SNOMED Clinical Terms®
Canonical Table Guide

January 2008 International Release

Table of Contents

Document History

Version	Notes
January 2002	• First Publication based on approved specifications
July 2002	• Date change
January 2003	• Moved to separate Developer Toolkit documentation
July 2003	• Moved to separate Canonical Table Guide
January 2004	• Date changes and convert to Version numbering for documentation
July 2004	• Correct typo • Added Document History
January 2005	• Updated Copyright. • Updated contacts.
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	•

Inventory of Documentation

The following 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 Reference Guide (TRG)

The TRG is intended for SNOMED CT implementers, such as software developers. The TRG assumes an information technology background. Clinical knowledge is not a prerequisite.

The TRG 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. It can be used to install and use SNOMED.

SNOMED CT Technical Implementation Guide (TIG)

The TIG is intended for SNOMED CT implementers, such as software 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 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 documentation is also included with the International Release of SNOMED CT:

- SNOMED CT Namespace Identifier Guide
- SNOMED CT Developer Toolkit Guide

1 Introduction

1.1 Purpose

This document describes the existing file structure of SNOMED Clinical Terms® as accepted by the International Health Terminology Standards Development Organization (IHTSDO) and released.

- 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 Canonical Table. This document is to provide a reference about the SNOMED CT technical structure of the Canonical Table 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
Rued Langgaards Vej 7, 5te
DK-2300 Copenhagen S
Denmark

Tel: +45 36 44 87 36

Fax: +45 44 44 87 36

2 How the Short Canonical Form is Derived

2.1 Introduction

This section is based on the process for generating the short canonical form used to generate the "canonical table" included in this release. This version of the short canonical form, while still useful, is no longer regarded as the best form for use in testing subsumption of pre and post-coordinated expressions.

Two draft documents covering this topic in greater detail are available from the IHTSDO. The first of these discusses the various abstract models and representational forms that may be applicable to SNOMED CT, and the second provides detailed advice on transformation to common normal forms in which alternative representations can be readily compared.

These additional documents are currently in draft form for discussion within the IHTSDO. However, they are recommended to implementers seeking a more complete, detailed and up to date understanding of alternative forms, transformations and comparisons.

This section describes the process by which the short canonical form is derived from the Relationships and Concepts that are included in a SNOMED CT release. This process is documented to enable developers using this data to understand its relationship with data released in the core tables.

2.2 Overview

Before the SNOMED CT core tables are released, they are processed using a description logic classifier. Description logic classification ensures that the Relationships released are logically consistent. This is an essential precursor to generation of the short canonical form.

Working from this logically consistent baseline, each Concept is tested as follows:

- Identify its proximal Primitive supertype Concept(s)
  - For each of these add an "IS A" subtype Relationship to the Canonical Table
- Identify its unshared defining characteristics
  - For each of these add an appropriate Relationship to the Canonical Table

The Concept being tested is referred to in the following more detailed descriptions as the FocusConcept.

2.3 Description Logic Classification

Description logic classification is undertaken prior to each release of SNOMED CT to produce a logically consistent set of Relationships. This process ensures that:

- The set of "IS A" subtype Relationships:
  - Is complete based on the current stated defining characteristics
    - Additional inferred "IS A" Relationships are automatically added where necessary
  - Contains no cyclical definitions
    - Cyclical definitions, such as $\boldsymbol{A}$ is a $\boldsymbol{B}$ + $\boldsymbol{B}$ is a $\boldsymbol{C}$ + $\boldsymbol{C}$ is a $\boldsymbol{A}$ are automatically detected and manually resolved
  - Refer only to most proximate supertypes
    - Stated Relationship sets such as $\boldsymbol{A}$ is a $\boldsymbol{B}$ + $\boldsymbol{B}$ is a $\boldsymbol{C}$ + $\boldsymbol{A}$ is a $\boldsymbol{C}$ are rationalized by excluding the redundant distant definition $\boldsymbol{A}$ is a $\boldsymbol{C}$
- Descendant subtypes inherit appropriate defining characteristics
  - If it is stated that $\boldsymbol{A}$ is a $\boldsymbol{B}$ + $\boldsymbol{B}$ has property $\boldsymbol{C}$, it must also be true that either:
    - $\boldsymbol{A}$ has property $\boldsymbol{C}$

    or

    - $\boldsymbol{A}$ has property $\boldsymbol{D}$ and $\boldsymbol{D}$ is a $\boldsymbol{C}$

