nehta

Technical Implementation Guide

Australian Medicines Terminology (Version 3 Model)

26 February 2013

Approved for Release

National E-Health Transition Authority

National E-Health Transition Authority Ltd
Level 25
56 Pitt Street
Sydney, NSW, 2000
Australia.
www.nehta.gov.au

Document information

Key information

Owner:	National Clinical Terminology and Information Service, NEHTA
Filename:	AMTv3model_TechnicalImplementationGuide.docx Last saved: 26/02/2013 5:11 PM
Review date:	5 August 2013
Contact for enquiries:	Dion McMurtrie, Technical Architect – Clinical Terminology and Information t: 07 3023 8449 e: dion.mcmurtrie@nehta.gov.au

Approvals

This document has been approved on the basis that the appropriate input has been obtained during its development.

Name	Position	Date
Elizabeth Donohoo	Clinical Terminology Manager	26 February 2013
Philip Wilford	Manager Reference Platform and Vendor Integration Support	

Quality reviews

Revision	Version	Reviewer(s)	Type of review	Purpose
	002	Various stakeholders/SMEs	Peer	To ensure adequate coverage and accuracy of the topics covered.
	003	Various stakeholders/SMEs	Peer	To ensure adequate coverage and accuracy of the topics covered.
	004	Technical writer	Editorial	To ensure overall clarity and consistency with NCTIS editorial standards.

This page is intentionally blank.

Table of contents

This page is intentionally blank.

1 Introduction

1.1 Purpose

This document's purpose is to provide implementation guidance, for software developers and technical consumers, that eases implementation of AMT v3 for common use cases.

In order to do this, this document provides:

- an overview of the AMT;
- an overview of characteristics of SNOMED CT[1] terminologies generally (of which the AMT is one);
- details of the AMT v3 model;
- details of the AMT v3 distribution form;
- guidance on using and interpreting AMT v3; and
- sample methods of querying AMT v3 data relevant to common use cases.

1.2 Intended audience

This document is intended for Technical Healthcare Vendors and Healthcare professionals. This includes developers and testers who are responsible for producing, assuring or maintaining products that integrate with AMT v3.

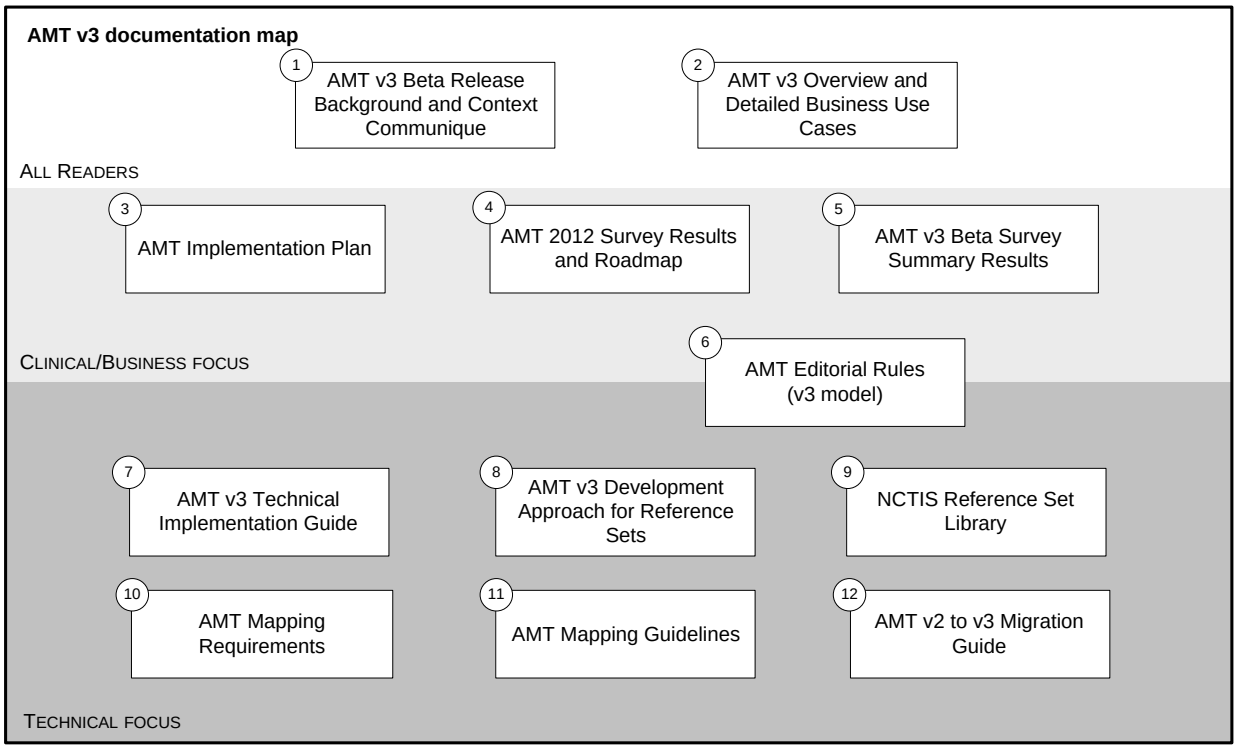
1.2.1 Documentation map

Business: Business owners, product managers, project managers, policy makers.

Clinical: Healthcare professionals and other end users.

Technical: Programmers, content developers, testers, information system suppliers, analysts, terminology/classification specialists, health IT professionals and researchers.

[1] This material includes SNOMED Clinical Terms® (SNOMED CT®) which is used by the permission of the International Health Terminology Standards Development Organisation (IHTSDO®). All rights reserved. SNOMED CT was originally created by The College of American Pathologists. IHTSDO®, SNOMED® and SNOMED CT® are registered trademarks of the IHTSDO.



Recommended reading lists for the different types of readers are as follows. Items with asterisks need only be read if relevant to the reader's needs.

Doc No.	Doc Name	Business	Clinical	Technical
1	*AMT v3 Beta Release Background and Context Communique*	Y	Y	Y
2	*AMT v3 Overview and Detailed Business Use Cases*	Y	Y	Y
3	*AMT Implementation Plan*	Y	Y	
4	*AMT 2012 Survey Results and Roadmap*	Y	Y	Y
5	*AMT v3 Beta Survey Summary Results*[2]	Y	Y	Y
6	*AMT Editorial Rules (v3 model)*		Y*	Y*
7	*AMT v3 Technical Implementation Guide*			Y
8	*AMT v3 Development Approach for Reference Sets*			Y*
9	*NCTIS Reference Set Library*			Y*
10	*AMT Mapping Requirements*			Y*
11	*AMT Mapping Guidelines*			Y*
12	*AMT v2 to v3 Migration Guide*			Y*

The prerequisites for each document are described in their respective introductions. However, the document numbers cited here give a rough indication as to the recommended sequence of reading.

[2] The *AMT v3 Beta Survey Summary Results* will be made available upon completion of feedback gathering and analysis, significantly after the AMT v3 Beta release itself.

1.3 Scope

This document is limited to discussing technical aspects of the AMT v3 model, data files and implementation considerations.

It does not cover a business overview of the AMT v3 product or drivers for the AMT product in general. These topics are covered by the *AMT v3 Overview and Use Cases* [1].

1.4 How to use this document

This is a long document containing varying detailed levels of information. However it is acknowledged that it is not necessary to know all of the content of this document in order to usefully work with AMT v3.

Please use the following table to determine the sections of this document most relevant to your needs.

Section	Description	Target audience
2.1 AMT Overview	Provides an overview of AMT the product, its purpose and uses.	Those new to AMT.
2.2 SNOMED CT terminologies	Provides a foundation in SNOMED CT as a basis for understanding the AMT v3 model.	Those with little knowledge of SNOMED CT seeking to understand the AMT v3 model.
2.3 AMT v3 model	Details of the elements and construction of the AMT v3 model.	Those wishing to work with AMT v3 product data.
3.1 Overview of Release Format 2 (RF2)	Provides an overview of SNOMED CT RF2 – the release format used to release AMT v3.	Those unfamiliar with RF2 who will work directly with AMT v3 release files.
3.2 Concrete domains and data type properties	Provides detail of the concrete domain reference sets used in AMT v3 to express datatype properties. These reference sets are an extension to SNOMED CT RF2.	Those unfamiliar with concrete domains who wish to work with the description logic underpinning the AMT v3 model, or simply access the numeric data properties associated with AMT v3 components.
3.3 RF2 distribution types	Describes the 3 release forms RF2 releases can provide – Full, Snapshot and Delta.	Those unfamiliar with these 3 release forms planning on downloading and processing AMT v3 release files regularly.
3.4 Distribution Normal and Stated Forms	Describes the differences between the Distribution Normal Form and Stated Form of SNOMED CT terminologies and how this may affect AMT v3 distributed relationships in future.	Those planning on writing code or processes which depend upon the relationships in AMT v3, in order to understand this area of future change for AMT v3 beyond the Beta data.

