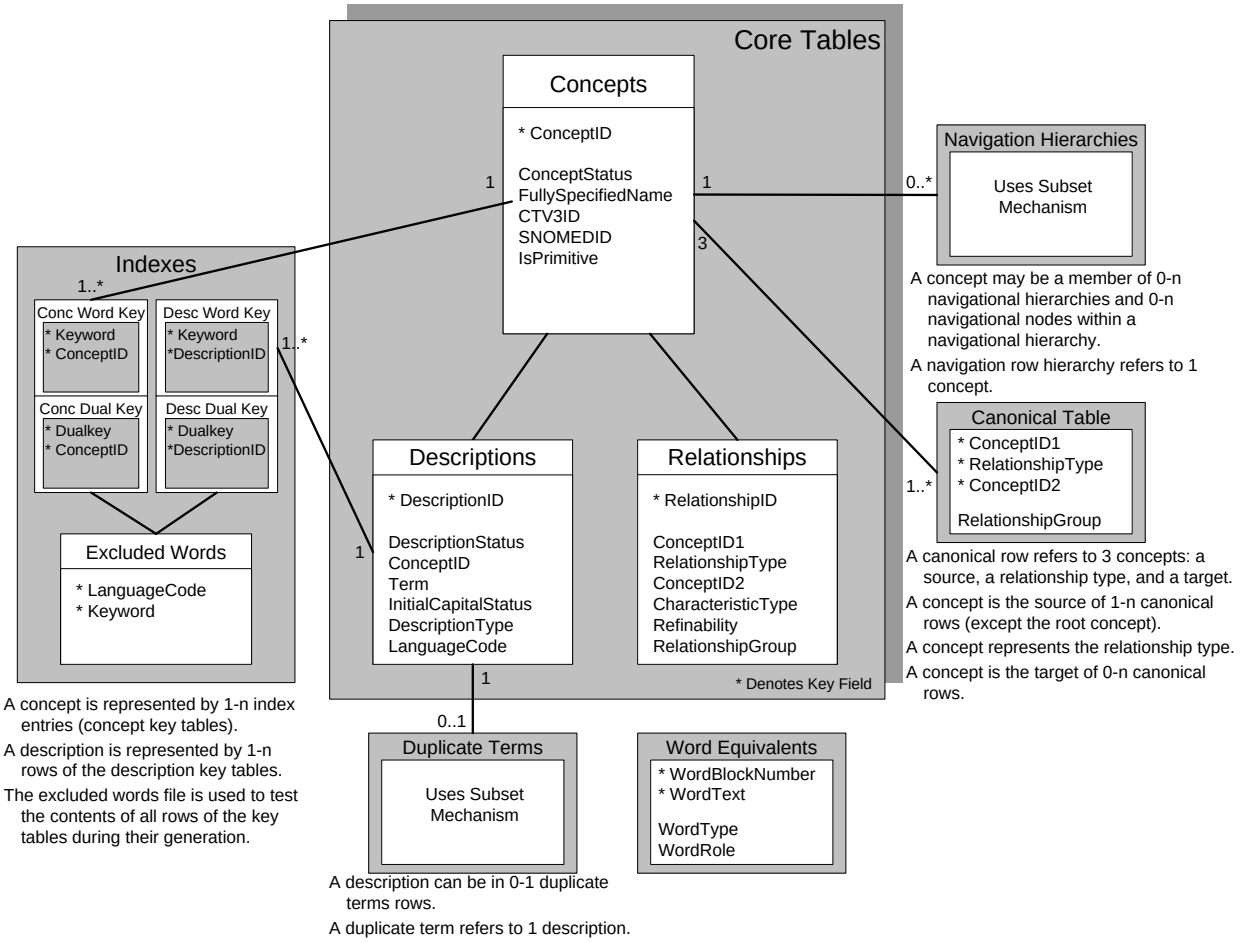
CTV3

This hierarchy shows SNOMED CT concepts as they were organized in the Clinical Terms Version 3 (Read Codes) terminology, one of the predecessor terminologies of SNOMED CT.

Top level

This hierarchy shows the SNOMED CT top level concepts.

Appendix A. Developer Toolkit Structure



Appendix B. Table Structure Details

B.1 ConcWordKey Table

ConcWordKey Table			
Each row in this table is a word followed by a reference to a Concept. A Concept is referenced if the word appears anywhere in the combination of the Fully Specified Name, Preferred Term, or Synonyms. A keyword can relate multiple Concepts. If there is a duplicate row where the keyword and ConceptId appear multiple times, the duplicates are omitted. Only one row is present for each combination of a keyword with a particular ConceptId, even if that keyword appears several times in an associated Description.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
Keyword	*String*	*0 to 9, a to z, A to Z and* dash "-"	*1 to 8*
A word used in the Term field of one or more Descriptions associated with the referenced Concept. Keywords are represented using only upper case letters and words of more than eight characters are represented as their first eight characters only.			
ConceptId	***SCTID***	*Digits* 0 *to* 9 *only*	*6 to 18*
The unique SNOMED Clinical Terms Identifier for a Concept that has one or more current Descriptions containing this keyword. *Note:* Current Descriptions are those with DescriptionStatus = 0.			