  If neither of these is stated, the inferred Relationship $\boldsymbol{A}$ has property $\boldsymbol{C}$ is added

2.4 Identifying Proximal Primitive Supertypes

1. Select all rows in the Relationships Table where:
   - ConceptId1 refers to the FocusConcept AND
   - RelationshipType refers to the “is a” Concept
2. For each of the Reference Concepts pointed to by ConceptId2 in each of the selected rows
   - If IsPrimitive =0:
     - Select all rows in Relationships Table where
       - ConceptId1 refers to the ReferencedConcept AND
       - RelationshipType refers to the “is a” Concept
     - For each of these selected rows repeat step 2 (recursive)
   - If IsPrimitive =1:
     - Add ReferenceConcept to the list of PrimitiveSupertypes
3. If in recursion as a result of (2b):
   - Exit that level of recursion and continue processing the rows selected at previous level
4. For each Concept in the list of PrimitiveSupertypes
   - If any of the other PrimitiveSupertypes have this Concept as a PrimitiveSupertype
     - Remove the Concept from the list of PrimitiveSupertypes
5. For each Concept in the list of PrimitiveSupertypes
   - Create a Relationship in the Canonical Table in which:
     - ConceptId1 refers to the FocusConcept
     - RelationshipType refers to the “is a” Concept
     - ConceptId2 refers to the PrimitiveSupertype
     - Relationship group = 0

2.5 Identifying Unshared Defining Characteristics

1. Select all rows in the Relationships Table where:
   - ❖ ConceptId1 refers to the FocusConcept AND
   - ❖ RelationshipType does not refer to the “is a” Concept AND
   - ❖ CharacteristicType = 0 (Defining characteristic)
2. Mark each of the selected rows as a provisional row for the Canonical Table
3. For each Concept in the list of PrimitiveSupertypes:
   - ❖ Select all rows in the Relationships Table where:
     - ✧ ConceptId1 refers to the PrimitiveSupertype AND
     - ✧ RelationshipType does not refer to the “is a” Concept AND
     - ✧ CharacteristicType = 0 (Defining characteristic)
   - ❖ For each selected row:
     - ✧ Unmark any provisional row that matches the selected row on the following attributes
       - ▪ RelationshipType AND ConceptId2
4. Add each remaining marked provisional row to the Canonical Table
   - ❖ The fields are populated with values of the corresponding fields in the Relationships Table

Appendix A. Canonical Table

Canonical Table

Each row in the Canonical Table represents a Relationship between two Concepts

Only relationships which are required to represent the short canonical form are included in this table. The algorithm by which the short canonical form is derived from data in the Relationships Table and Concepts Table is detailed below.

Canonical Table Key Fields	Field Type	Permitted characters	Length
ConceptId1	**SCTID**	*Digits* ***0*** *to* ***9*** *only*	*6 to 18*
The unique SNOMED Clinical Terms identifier of the Concept which is defined by this Relationship.			
RelationshipType	**SCTID**	*Digits* ***0*** *to* ***9*** *only*	*6 to 18*
The unique SNOMED Clinical Terms identifier of the Concept which represents the type of relationship between the related Concepts.			
ConceptId2	**SCTID**	*Digits* ***0*** *to* ***9*** *only*	*6 to 18*
The unique SNOMED Clinical Terms identifier of the Concept which is the target of this Relationship.			
RelationshipGroup	*Integer*	***0 to 9***	***1 to 2***
An integer value that expresses an association between two or more Relationships. The default Relationship group value is zero and this applies to all Relationships that have not been stated to be associated with any other Relationships. All Relationships that share the same ConceptId1 and the same non-zero Relationship group value are associated with one another. Any Relationships that share the same ConceptId1 but have different Relationship group values are not associated with one another. See example below:			