Section	Description	Target audience
3.5 Transitive Closure	Describes what a transitive closure is and how it may be used when querying AMT v3 product data or reporting.	Those planning on writing code or queries to manipulate, transform or report on AMT v3 product data.
3.6 Modules	Details the modules used in the AMT v3 RF2 release files and their meaning.	Those planning on processing AMT v3 release files regularly, particularly if being used in combination with SNOMED CT-AU or SNOMED CT International.
3.7 History Tracking Mechanism	Describes the history tracking mechanism used in SNOMED CT and how it is rendered in RF2 and hence AMT v3.	Those planning on taking regular updates of AMT v3 data.
4.1 Mapping	Describes a mapping based approach to implementing AMT v3.	Those considering a mapping based implementation of AMT v3.
4.2 Native implementation	Describes options for implementing AMT v3 natively in an implementation.	Those considering a native implementation of AMT v3.
4.3 Designing for data updates	Describes principles behind designing implementations of terminology and specifically AMT v3 for regular updates.	Those planning on taking regular updates of AMT v3 data.
4.4 Searching and capturing input	Describes key concepts and considerations when designing and implementing a system capturing terminology encoded data input.	Those planning on using AMT v3 in a clinical user interface for data entry.
4.5 Recording and rendering recorded data	Concepts and considerations when recording AMT v3 encoded clinical records and rendering those records back to clinical users.	Those planning on using AMT v3 to record clinical data.
4.6 Retrieval and analytics	Describes ways to use AMT v3 product data to retrieve and report on AMT v3 encoded records.	Those needing to report on or analyse AMT v3 encoded records.
5 Sample code and scripts	Provides sample schema, queries and scripts to educate about the AMT v3 model and data through tangible demonstration queries.	Those needing to work intensively with AMT v3 product data as release in the RF2 release files.
6 Testing and Conformance	Provides information on testing and conformance considerations for implementations.	Those planning on implementing AMT v3 in a system.

Section	Description	Target audience
7 Implementation considerations	Provides AMT v3 product information and implementation related considerations useful to consider when planning an implementation.	Those planning an implementation of AMT v3.

1.5 Sample scripts

Please note that the sample scripts provided in this document and associated release materials are provided solely as aids to assist in understanding AMT v3 in more depth. These scripts should not be construed as suggested bases for implementation.

1.6 Questions and feedback

NCTIS's product development relies on the input and co-operation of the healthcare community. We value your feedback and encourage questions, comments or suggestions about our products.

To provide feedback, or for further information regarding licensing, please contact us via email at terminologies@nehta.gov.au.

2 AMT v3 model technical overview

2.1 AMT Overview

This section provides a very brief overview of the AMT. For a more detailed description please refer to *AMT v3 Overview and Use Cases* [1].

2.1.1 What the AMT is

The AMT is a terminology of Australian medicines aimed at supporting Australian eHealth in the broad areas of:

- Prescribing
- Recording
- Reviewing
- Issuing (including dispensing)
- Administering
- Transferring information

To enable use cases in these areas, the AMT uniquely and unambiguously identifies and describes commonly used generic and branded[3] medications in Australia. The AMT contains defining information about the products, for example ingredients, strengths and forms, as well as information about how these products relate to one another.

The AMT provides:

- Consistent identification of branded and generically equivalent medicines.[4]
- Consistent naming conventions and terminology used to describe and search for medications.

2.1.2 What the AMT is not

The AMT is not a product database. For example, the AMT does not contain non-defining information about products, such as:

- PBS pricing
- PBS and general product availability
- Poisons schedule information
- Adverse effects
- Dose checking
- Indications
- Contraindications
- Drug-drug interactions
- Cautionary advice

In order to implement systems requiring this type of additional product information, additional data sources need to be sought.

[3] Commonly referred to as 'trade' medications.

[4] Branded and equivalent generic such as |*Amoxil 500 mg capsule: hard, 1 capsule*| and |*amoxycillin 500 mg capsule*| as the equivalent generic. Note that the AMT **does not** include bioequivalence.

2.1.3 What the AMT contains

The AMT contains identifiers and descriptions for the majority of 'Registered' (AUSTR) products contained in the Australian Register of Therapeutic Goods. Many 'Listed' (AUSTL) products are also included. Some AUSTR products are available only by prescription whereas AUSTL products can be purchased over-the-counter.

These medications are represented at a pack, container/pack or unit level, for example:

Pack level	Amoxil 500 mg capsule: hard, 20 capsules
Container pack level	Amoxil 500 mg capsule: hard, 20 capsules, blister pack
Unit level	Amoxil 500 mg capsule: hard, 1 capsule

AMT also derives generic representations of the branded (trade) products so that medications may be referred to generically. Taking the above examples:

Pack level	amoxycillin 500 mg capsule, 20
Unit level	amoxycillin 500 mg capsule
Generic product level	amoxycillin product

AMT's relationships link the generic and branded concepts together, such that it is possible to find the generic form or a branded product, or find all the branded products for a given generic product.

Finally AMT includes defining attributes of these generic and branded products as atomically accessible data. These attributes include:

- Ingredients[5,6]
- Strength
- Unit size
- Form
- Pack size
- Sub/component packs
- Container types

2.2 SNOMED CT terminologies

AMT v3 is a SNOMED CT terminology. It shares the same format and top-level hierarchy as SNOMED CT, and hence the same 'semantic space'. Therefore to fully understand AMT v3, its model and structure, it is useful to first understand the SNOMED CT terminology.

The following sections provide a very brief overview of the components of SNOMED CT terminology, their purpose, and how they work. Sections on AMT v3's model further on in this document should be read in context of this foundation.

[5] Ingredients represented in AMT v3 are limited to the intended active ingredients, not the complete ingredients of the medicines – for more detail please refer to the *AMT v3 Editorial Rules* [3].

[6] Note that in AMT v3, unlike AMT v2, |*Has pharmaceutical ingredient*| is not present and only intended active ingredients are modelled.

2.2.1 Concepts

Concepts identify and represent a category or class of thing needed to be recorded within health care.

Concepts can be very specific |*Acute pneumococcal bronchitis*|, or much more general |*Bacterial respiratory infection*|. General concepts whose meaning encompasses more specific concepts' meaning are said to 'subsume' the more specific concepts. Similarly a specific concept whose meaning falls completely within another concept's meaning is said to be 'subsumed by' the other concept.

2.2.2 Descriptions

Every concept has two or more descriptions. Concepts must have a 'Fully Specified Name' (FSN) description, which represents the formal, unambiguous meaning of the concept. Consequently the Fully Specified Name may be quite long, cumbersome and/or formal – not generally the way most clinicians usually refer to the concept.

Additionally concepts must have one or more 'Synonyms', one of which must be declared as the preferred way of referring to the concept (refer to Section 3.1.7). Unlike formal Fully Specified Names, Synonyms are common names for concepts. However as humans use some common names for different concepts depending on their contexts, Synonyms in SNOMED CT are not unique for a concept. That is, two concepts may both have synonyms with the same text (note they would still have differing description identifiers). Synonyms thereby introduce some ambiguity for the sake of usability.

2.2.3 Relationships

Relationships in SNOMED CT terminologies, and hence AMT, link two concepts and have a 'relationship type'. Each relationship for a concept is actually an individual logical assertion about the concept. Taken together, the relationships originating from a concept form a set of statements that define the meaning of the concept relative to the other concepts in the terminology.

These relationships attempt to represent the meaning of the concept's 'Fully Specified Name' description. Where there is a discrepancy the 'Fully Specified Name' is always considered to be correct.