B.2 DescWordKey Table

DescWordKey Table			
Each row in this table is a word followed by a reference to a Description in which this word appears. A keyword may relate to multiple descriptions. If there is a duplicate row where the Keyword and DescriptionId appear multiple times, the duplicates are omitted. Only one row is present for each combination of a keyword with a particular DescriptionId, even if that keyword appears several times in the referenced Description.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
Keyword	*String*	*0 to 9, a to z, A to Z and* dash "-"	*1 to 8*
A word used in the Term field of the referenced Description. Keywords are represented using only upper case letters and words of more than eight characters are represented as their first eight characters only.			
DescriptionId	***SCTID***	*Digits* 0 *to* 9 *only*	*6 to 18*
The unique SNOMED Clinical Terms Identifier for a Description that contains a keyword.			

B.3 ConcDualKey Table

ConcDualKey Table			
Each row in this table is a six-character string representing the first three letters of a pair of words followed by a reference to a Description in which these two words appear. Only one row is present for each combination of a dualkey with a particular ConceptId, even if that dualkey appears several times in the related Descriptions.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
Dualkey	*String*	*0 to 9, a to z, A to Z and* dash "-"	*6*
A string derived from a concatenation of the first three letters of a pair of words used in the Term field of the referenced Concept.			
ConceptId	***SCTID***	*Digits* 0 *to* 9 *only*	*6 to 18*
The unique SNOMED Clinical Terms Identifier for a Concept with associated current Descriptions that contain both the words represented by this dualkey. *Notes:* Current Descriptions are those with DescriptionStatus = 0.			

B.4 DescDualKey Table

DescWordKey Table			
Each row in this table is a six-character string representing the first three letters of a pair of words followed by a reference to a Description in which these two words appear. Only one row is present for each combination of a dualkey with a particular DescriptionId, even if that dualkey appears several times in the referenced Description.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
Dualkey	*String*	*0 to 9, a to z, A to Z and* dash "-"	*6*
A string derived from a concatenation of the first three letters of a pair of words used in the Term field of the referenced Description.			
DescriptionId	***SCTID***	*Digits* 0 *to* 9 *only*	*6 to 18*
The unique SNOMED Clinical Terms Identifier for a Description that contains both the words that compose this dualkey.			

B.5 Excluded Words Table

ExcludedWords Table			
Each row in the Excluded Words Table is a word excluded from the list of possible keywords and dualkeys. Words are excluded if they are frequently used and are so limited in semantic specificity that they impair rather than enhance searches. Word exclusions are language or dialect specific.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
LanguageCode	*String*	*0 to 9, a to z, A to Z and* dash "-"	*1 to 8*
A string identifying a language and or dialect in the word is excluded from keyword generation. Consists of a code and optionally a sub-code. If a sub-code is present it is separated from the code by a dash ("-"). ❖ The code is the ISO639 language code, which is a string of two lower-case letters. ISO639 is the International Standard for "Codes for the representation of names and languages." ❖ The sub code is a string of upper-case letters. This will either be: ✧ A two-letter ISO3166 country code. ISO3166 is the International Standard for "Codes for the representation of names of countries." ✧ A string of more than two letters, which is registered with IANA as a sub code for the language. IANA is the Internet Assigned Numbers Authority. This structure follows Internet conventions. Examples: "en" for "English," "es" for Spanish, "en-US" for United States English, "en-GB" for British English.			
Keyword	*String*	*0 to 9, A to Z and dash* "-"	*1 to 8*
A word used in Descriptions in the Descriptions Table but excluded from keyword generation. Words are represented using only upper case letters and words of more than eight characters are represented as their first eight characters only.			