ConceptId1	RelationshipType	ConceptId2	RoleGroup
Removal of calculus of urinary bladder	Direct morphology	Calculus	0
Removal of calculus of urinary bladder	Method	Removal	1
Removal of calculus of urinary bladder	Procedure site - Indirect	Urinary bladder	1

Appendix B. Illustrations of the Short Canonical Form

This example describes the differences between a full set of Relationships (Figure 1) and the short canonical form (Figure 2). In Figure 2:

- ❖ Each box in these diagrams represent a Concept
  - ✧ The text in the lower part of the box represents the defining characteristics of the Concept
  - ✧ Shaded boxes containing the word "Primitive" indicate that a Concept is Primitive – that is, the concept is not Fully Defined
- ❖ The arrows represent "IS A" Relationships
  - ✧ Bold arrows represent added Relationships resulting from generation of the short Canonical form

Simple computation based on the canonical form

The short canonical form reduces complexity and duplication in the defining characteristics without losing any of the information embedded in the definition. Thus it provides a minimal form of representation which can be used for comparing two Concepts.

Table 3 compares the definitions of "Aluminum pedal bike" in the full and short Canonical forms and illustrates that they are derivable from one another without any loss of information. It is a useful example to explain the canonical form.

Table 1 – Mapping between full and short-canonical forms

Full Form	***Derivation from Short Canonical Form***
Is a = Pedal bike	Derivable from "Is a = Bike" + "Powered-by = Pedals"
Is a = Aluminum machine	Derivable from "Is a = Metal machine" + "Substance = Aluminum"
Origin = Man-made	Derivable from "Is a = Metal machine" + definition of Metal machine
Made-of = Aluminum	Stated "Substance = Aluminum"
Moves-on = Two wheels	Derivable from "Is a = Bike" + definition of "Bike"
Powered-by = Pedals	Stated "Powered-by = Pedals"
Short Canonical Form	***Mapping to Full Form***
Is a = Bike	Derivable from "Is a = Pedal bike" + definition of "Pedal bike"
Is a = Metal machine	Derivable from "Is a = Aluminum machine" + definition of "Aluminum machine"
Made-of = Aluminum	Stated "Substance = Aluminum"
Powered-by = Pedals	Stated "Powered-by = Pedals"

The significance of this is that the short canonical form is the simplest form that is sufficient to answer various questions about an "Aluminum pedal bike".

For example, it is possible to determine answers to the following:

- ❖ Is an "aluminum pedal bike" a "machine"? – YES
- ❖ Is an "aluminum pedal bike" made of "metal" – YES
- ❖ Is an "aluminum pedal bike" a "motor bike"? – NO