2.2.4 Subtypes / Supertypes

SNOMED CT uses a special type of relationship, known as IS A, to represent subtype/supertype relationships between concepts. This relationship is used where a concept is wholly subsumed by another concept.

For example |*Panadol 500 mg tablet: film coated, 1 tablet*| IS A subtype (child) of |*paracetamol 500 mg tablet*| (parent).

Any statements (relationships in SNOMED CT) made by the parent concept are also true of the child concept. Taking the same example, all statements made of |*paracetamol 500 mg tablet*| are true of |*Panadol 500 mg tablet: film coated, 1 tablet*|.

2.2.5 Defined and Primitive concepts

In SNOMED CT concepts may be considered 'defined' or 'primitive', which indicates if the concept's set of relationships[7] are sufficient to define the concept. This is SNOMED CT's mechanism for representing necessity and sufficiency – http://en.wikipedia.org/wiki/Necessity_and_sufficiency.

All RF2 relationships are necessary (as opposed to sufficient) conditions. To distinguish whether a concept's relationships are sufficient in SNOMED CT concepts are marked as 'defined' or 'primitive'.

Defined concept: The concept's set of relationships is sufficient to define the concept in terms of the other concepts in the terminology (i.e. none of its defining characteristics are missing from its set of relationships).

Primitive concept: While necessary, the concept's set of relationships is not sufficient to define the concept in terms of the other concepts in the terminology.

An example of a defined concept in AMT v3 is |*paracetamol (medicinal product)*|. This concept's definition consists of:

- a relationship to the concept |*paracetamol (substance)*| to define the ingredient it contains; and
- a relationship to the concept |*medicinal product*| to define it as a type of product.

This definition means that any concept that contains paracetamol and is also a type of medicinal product is therefore a subtype of the concept |*paracetamol (medicinal product)*|.

However the concept |*paracetamol (substance)*| is declared as primitive in AMT v3 as its definition only has one relationship stating it is a subtype of |*Substance*|.

To define |*paracetamol (substance)*| within AMT v3 would require adding relationships and concepts defining the molecular structure of paracetamol, which is not relevant to AMT's use cases. Therefore these relationships are omitted and substance concepts in AMT v3 are declared primitive.

Similarly form concepts like |*tablet*| and unit of measure concepts like |*milligram*| are also primitive in AMT v3. For a full list of which AMT v3 concepts are defined and which are primitive, refer to Figure 2 on p.22.

2.2.6 Translation to common knowledge representation languages

Most implementations will not need to reason with the content of AMT or SNOMED CT, however it is possible if required.

The set of concepts, relationships and primitive/defined status can be translated into a common knowledge representation language for this purpose. The IHTSDO provide scripts to translate between SNOMED CT release formats and common knowledge representation formats such as OWL and KRSS.

Note that at present these IHTSDO scripts cannot be used on AMT v3, as AMT v3 uses an additional description logic feature not yet implemented in these scripts – concrete domains. The IHTSDO scripts will be updated as the concrete domains specification proceeds through the IHTSDO standards process.

[7] The set of relationships where the concept is the source of the relationship form the concept's relationships.

2.2.7 Open versus closed world assumption

Unlike most information models and databases, which operate under a 'closed world assumption', SNOMED CT terminologies like AMT operate under an 'open world assumption'.

Under a closed world assumption, any statement not known to be true is implicitly false. However under an open world assumption, the omission of a statement simply means that it is unknown – and could be true or false.

For example:

Statement:	Mary is a citizen of France.
Question:	Is Paul a citizen of France?
'Closed world' (e.g. SQL) answer:	No.
'Open world' answer:	unknown.

SNOMED CT concepts simply represent what is known, and do not include or preclude other conditions that may be true or false.

For example |*fracture of tibia*| obviously does not mean a fracture of the fibula or femur can be assumed. However the omission of a statement about the fibula and femur doesn't rule out a fracture of fibula or femur either. The statement |*fracture of tibia*| simply states that a tibia fracture exists, and any state of the fibula, femur or any other bone in the body is unknown.

Similarly for AMT, the concept |*paracetamol (medicinal product)*| has a relationship HAS INTENDED ACTIVE INGREDIENT to |*paracetamol (substance)*|. This states the presence of at least paracetamol in the product; however it makes no statement about the presence or absence of any other ingredients (active or otherwise). Therefore |*paracetamol + codeine (medicinal product)*| is a subtype of |*paracetamol (medicinal product)*| because at least paracetamol is present.

This differs from a closed world model which would assume |*paracetamol (medicinal product)*| contained only paracetamol and no other ingredients because no others were stated. To achieve the closed world semantics of |*paracetamol (medicinal product)*| in an open world model would require the addition of a statement that |*paracetamol (medicinal product)*| contains only |*paracetamol (substance)*| and a renaming of the concept to |*paracetamol only (medicinal product)*|.

2.2.8 Existential and universal restrictions

All SNOMED CT (hence AMT) relationships are expressed using existential restrictions. Universal restrictions are currently not supported in SNOMED CT's subset of description logic features.

Put simply, existential restrictions mean 'at least' or 'some', whereas universal restrictions mean 'for all' or 'only'.

To illustrate, consider the concept |*paracetamol (medicinal product)*| which is defined with a HAS INTENDED ACTIVE INGREDIENT relationship to |*paracetamol (substance)*|. The existential restriction applies to this relationship, and means 'at least an intended active ingredient of paracetamol must be present'. Therefore |*paracetamol (medicinal product)*| is really defined as 'paracetamol containing product' – i.e. any concept that contains at least paracetamol (such as 'paracetamol and codeine') is a subtype of |*paracetamol (medicinal product)*|.

If this relationship was modelled with a universal restriction instead, the concept would mean ‘paracetamol only product’. If this was the case, only concepts containing paracetamol and no other ingredients would be considered to be subtypes of |*paracetamol (medicinal product)*|. However, universal restrictions are not in use in SNOMED CT, AMT v2, or AMT v3. All relationships encountered in AMT v3 have the abovementioned ‘at least’ semantics.

2.2.9 Concept models

SNOMED CT terminologies define concept models for different hierarchies of content which define patterns and rules for concepts in those hierarchies. Concept models play a similar role for terminology that schemas do for relational data.

Each concept model essentially specifies the pattern of logic used to define concepts within that domain. For example, concept models define rules for a hierarchy or group of hierarchies that govern:

- the types of relationships used to define concepts;
- the combinations and groupings of relationships used to define concepts; and
- which types of concepts will be ‘defined’ and which will be ‘primitive’.

SNOMED CT contains numerous concept models for different types of content. AMT v3 contains only one concept model – the AMT v3 model.

2.3 AMT v3 model

This section provides a technical description of the AMT v3 model. In particular, it explains the various elements of the model, how they relate and what can be expected of the data conforming to this model. For a more detailed business description of the model, why it is modelled as it is, and the use cases it supports, please refer to *AMT v3 Overview and Use Cases* [1].

2.3.1 AMT v3 model diagrams

Figure 2 on p.22 uses a UML-like notation. Note that while this notation has been adapted from UML in an attempt to convey the structure of AMT v3, the closed world semantics implied by UML class diagrams cannot be assumed in the actual semantics of AMT v3. Some key differences worth noting are:

- 'Cyclic' IS A relationships shown on the diagram, such as the IS A relationship from |*Form*| to itself and |*Container*| to itself represent a sub-hierarchy of concepts. That is, a |*Form*| concept has one IS A relationship to either the AMT v3 concept |*Form*| or another form concept.

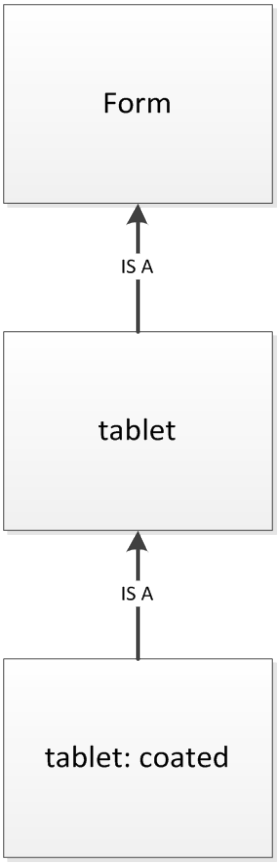


Figure 1: IS A relationships

- Individual concepts in SNOMED CT (and hence AMT) do not represent instances (i.e. things); they are classes (i.e. types/categories of things). For example |*medicinal product*| is an AMT concept which is the parent of many specific products such as |*paracetamol (medicinal product)*|. However |*paracetamol (medicinal product)*| is not an instance, it is itself a class that is a subclass of |*medicinal product*|.