B.6 Word Equivalents Table

Word Equivalents Table			
Each row in this table represents the potential equivalence between a word, phrase, or abbreviation and other words, phrases or abbreviations.			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
WordBlockNumber	*Integer*	*0 to 9*	*1 to 10*
A 32-bit integer shared by a set of equivalent words or phrases. The words, phrases and abbreviations that share a common WordBlockId value are interchangeable for the purposes of searches. *Example:* An equivalent block could contain the following: “TB”, “tuberculosis”, “tuberculous” *Note:* WordBlockId is not maintained as a unique identifier across releases. It should only be regarded as an index to link equivalents in the context of a particular release.			
WordText	*String*	*0 to 9 only, A to Z and dash “-”*	*1 to 8*
A word, phrase or abbreviation that is equivalent to the WordText of other rows that share the same WordBlockId. *Note:* If a word or phrase has two or more possible meanings it may be represented by more than one row in this table. Each row containing the same WordText must be associated with a different WordBlockId value.			
Data Fields	**Field Type**	**Permitted Characters**	**Length**
WordType	*Enumerated*	*See listed values*	*2*
An integer indicating the type of equivalence *Values (suggested)* 0 unspecified 1 word form variant (e.g. "abdomen", "abdominal") 2 word equivalents (e.g. "renal", "kidney") 3 abbreviation or acronym (e.g. "MI" → "myocardial infarction") 4 equivalent phrase (e.g. "MI" → "myocardial infarction")			
WordRole	*Enumerated*	*See listed values*	*2*
An integer indicating the usual role of this word. This should be considered if attempting to find a post-coordinated combination of Concepts that matches a phrase. *Values (suggested)* 0 unspecified 1 general qualifier 2 topography 3 topography qualifier 4 object (including organism or substance) 5 action 6 unit of measure *Note:* All rows with the same WordBlockId value must have same WordRole			

B.7 Duplicate Term Subset Members Table

Duplicate Terms Subset Members Table			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
SubsetId	***SCTID***	*0 to 9*	*6 to 18*
The unique SNOMED CT Identifier for the Subset to which this applies			
MemberId	***SCTID***	*0 to 9*	*6 to 18*
The Description Identifier for a SNOMED CT Concept that has a duplicate term.			
MemberStatus	*Integer*	*0 to 9*	*1 to 5*
The status of identified member in this Subset. The value of MemberStatus must be greater than zero.			
Duplicate Terms Subset	The priority assigned to this Concept. *Notes:* The *lower* the value, the *greater* the priority. Thus the highest priority is assigned to Concepts with the MemberStatus value “1” (first). The priority should be used by applications to determine the items to be displayed first or selected most readily.		
Data Fields	**Field Type**	**Permitted Characters**	**Length**
LinkedId	***SCTID***	*0 to 9*	*6 to 18*
Duplicate Terms Subset	The MemberId (and therefore, the DescriptionId) of the Member with the same Term that has the highest priority. A group of duplicate terms can be identified since they will all point to the same Member ID.		

B.8 Navigation Subset Members Table

Navigation Subset Members Table			
Key Fields	**Field Type**	**Permitted Characters**	**Length**
SubsetId	***SCTID***	*0 to 9*	*6 to 18*
The unique SNOMED CT Identifier for the Subset to which this applies.			
MemberId	***SCTID***	*0 to 9*	*6 to 18*
The identifier of a SNOMED CT Concept.			
MemberStatus	*Integer*	*0 to 9*	*1 to 5*
The status of the identified member in this Subset. The value of MemberStatus must be greater than zero.			
Navigation Subset	The MemberStatus specifies the order of the child Concepts within the set of Navigation Links from the same parent Concept. The combination of SubsetId and MemberId and MemberStatus forms the unique key.		
Data Fields	**Field Type**	**Permitted Characters**	**Length**
LinkedId	***SCTID***	*0 to 9*	*6 to 18*
Navigation Subset	The ConceptId of a Navigation child of the Concept identified by the MemberId.		

Appendix C. Components of the Developer Toolkit

Several components of SNOMED Clinical Terms® have been collected into a Developer Tool Kit to assist software application developers.

The components of the Developer Toolkit include:

Category	Component	Comments
Indexes	Excluded Words Table Description Word Key Table Concept Word Key Table Description Dual Key Table Concept Dual Key Table	These indexes have been pre-generated for the English language using the SNOMED CT® Descriptions Table.
Navigation Hierarchies/ Subsets	Two samples provided: • SNOMED CT top level hierarchies • CTV3 navigation	
Additional Components	Duplicate Terms Table Word Equivalents Table Index Generator	

Appendix D. Distribution Table Technical Summary

D.1 Keys

Colored blocks in the keys column identify keys that may be required or useful for effective implementation.

- Primary keys (blue),
- Recommended keys:
  - Critical fields (red) are required for the relevant functionality.
  - User fields (green) are not essential but may improve functionality.
- Foreign keys used in joins to other tables (purple).

A number is used to indicate the order of fields in combined keys. A letter in the primary key identifies this as a target for a foreign key pointer. The same letter in a foreign key indicated a join based on the primary key identified by the same letter. Foreign keys marked "X" may refer to any component with an SCTID primary key.