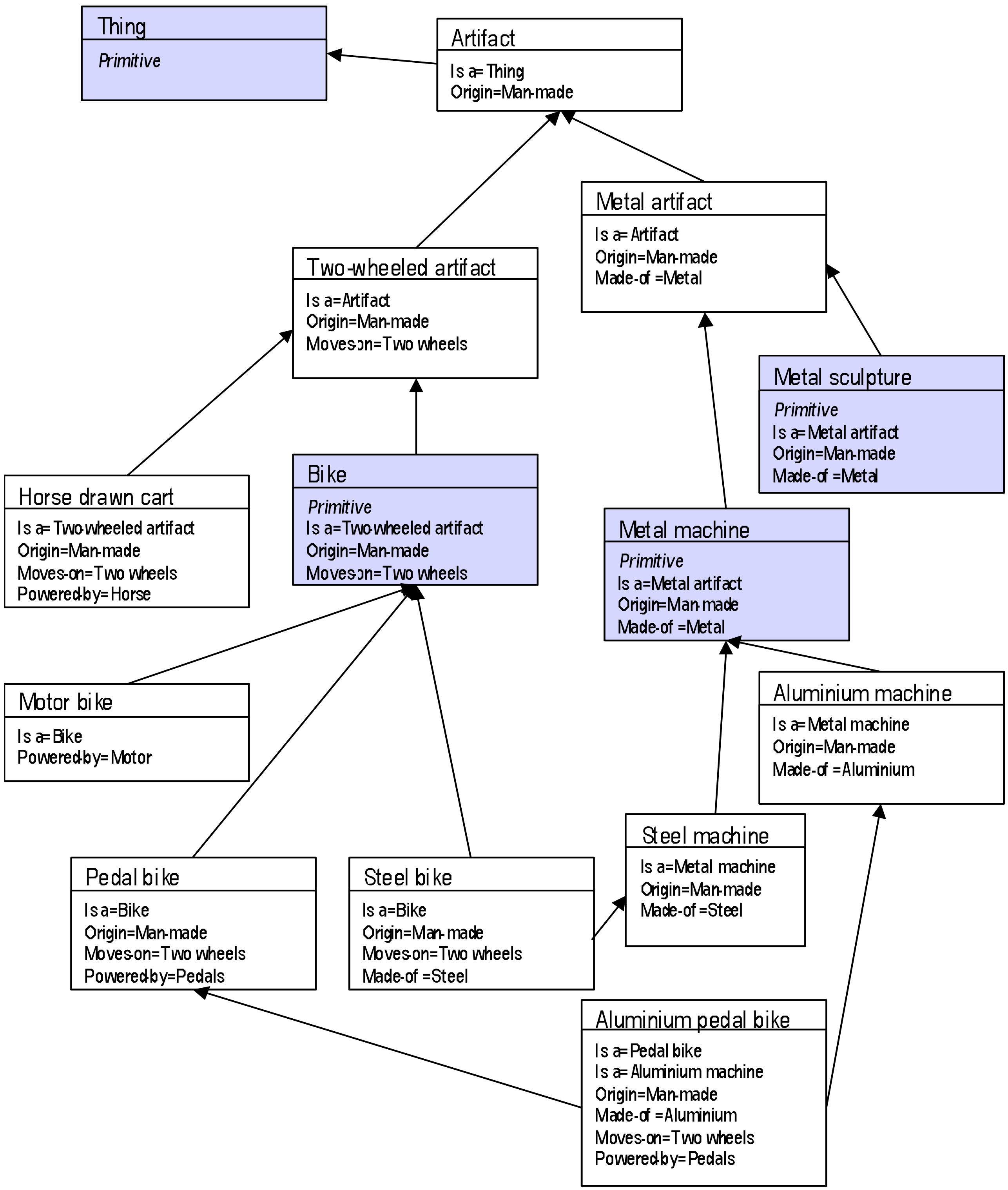

Figure 1 – Example of a set of defining relationships

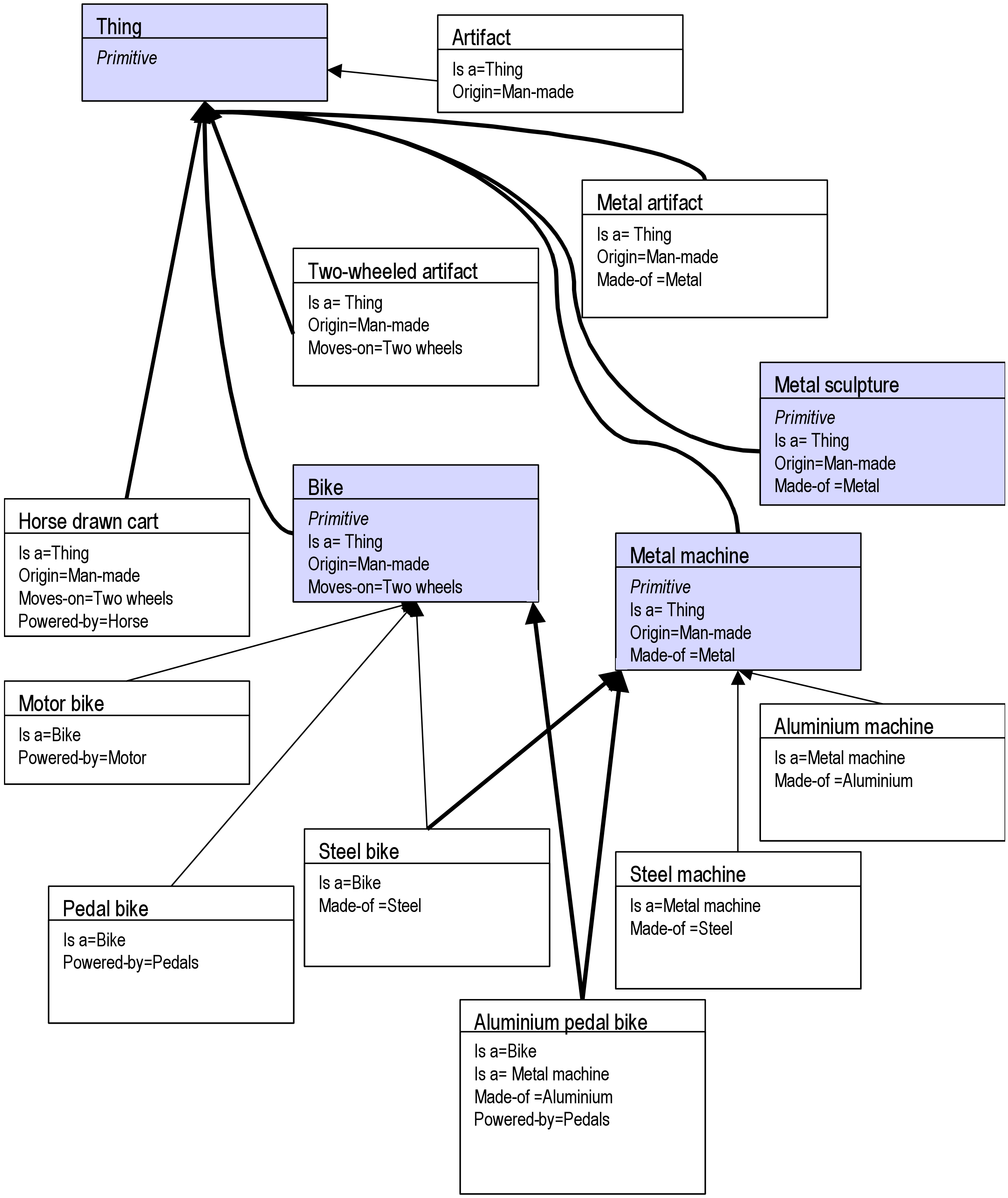

Figure 2 – Example of a short canonical form derived from Figure 1

Comparing post-coordinated representation using the short canonical form

Post-coordinated representations of Concepts can also be compared with the short canonical form. However, primitive Concepts may impose limits on this. For example, the Concept “Steel pedal bike” does not exist in these illustrations. However, it could be represented by three different post-coordinated forms:

- “Pedal bike” + “Made-of = Steel”
- “Steel bike” + “Powered-by = Pedals”
- “Steel machine” + “Moves on = Two wheels” + “Powered-by = Pedals”

If the rules for generating the short canonical form are applied these representations the results are similar but incomplete representations of the expected short canonical form.

Table 2 – Equivalence of pre and post-coordinated forms

Expected	**Mapping from post-coordinated representation**
Short Canonical Form	**“Pedal bike” + “Made-of = Steel”**
Is a = Bike	Derivable from “Is a = Pedal bike” + definition of “Pedal bike”