The different types of entities in this diagram are represented as follows:

- A blue box denotes one of the AMT notable concepts (Section 2.3.2).
- A red box denotes an AMT reference set member (Section 2.3.4).

  Note that concrete domain reference sets referencing relationships have specific semantics specified in the *RF2 Specification Change Request: Addition of Concrete Domains* [2].
- A grey box denotes an AMT concept not from the notable classes (Section 2.3.3).

Although the AMT model is typically drawn with the most abstract concepts at the top, it is most easily understood by reading in the opposite direction to the typical reading order, that is: from the bottom right-hand corner upwards and to the left. Reading bottom-right to top-left takes the reader from the concepts representing the physical real world entities steadily up to higher levels of abstraction, that is, to more generic forms.

The descriptions of each component are assembled based on consistent editorial rules. For more details of these rules, and for full definitions of all AMT concepts, particularly in relation to policies on describing active ingredients and salts, please refer to *AMT v3 Editorial Rules* [3].

This model diagram should be read in context of AMT v3 being a SNOMED CT terminology – refer to Section 2.2 SNOMED CT terminologies.

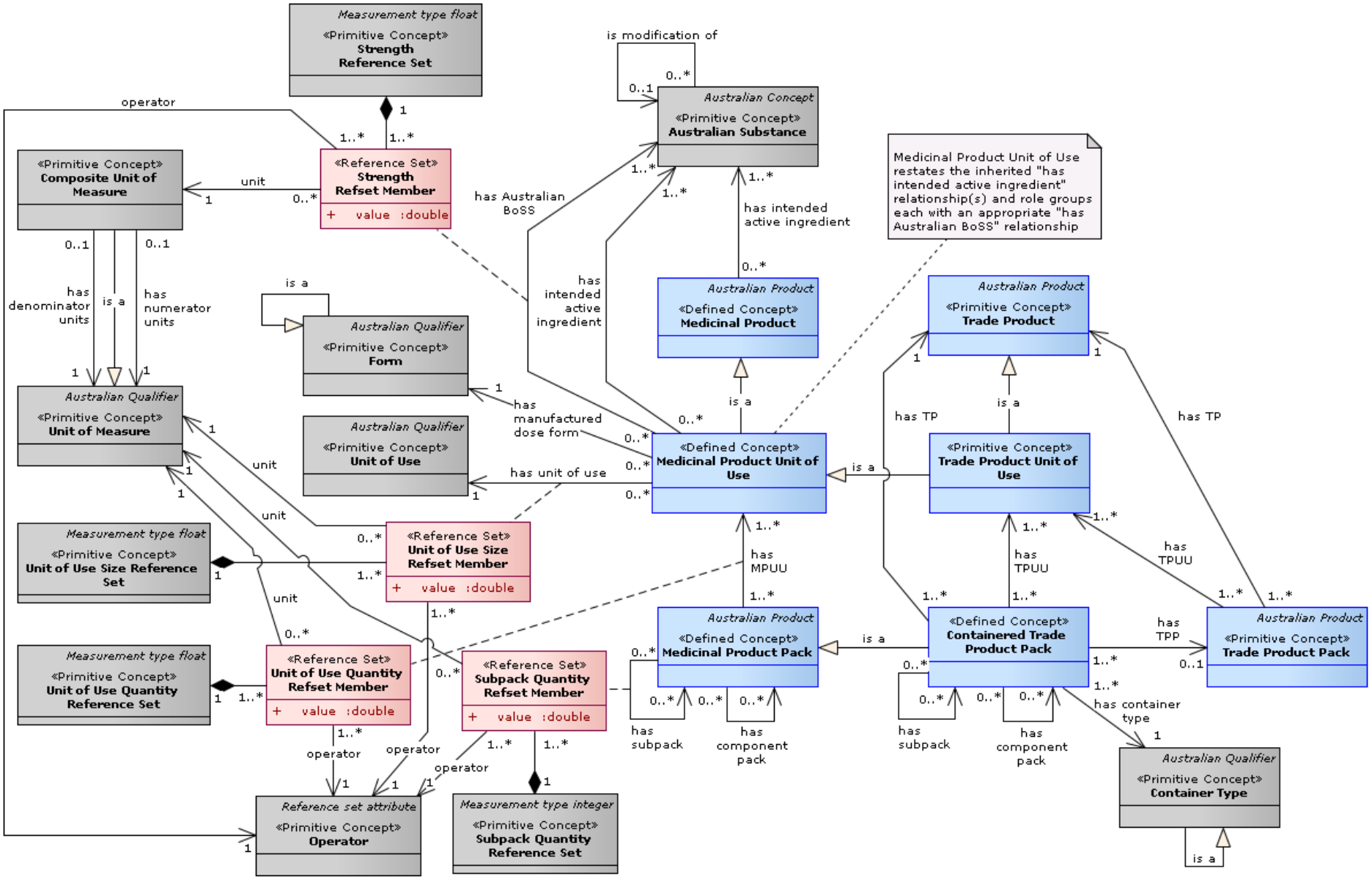


Figure 2: AMT v3 model

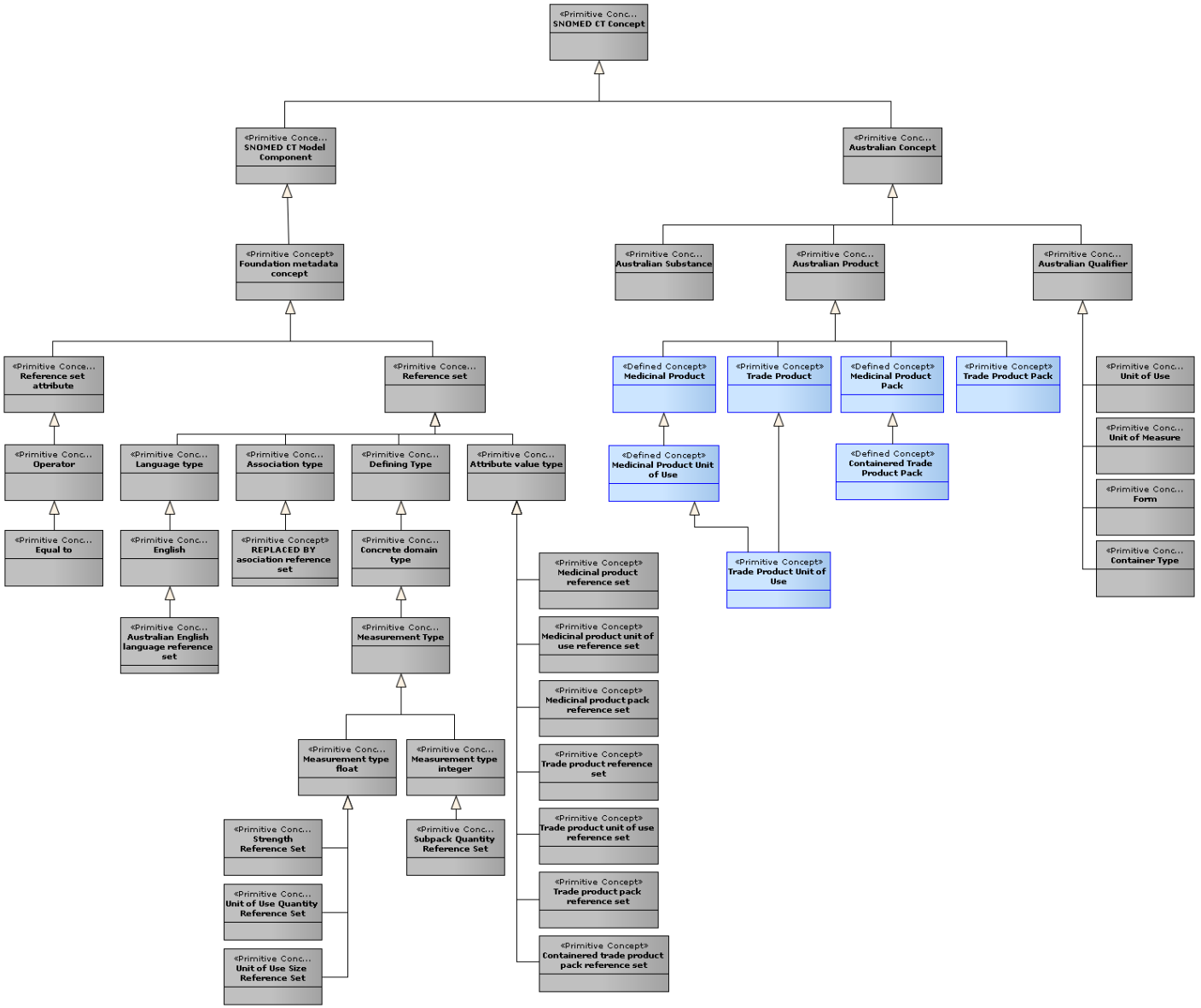


Figure 3 AMT v3 model concept hierarchy

2.3.2 'Notable concepts'