Table 3 – Phrase Search Tables

Table and field names	Type	Size		Partition	Indices				Use of recommended additional indices
		Integer bits	String length		Primary	Recommended		Foreign	
ExcludedWords	Table	-	-	-					
LanguageCode	String	-	8	-	1				
Keyword	String	-	8	-	2				
DescWordKey	Table	-	-	-					
Keyword	String	-	8	-	1				
DescriptionId	SCTID	64	18	1	2			D	
ConcWordKey	Table	-	-	-					
Keyword	String	-	8	-	1				
ConceptId	SCTID	64	18	0	2			C	
DescDualKey	Table	-	-	-					
Dualkey	String	-	8	-	1				
DescriptionId	SCTID	64	18	1	2			D	
ConcDualKey	Table	-	-	-					
Dualkey	String	-	8	-	1				
ConceptId	SCTID	64	18	0	2			C	
Word Equivalents	Table	-	-	-		L			
WordBlockNumber	Integer	32	10	-	1				**L:** To access all instances of a particular word or phrase to allow the relevant blocks to be identified.
WordText	String	-	8	-	2	1			
WordType	Enum	8	2	-		2			
WordRole	Enum	8	2	-					

Table 4 – Enumerated Values in Word Equivalents Table

Table and Field	Value	Name	Description
Word Equivalents WordType	0	unspecified	The default value
	1	word form variant	e.g. “abdomen”, “abdominal”
	2	word equivalents	e.g. “renal”, “kidney”
	3	abbreviation or acronym	e.g. “MI” → “myocardial infarction”
	4	equivalent phrase	e.g. “heart attack”→ “myocardial infarction”
Word Equivalents WordRole	0	unspecified	The default value
	1	general qualifier	e.g. “mild”, “severe”, “emergency”, etc.
	2	topography	e.g. “arm”, “abdomen”, “abdominal”, etc.
	3	topography qualifier	e.g. “left”, “upper”, etc.
	4	object	Including organism or substance
	5	action	e.g. “remove”, “removal” “excision”, etc.
	6	unit of measure	e.g. “ml”, “oz”, etc.

Appendix E. SNOMED Index Generator

E.1 Index Generator Installation and Processing

E.1.1 Installation

These programs require Java SDK to run. If it is not already installed, it can be downloaded at no charge at java.sun.com. Version 1.3.1 or later of JDK should be used.

These programs were tested in a Microsoft Windows 2000 environment.

Unzip sct_index_generator_20040731.zip. When extracting with WinZip, be sure to select the "Use Folder Names" option. In some environments, it is best to install directly on a drive such as the C Drive, rather than to install on a directory such as My Documents that may be treated in a special fashion by the operating system.

Refer to readme.txt for any additional information.

E.1.2 Processing

The Index Generator requires two input files:

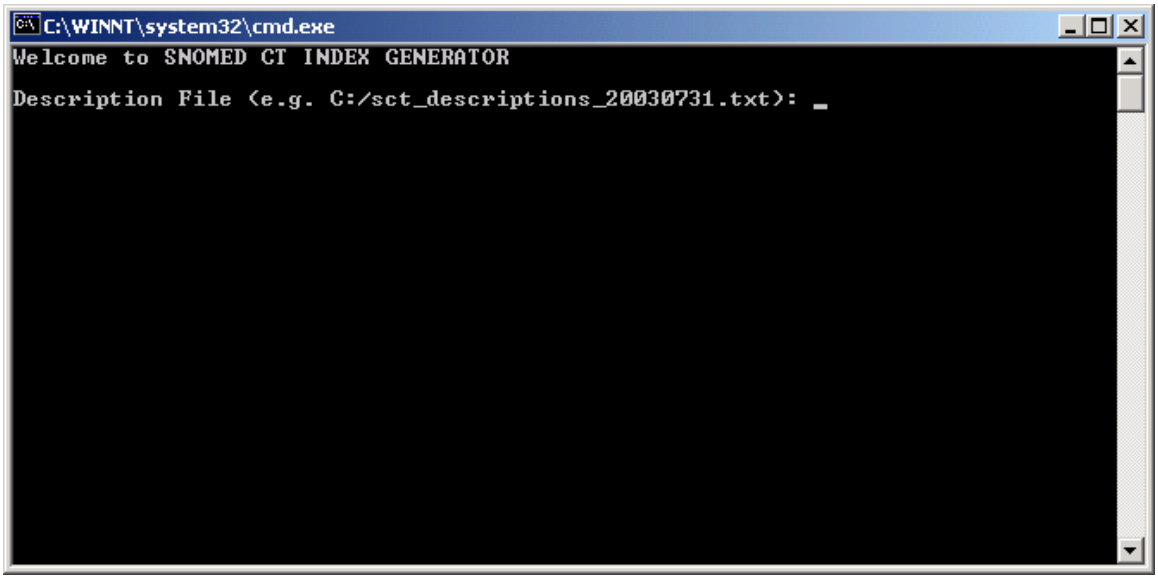
- A Descriptions File in standard SNOMED CT format. See SNOMED CT Technical Reference Guide for details about the format.
- An Excluded Words Table. A default file is provided with the Index Generator in the /CONF subdirectory. This table may be modified. See elsewhere in this Developer Toolkit documentation for the format of this file.