Is a = Metal machine	*Not derivable from this representation (see following notes)*
Made-of = Steel	Stated “Substance = Steel”
Powered-by = Pedals	Stated “Powered-by = Pedals”
Short Canonical Form	**“Steel bike” + “Powered-by = Pedals”**
Is a = Bike	Derivable from “Is a = Steel bike” + definition of “Steel bike”
Is a = Metal machine	*Not derivable from this representation (see following notes)*
Made-of = Steel	Derivable from “Is a = Steel bike” + definition of “Steel bike”
Powered-by = Pedals	Stated “Powered-by = Pedals”
Short Canonical Form	**“Steel machine” + “Moves on = Two wheels” + “Powered-by = Pedals”**
Is a = Bike	*Not derivable from this representation (see following notes)*
Is a = Metal machine	Derivable from “Is a = Steel machine” + definition of “Is a = Steel machine”
Made-of = Steel	Derivable from “Is a = Steel machine” + definition of “Is a = Steel machine”
Powered-by = Pedals	Stated “Powered-by = Pedals”
Moves-on = Two wheels	Stated “Moves-on = Two wheels”

Notes:

This lack of complete equivalence illustrates the incompleteness of the definitions rather than any inherent limitation in the method. The enhanced definitions in Figure 3, and the resulting revised canonical form shown in Figure 4, illustrate this.