AMT v3's model is based around seven concept classes known as the 'Seven notable concepts'[8]. These concept classes head hierarchies of concepts representing different abstractions of branded products and their generic product equivalents at various levels of granularity.

The following sections describe each of these concept classes and their relationships, and are intended to be read in combination with Figure 2 on p.22.

Note that the left hand side of the 'seven notable concepts' are all prefixed 'Medicinal' and the right hand side 'Trade'. This indicates that the concept classes on the left represent generic medicines, whereas the concept classes on the right represent trade or branded medicines. See Figure 4 above for an example.

It is also worth noting that each concept class's relationships have been pushed up to the most abstract concept, however they are 'inherited' down to all descendants. For example the HAS INTENDED ACTIVE INGREDIENT relationships of a Medicinal Product Unit of Use concept are inherited by child concepts (which in this case are Trade Product Unit of Use concepts). See Section 2.2.4 Subtypes/Supertypes, which explains that all statements (relationships) true of a parent class are by definition true of the child.

Each of the following sections is illustrated by reference to the same five medicines, as defined below:

A	A simple oral solid medication – amoxycillin with a brand of Amoxil.
B	A brand name used across multiple generic medicines – Canesten (in this case Canesten Clotrimazole).
C	A combination pack Nexium Hp7 (one of the more complex examples likely to be encountered).
D	A manufacturer's generic – 0.9% sodium chloride solution manufactured by Baxter.
E	A medicine where two active ingredients are combined into the same formulation – paracetamol + codeine branded as Panadeine Forte.

The following diagram shows a simple example of how these concepts relate to each other for a single product.

Note that descriptions for examples used in this section follow AMT v3 editorial rules. This includes the use of:

- "+" to separate multiple ingredients of a multi-ingredient product and
- "(&)" to separate components of a subpack or component pack product.

Refer to *AMT v3 Editorial Rules* [3] for more details of these rules.

Note that the following sections describe the Stated Form of the AMT v3 data, as this is the form of the AMT v3 Beta. Note that expected cardinality of relationships expressed in the following sections will change in a Distribution Normal Form rendering of the same data. For more details refer to Section 3.4 Distribution Normal and Stated Forms.

[8] That is: Medicinal Product (MP), Medicinal Product Unit of Use (MPUU), Medicinal Product Pack (MPP), Trade Product (TP), Trade Product Unit of Use (TPUU), Contained Trade Product Pack (CTPP), and Trade Product Pack (TPP).

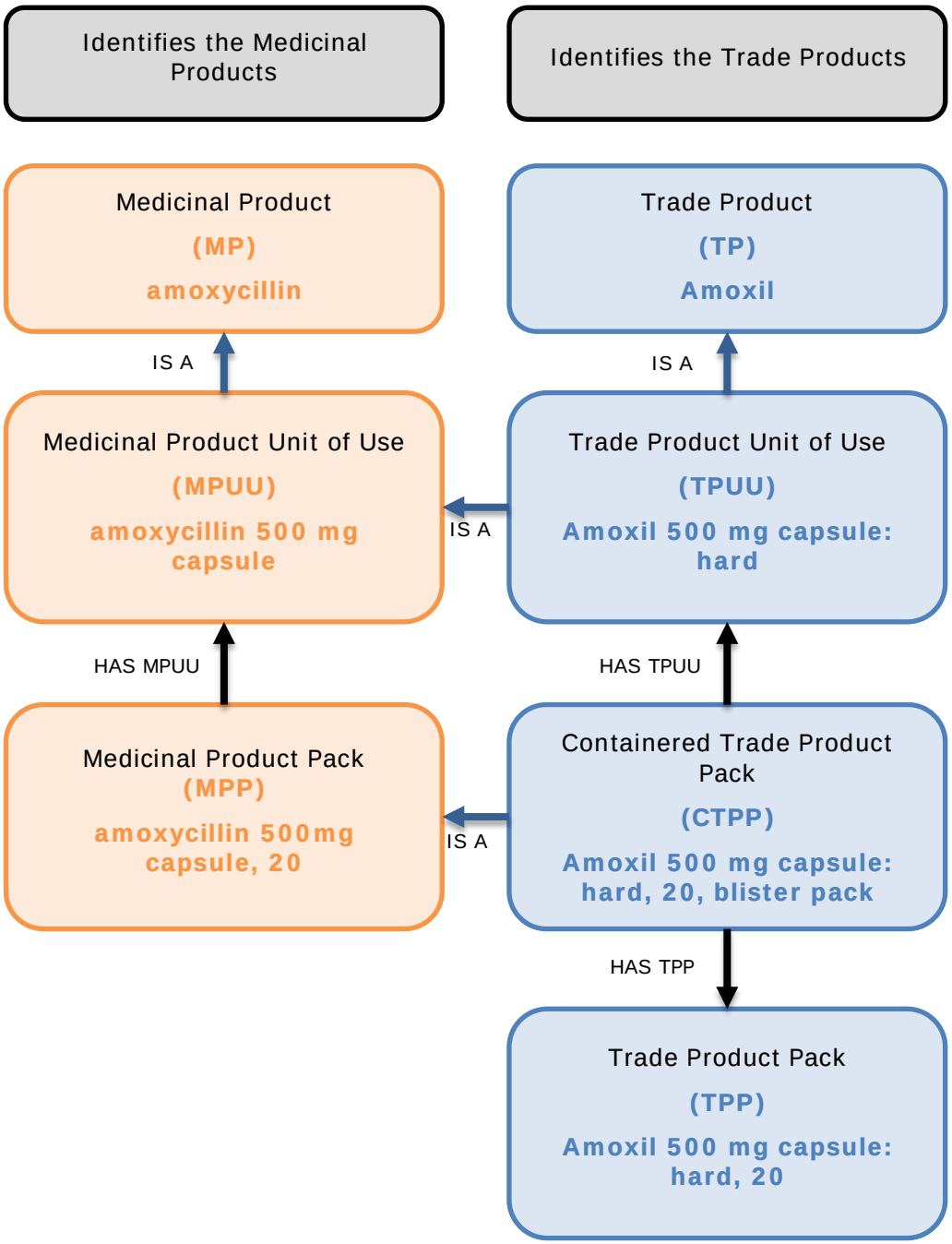


Figure 4 Seven notable concepts example

2.3.2.1 Medicinal Product

Medicinal Product is the abstract formulated representation of the therapeutic active ingredients that are used in the treatment of human patients in Australia.

A Medicinal Product concept is distinct in definition from a substance in that it represents the abstract notion of one or more substances combined into a product intended for medical use. Substance concepts, by contrast, represent the notion of some undefined quantity of a single substance.

While the Medicinal Product concept 21433011000036107 |*paracetamol*| and the Substance concept 2442011000036104 |*paracetamol*| have the same preferred term, they are distinct concepts with different meanings. The Medicinal Product concept 21433011000036107 |*paracetamol*| means 'paracetamol containing product' whereas the Substance concept 2442011000036104 |*paracetamol*| means 'some undefined quantity of paracetamol molecules'. The major difference between the concepts is that a Medicinal Product concept includes the abstract formulation of the substance(s) into a medicinal product.