The Index Generator then creates the single and dual wordkey indexes in a designated directory. These files are described elsewhere in this Developer Toolkit documentation.

The Index Generator uses UTF-8 file formats since the Description Table contains special characters, even in English, usually due to the names of individuals.

The Index Generator is designed to run either using a graphical user interface (GUI) or a command line interface. A GUI option may be provided in a future release.

Command Line format: To use the command line format, under INSTALL-DIR execute run.bat. A window similar to the following will appear. It allows entry of the location of the input and output files by typing in the exact location of those files.



E.1.3 Directory Structure

INSTALL-DIR, the root directory of the Index Generator, contains one application file and readme.txt. Use run.bat to start the application in command-line mode.

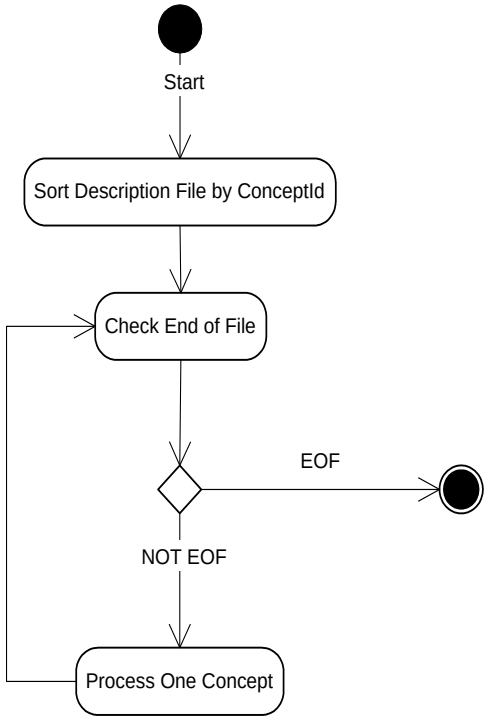
Subdirectories:

INSTALL-DIR/conf:	Configuration files used by the Index Generator
INSTALL-DIR>/lib:	Library jar files
INSTALL-DIR>/output	Used to hold temporary files generated during the process

E.2 Program Architecture and Design

The general processing is shown in Figure 1.3.1. First, the input Description File is sorted by ConceptId so that descriptions having the same ConceptId are next to each other. Then each concept is processed one by one until the end of the Description File is reached.

Figure 1.3.1



For each concept, the process is shown in Figure 1.3.2.