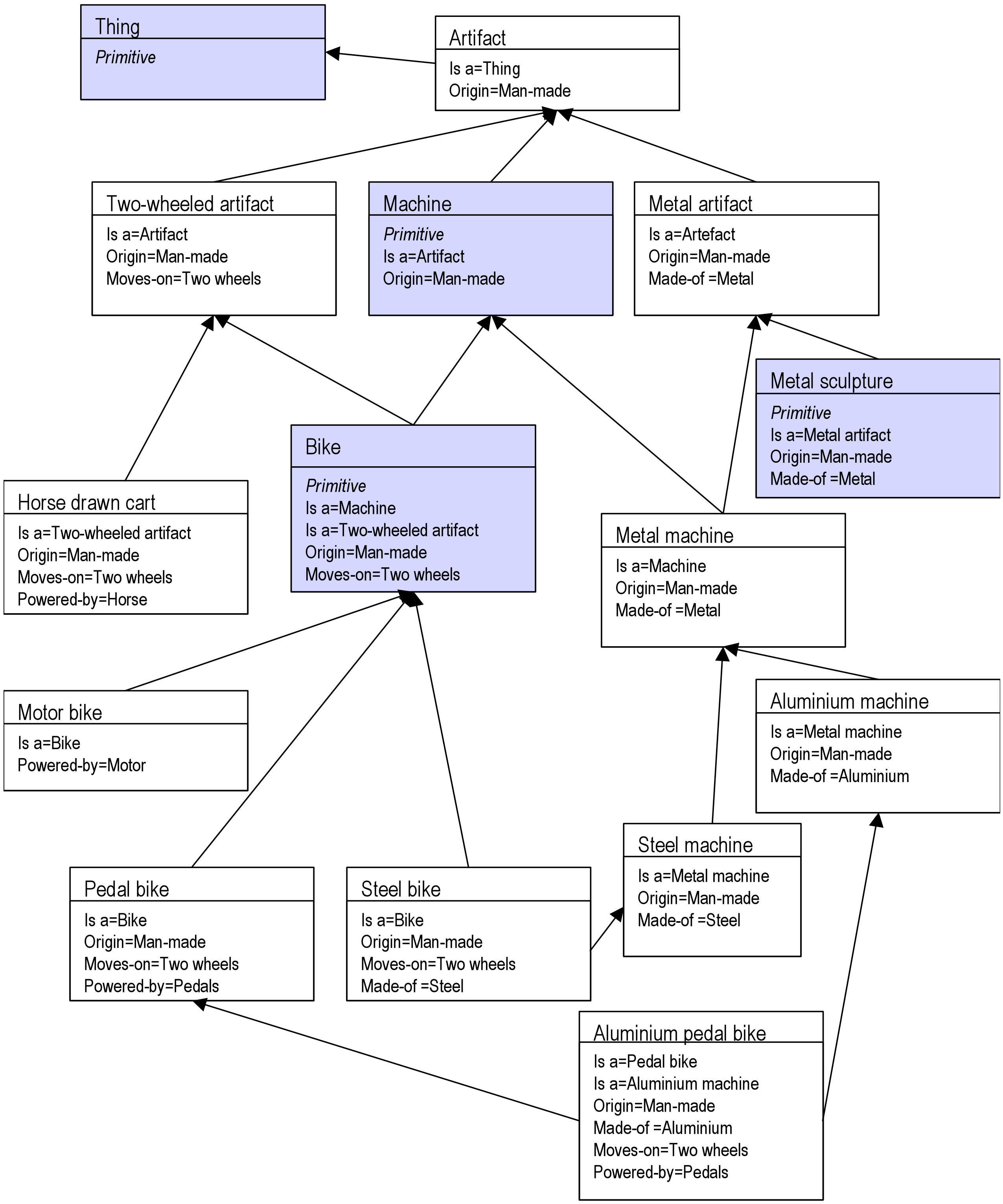

Figure 3 – Enhanced example of set of relationships after adding a new concept (Machine)