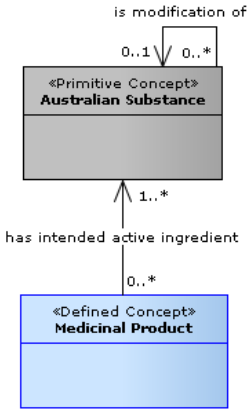


Figure 5 Medicinal Product (with direct relationships)

As an example the Medicinal Product 21433011000036107 |*paracetamol*| represents an abstraction of the MPUU concepts |*paracetamol 500 mg tablet*| or |*paracetamol 665 mg tablet: modified release, 1 tablet*|. The Substance concept 'paracetamol' represents an undefined quantity of paracetamol molecules.

Medicinal Product concepts:

- Are a subtype of the concept 30497011000036103 |*medicinal product*|
- Have one or more HAS INTENDED ACTIVE INGREDIENT relationships to Substance concepts:
    - An 'intended active ingredient' (Substance concept) represents a base active ingredient, or a salt active ingredient if the salt is deemed clinically significant. For example, |*paracetamol*| is a base ingredient and |*calcium carbonate*| is a salt ingredient. Refer to *AMT v3 Editorial Rules* [3] for further definitions of base/salt ingredients and clinical significance rules.

Examples of MPs include:

A	amoxycillin
B	clotrimazole
C	• esomeprazole • clarithromycin • amoxycillin
D	sodium chloride
E	paracetamol + codeine

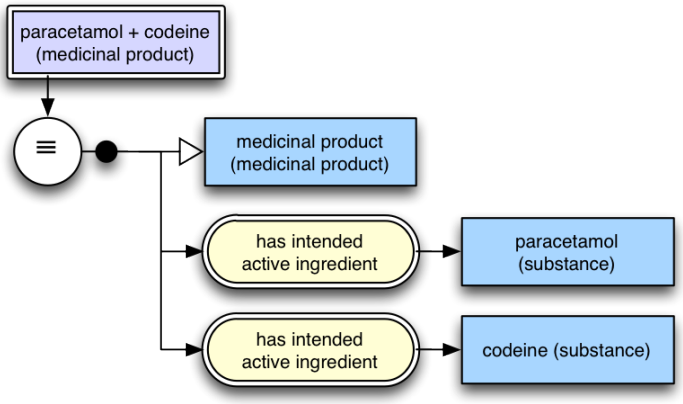


Figure 6 Example MP Stated Form modelling

2.3.2.2 Medicinal Product Unit of Use

Medicinal Product Unit of Use concepts represent an abstract formulation containing active ingredient, strength and form in a single dose form or a unit of use component of a multi-component formulation, devoid of brand.

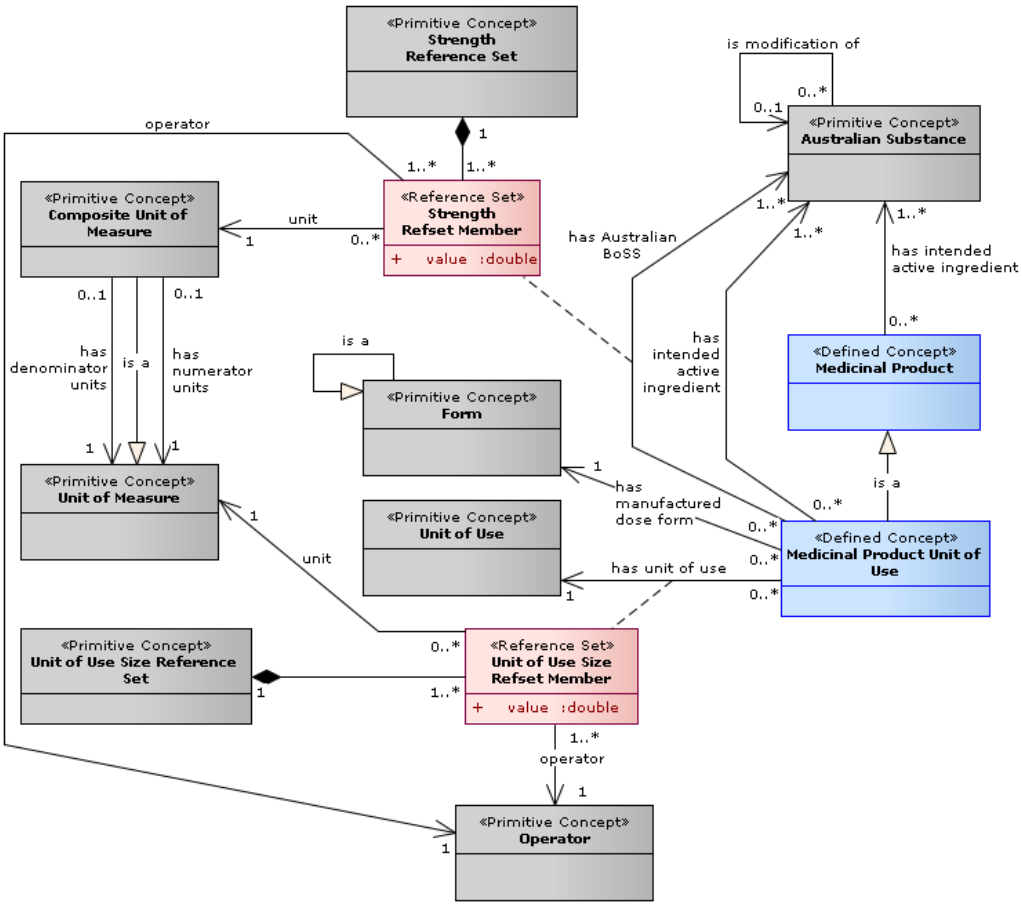


Figure 7: Medicinal Product Unit of Use (with direct relationships)

Each Medicinal Product Unit of Use concept:

- Is a subtype of the concept 30450011000036109 |*medicinal product unit of use*|.

- Has an IS A relationship to a Medicinal Product concept.
- Has one or more HAS INTENDED ACTIVE INGREDIENT relationships to Substance concepts. The target |*Substance*| concepts of these HAS INTENDED ACTIVE INGREDIENT relationship(s) exactly match the target |*Substance*| concepts of the HAS INTENDED ACTIVE INGREDIENT relationship(s) inherited from the supertype Medicinal Product concept.
- Has one HAS AUSTRALIAN BoSS relationship for each HAS INTENDED ACTIVE INGREDIENT relationship with each pair 'role grouped' together (refer to Section 3.1.4.1 Relationship groups).
    - Each HAS AUSTRALIAN BoSS relationship represents the Basis of Strength Substance (BoSS) used to express the strength of the intended active ingredient to which it is 'role grouped'. The BoSS Substance is as defined in the product registration details and Product Information document as defined by the TGA.
    - The remainder of the strength of each ingredient (numeric value and unit of measure) is attached to the HAS AUSTRALIAN BoSS relationship using the *Strength reference set* (see Section 2.3.4.1).
    - Note that strength details expressed in the *Strength reference set* are normalised to a denominator of one, whereas MPUU descriptions remain in a non-normalised representation closest to clinical usage. For example an MPUU description may express '300 mg/20 mL' which will be '15 mg/mL' in the *Strength reference set*.
- has exactly one HAS MANUFACTURED DOSE FORM relationship to a Form concept that represents the concept's dose form, e.g. tablet
- has exactly one HAS UNIT OF USE relationship to a Unit of Use concept that represents the concept's Unit of Use. The HAS UNIT OF USE relationship is quantified by a member of the *Unit of use size reference set*, which specifies the overall size of the MPUU (refer to Section 2.3.4.2 Unit of use size reference set).