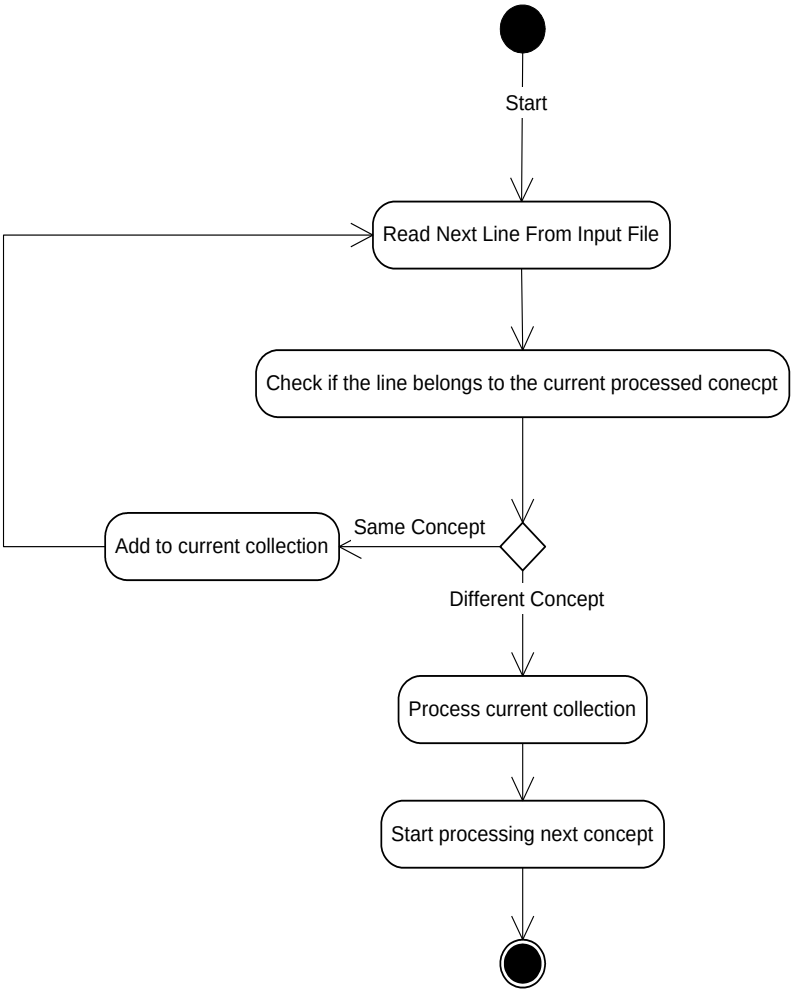


Figure 1.3.2

Performance was the primary concern since the Description File could easily contain millions of clinical terms. With this design, only lines related to the same concept are ever loaded into the memory together at once for processing. So the memory requirement is dramatically reduced. Since the input file is presorted before processing, only one sequential reading is needed for the entire input file for the purpose of generating all the index files. It saves processing time while eliminating the needs of intermediate external storage.

The data flow is illustrated in Figure 1.3.3. The description file is encoded in UTF-8 format. The reader in this system fetches bytes from the input stream and converts them to the Unicode format, which is the internal representation of strings in Java. When the system is ready to write strings out to the output index file, the writer takes the entire string and converts them into UTF-8 format. Then it sends them out to the output stream. The readers and writers are buffered for performance reason.

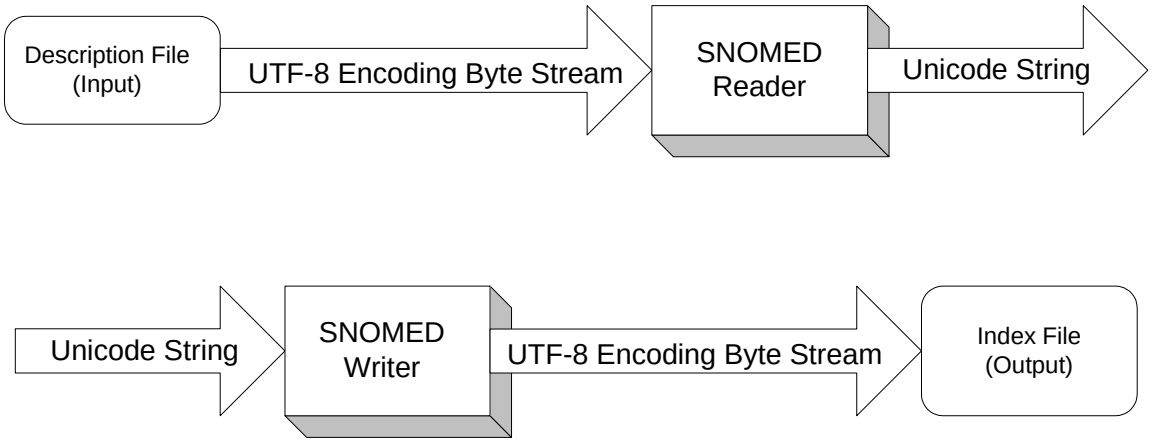
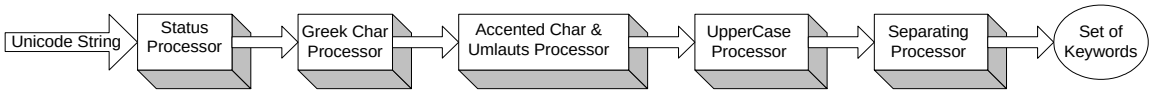


Figure 1.3.3

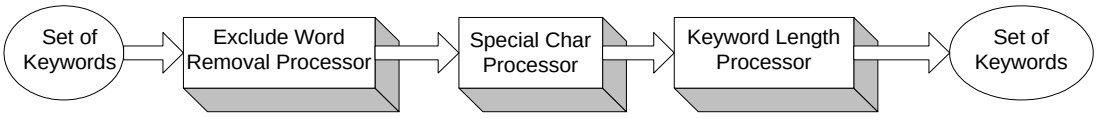
For each line the system reads in, it strips the term string out, then sends it through a series of processors to obtain the source text to work on, as shown in Figure 1.3.4. The end result is a distinct set of candidate keywords.

Figure 1.3.4



This set of candidate keywords goes through further processing resulting in the final set of single keywords generated for one description as shown in Figure 1.3.5.

Figure 1.3.5



The keywords for a concept are processed similarly, except that the original text for the concept is obtained by concatenating together the terms having the same ConceptId.

E.3 Subsystems

The system consists of three components, GUI Interface component, Command-Line Interface Component, and Engine Component, as shown in Figure 2.1.