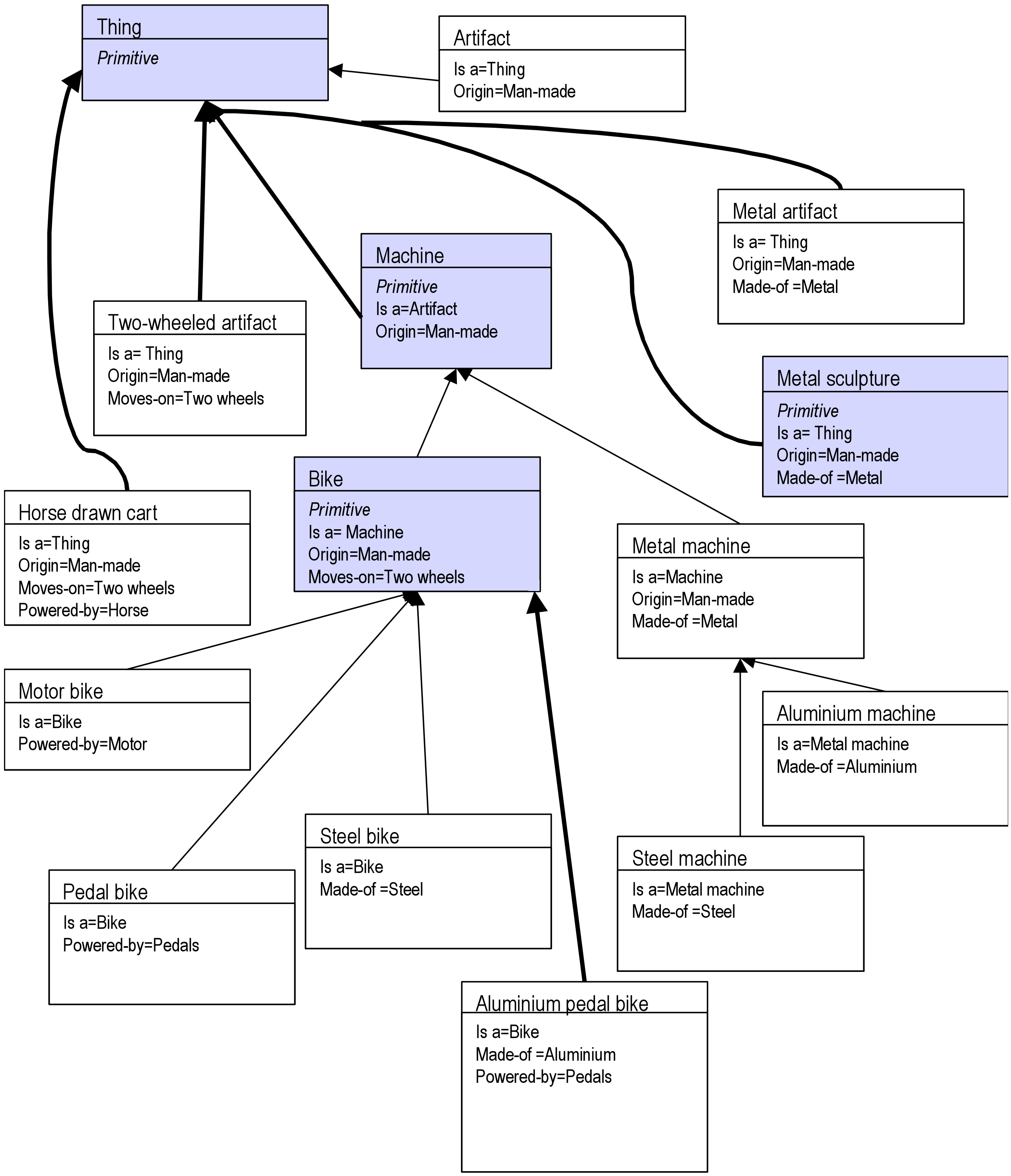

Figure 4 – Enhanced example of short canonical form after adding a new concept (Machine)

Improved comparisons with enhanced definitions

The changes between Figure 1 and Figure 3 are as follows

- ❖ Addition of a new primitive Concept “Machine”
- ❖ “Metal machine” is now Fully Defined as “is a = Machine”+ “Made-of = Metal”
- ❖ “Bike” remains primitive but has a new Relationship “is a = Machine”.