Medicinal Product Unit of Use concepts are a generic form of one or more equivalent Trade Product Units of Use concepts, for example:

A	amoxycillin 500 mg capsule
B	clotrimazole 1% cream
C	• esomeprazole 20 mg tablet: enteric • clarithromycin 500 mg tablet • amoxycillin 500 mg capsule
D	sodium chloride 0.9% (9 g/1 L) injection, bag
E	paracetamol 500 mg + codeine phosphate 30 mg tablet

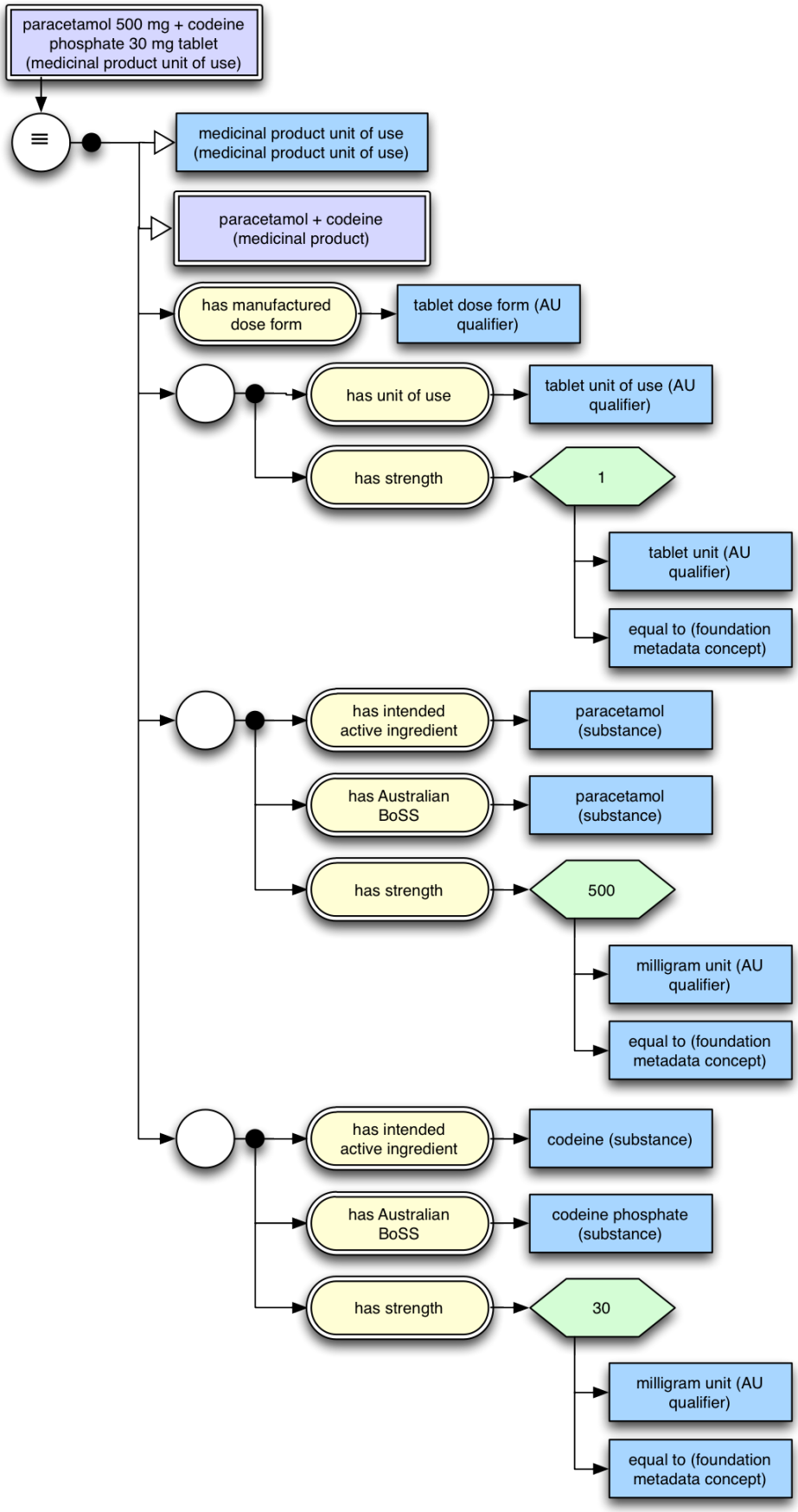


Figure 8 Example MPUU Stated Form modelling

2.3.2.3 Medicinal Product Pack

Medicinal Product Pack (MPP) concepts represent the abstract concept of a marketable medicinal entity available for patient use, devoid of brand and container type. Each MPP concept is the generic version of a set of one or more equivalent Containered Trade Product Pack concepts. MPP concepts also contain generic information about subpacks and component packs, corresponding to the subpack and component packs at the CTPP level.

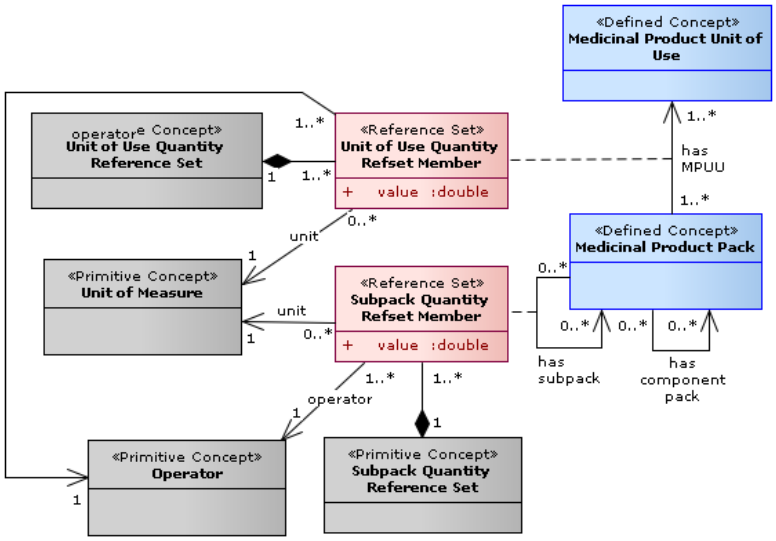


Figure 9: Medicinal Product Pack (with direct relationships)

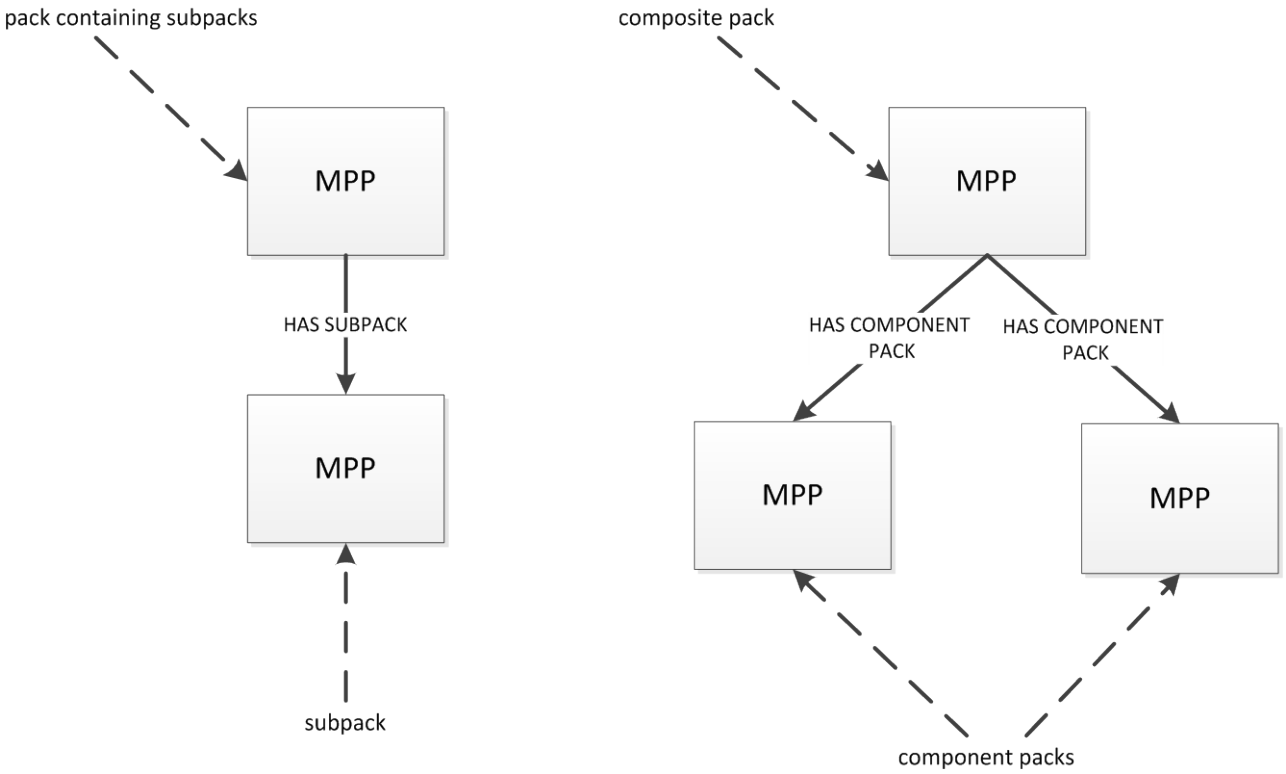
Each Medicinal Product Pack concept:

- Is a subtype of the concept 30513011000036104 |*medicinal product pack*|.
- Has one or more HAS MPUU relationships to the Medicinal Product Unit of Use concepts it contains. Each HAS MPUU relationship is quantified by the *Unit of use quantity reference set*, which specifies the quantity of the Medicinal Product Unit of Use instances the Medicinal Product Pack contains (refer to Section 2.3.4.3 Unit of use quantity reference set).
- Has multiple HAS COMPONENT PACK relationships to other Medicinal Product Pack concepts if the concept is a *composite pack*
- Has one HAS SUBPACK relationship to another Medicinal Product Pack concepts if the concept is a *pack containing subpacks*.