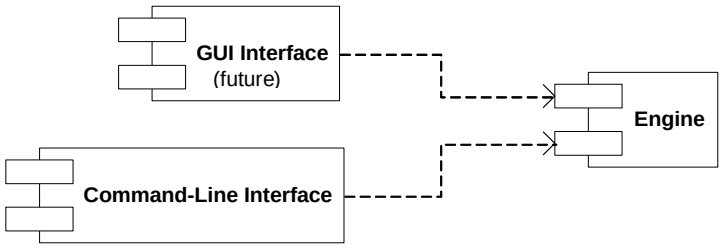


Figure 2.1

The Engine component contains the implementation of the generation algorithm and processing logic, as described in Chapter 1. It provides a universal interface (*org.cap.snomed.CTIndexGeneratorService)* for different clients to invoke its service.

E.3.1 SNOMED CT Index Generation Engine

The main classes are shown in Figure 2.1.1. Class *SimpleFileCTIndexGeneratorServiceImpl* implemented the interface *CTIndexGeneratorService* and plays central role here. It uses *CTFileSorter* class for sorting both the input and output file. It implemented the Decorator pattern to chain the processors together (as illustrated in Figure 1.3.4 and 1.3.5).

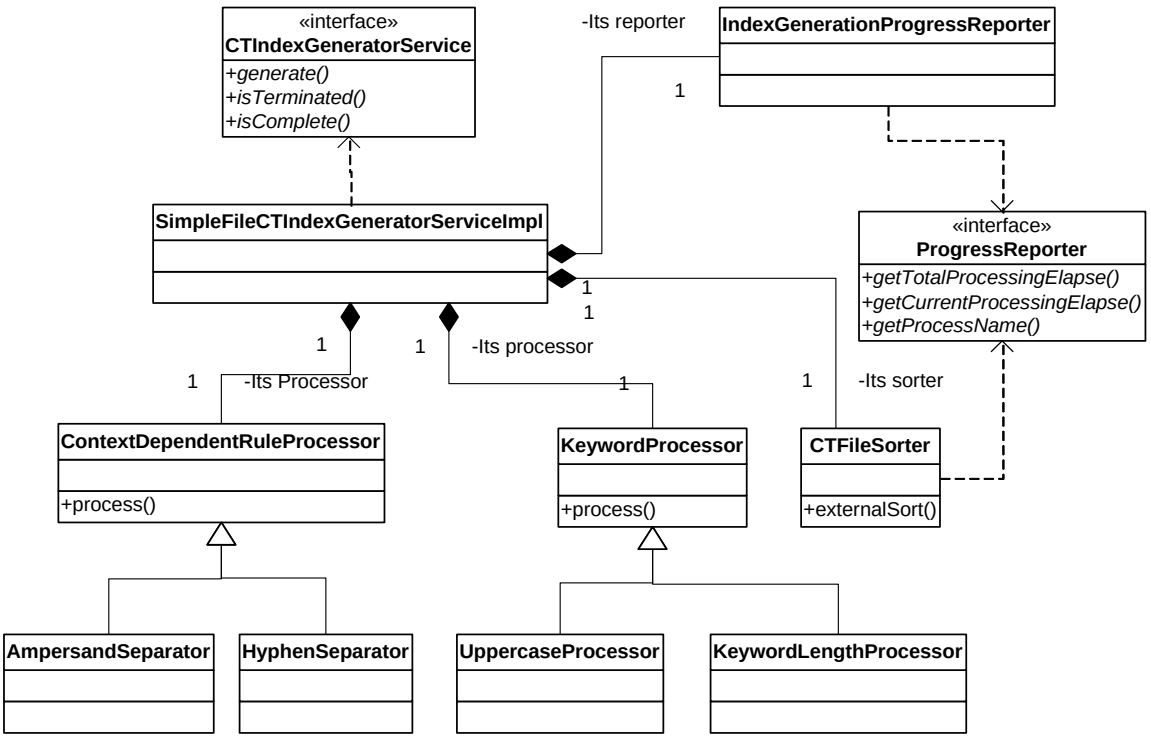


Figure 2.1.1

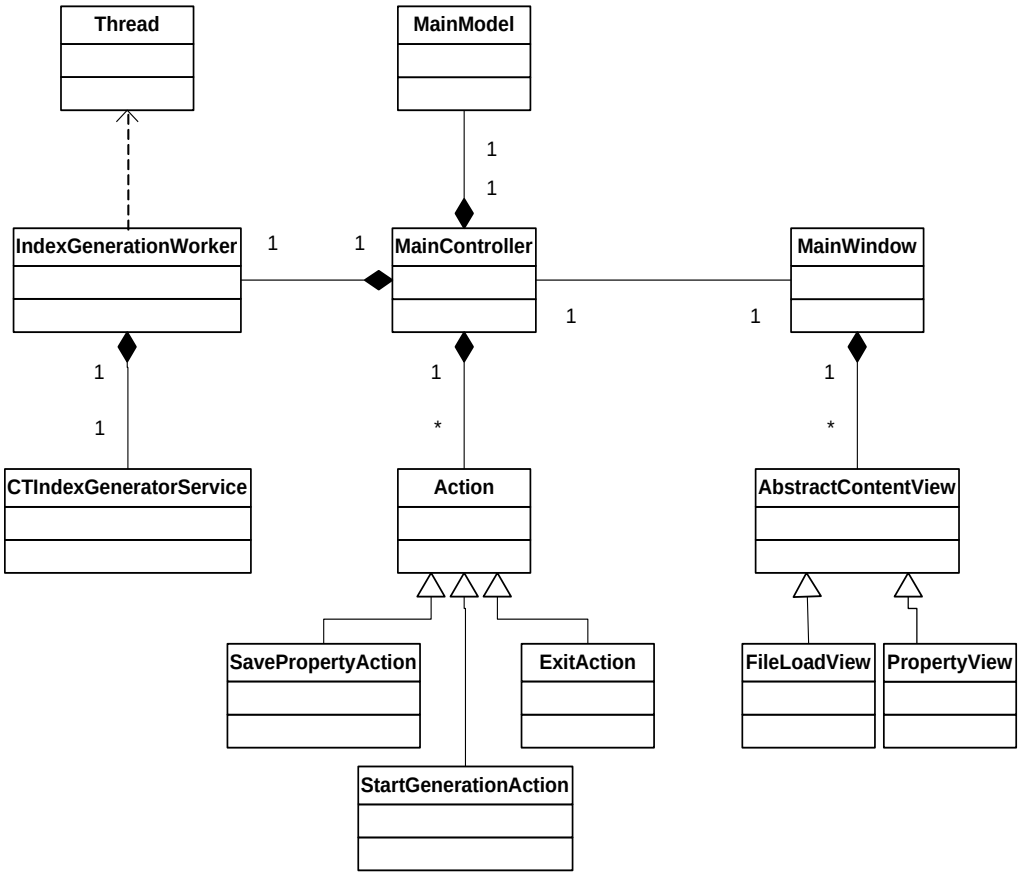
Index generation process usually takes a long time so the process is best to run on a separate thread. To provide access to allow other threads to interact with it, the *CTFileSorter* and *IndexGenerationProgressReporter* class implemented the interface *ProgressReporter*, so they could be polled for the progress or be interrupted.

E.3.2 SNOMED CT Index Generation GUI Interface (future)

The graphical user interface subsystem uses the classic Model-View-Controller design pattern. Only one controller (*MainController* class) and one model (*MainModel* class) are used here since the system was fairly simple and straightforward. The view (*MainWindow* class) consists of menu bar, tool bar, and a content area. The content area is essentially a container to host instances of *AbstractContentView* class.