As a result, the short canonical form of “Aluminum pedal bike” as seen in Table 5 is simplified “Is a = Bike” + “Made-of = Aluminum” + “Powered-by = Pedals”. The reference to “Metal machine” in the previous version is superfluous for the following reasons:

- ❖ “Metal machine” now has a primitive supertype “Machine”
- ❖ This is also *PrimitiveSupertype* of “Bike”
- ❖ “Bike” itself is a *PrimitiveSupertype* of the “Aluminum pedal bike”
  - ✧ Thus “Machine” is not a proximal PrimitiveSupertype.

Table 3 – Revised mapping between the full and short-canonical forms

Full Form	***Derivation from Short Canonical Form***
Is a = Pedal bike	Derivable from “Is a = Bike” + “Powered-by = Pedals”
Is a = Aluminum machine	Derivable from “Is a = Bike” + “Bike – is a = Machine” + “Substance=Aluminum” + Definition of “Aluminum machine”
Origin = Man-made	Derivable from “Is a = Bike” + “Bike – is a = Machine” + Definition of “Machine”
Made-of = Aluminum	Stated “Substance = Aluminum”
Moves-on = Two wheels	Derivable from “Is a = Bike” + definition of “Bike”
Powered-by = Pedals	Stated “Powered-by = Pedals”
Short Canonical Form	***Mapping to Full Form***
Is a = Bike	Derivable from “Is a = Pedal bike” + definition of “Pedal bike”
Made-of = Aluminum	Stated “Substance = Aluminum”
Powered-by = Pedals	Stated “Powered-by = Pedals”

Two of the post-coordinated representations discussed earlier now create canonical forms that are identical with the pre-coordinated representation.

- ❖ “Pedal bike” + “Made-of = Steel”
  - ✧ “Is a = Bike” + “Made-of = Steel” + “Powered-by = Pedals”
- ❖ “Steel bike” + “Powered-by = Pedals”
  - ✧ “Is a = Bike” + “Made-of = Steel” + “Powered-by = Pedals”

Residual limitations relating to primitive concepts

The third post-coordinated representation still produces a different result

- ❖ “Steel machine” + “Moves on = Two wheels” + “Powered-by = Pedals”
  - ✧ “Is a = Machine” + “Made-of = Steel” + “Powered-by = Pedals” + “Moves on = Two wheels”

The difficulty is that “Bike” is stated to be primitive. This correctly recognizes that a machine, which moves on two wheels, may be something other than a “Bike”. Thus this information provided is insufficient to specify that this is a “Bike”.

An examination of the possibilities might identify “Pedal bike” as the only Concept in the thesaurus that “Is a = Machine” + “Powered-by = Pedals” + “Moves on = Two wheels”. However, even this would not prove that this post-coordinated representation did not refer to some other type of device.

In cases such as this, generating the canonical form correctly identifies the inherent ambiguity in some superficially similar post-coordinated representations.

Appendix C. 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.
- ❖ A 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 4 – Canonical Table

Table and field names	Type	Size		Partition	Indices				Use of recommended additional indices
		Integer bits	String length		Primary	Recommended		Foreign	
Canonical	Table	-	-	-		**M**	**N**		
ConceptId1	SCTID	64	18	0	1	1		C	**M:** To access all or selected canonical Relationships of a specified source Concept. **N:** To access all or selected canonical Relationships to a specified target Concept.
RelationshipType	SCTID	64	18	0	2	3	2	C	
ConceptId2	SCTID	64	18	0	3		1	C	
RelationshipGroup	Integer	16	5	-	4	2			