Currently there are two subclasses of *AbstractContentView* class: *FileLoadView* for loading description file and starting generation process; and *PropertyView* for defining related properties. Only one of them is visible at a time, while the other is hidden. The *MainController* class plays central role here and is responsible for the screen navigation, starting/stopping generation service thread, and updating models. It keeps an instance of *IndexGenerationWorker* class, which is a subclass of *java.lang.thread*, to start/stop the actual index generation process. Since the generation process was running on a separate thread, the controller is able to poll for the progress periodically and update the GUI widgets (progress bar, status line, etc.) independently on a different thread.

Figure 2.2.1



E.3.3 SNOMED CT Index Generation Command-Line Interface

The command-line interface subsystem consists of one main class, *CTIndexGenerator*. Similar to GUI, it runs the index generation process on a separate thread, an instance of Ind*exGenerationWorker* class. The main thread is able to poll for the progress periodically and report it.

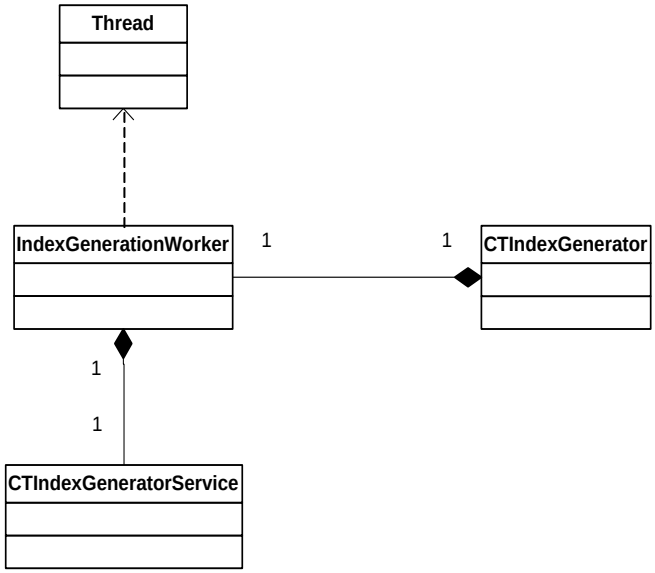


Figure 2.3.1