Where:

- A *composite pack* is a Medicinal Product Pack concept that has multiple HAS COMPONENT PACK relationships to component Medicinal Product Pack concepts. Composite packs represent the composition of multi-component kits.
- A *pack containing subpacks* is a Medicinal Product Pack concept that has one or more HAS SUBPACK relationships to subpack Medicinal Product Pack concepts. *Packs containing subpacks* represent a sequential multi-component item (currently in AMT v3 these include oral contraceptives and some hormone replacement therapy products).
    - Each HAS SUBPACK relationship is quantified by the Subpack quantity reference set, which specifies the number of *subpacks* contained within each *pack containing subpacks* (see 2.3.4.4 Subpack quantity reference set).

- a *component pack* is a Medicinal Product Pack that is the target of a HAS COMPONENT PACK relationship from a composite Medicinal Product Pack. Component packs represent parts of a multi-component kit with separate primary packaging.
- a *subpack* is a Medicinal Product Pack that is the target of a HAS SUBPACK relationship from a Medicinal Product Pack. *Subpacks* represent a part of a sequential multi-component item (currently oral contraceptives and some hormone replacement therapy products).



Examples of MPPs include:

A	amoxycillin 500 mg capsule, 20
B	clotrimazole 1% cream, 20 g
C	• esomeprazole 20 mg tablet: enteric [14 tablets] (&) clarithromycin 500 mg tablet [14 tablets] (&) amoxycillin 500 mg capsule [28 capsules], 1 pack • clarithromycin 550 mg tablet, 14 • esomeprazole 20 mg tablet: enteric, 14 • amoxycillin 500 mg capsule, 28
D	sodium chloride 0.9% (9 g/1 L) injection, 1 x 1 L bag
E	paracetamol 500 mg + codeine phosphate 30 mg tablet, 20

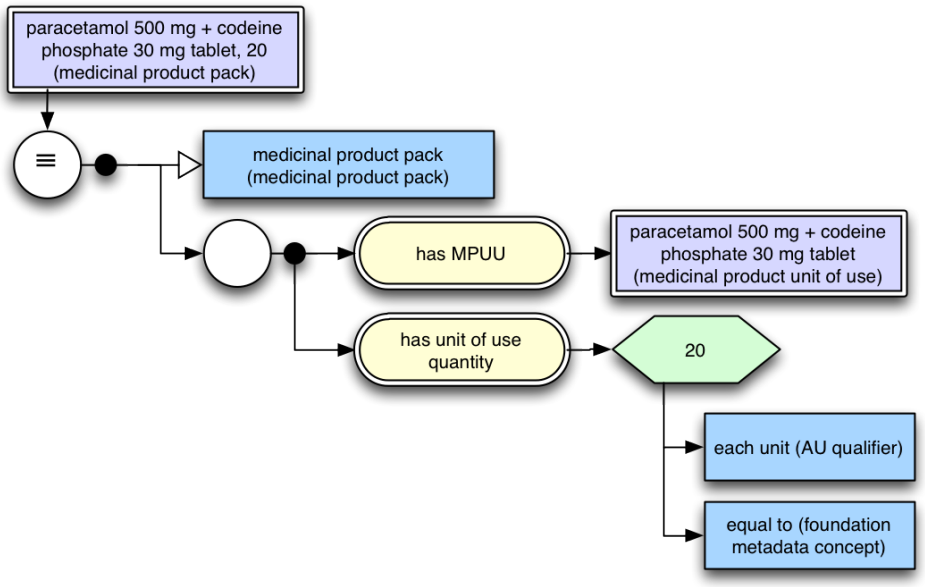


Figure 10 Example MPP Stated Form modelling

Examples of component pack MPPs are:

- esomeprazole 20 mg tablet, 14
- clarithromycin 500 mg tablet, 14
- amoxycillin 500 mg capsule, 28

Which are component packs of the composite pack MPP:

- esomeprazole 20 mg tablet [14 tablets] (&) clarithromycin 500 mg tablet [14 tablets] (&) amoxycillin 500 mg capsule [28 capsules], 1 pack

An example of a subpack MPP is:

- levonorgestrel 50 microgram + ethinyloestradiol 30 microgram tablet [6 tablets] (&) levonorgestrel 75 microgram + ethinyloestradiol 40 microgram tablet [5 tablets] (&) levonorgestrel 125 microgram + ethinyloestradiol 30 microgram tablet [10 tablets] (&) inert substance tablet [7 tablets], 28

which is a subpack of the MPP

- levonorgestrel 50 microgram + ethinyloestradiol 30 microgram tablet [24 tablets] (&) levonorgestrel 75 microgram + ethinyloestradiol 40 microgram tablet [20 tablets] (&) levonorgestrel 125 microgram + ethinyloestradiol 30 microgram tablet [40 tablets] (&) inert substance tablet [28 tablets], 112 [4 x 28 tablets]

For a full definition of subpacks and component packs and how they are differentiated please refer to Appendix A.

2.3.2.4 Trade Product

The Trade Product (TP) represents the product brand name, for either single component products, or components of multi-component products regardless of ingredients. The TP may also include any additional detail necessary for identification. For example: strength representation for multi-ingredient products (where this is necessary for differentiation); proprietary form, delivery device or container; an alternate name which has market recognisability.

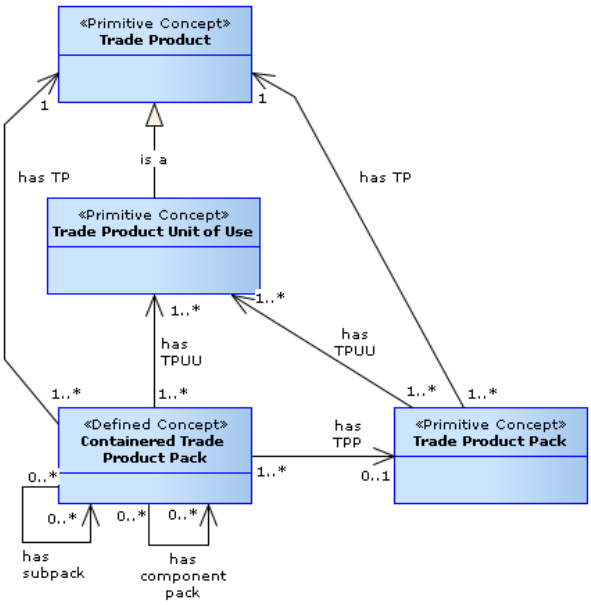


Figure 11: Trade Product (with direct relationships)

Trade Product concepts are always primitive and are used to define the trade side of the 'notable concepts' and distinguish them from the generic side.

Examples of TPs include:

A	Amoxil
B	Canesten Clotrimazole
C	• Nexium • Klacid • Amoxil • Nexium Hp7
D	Sodium Chloride (Baxter)
E	Panadeine Forte

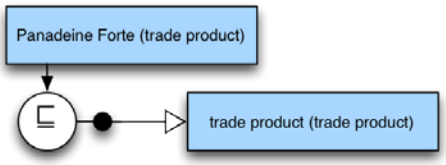


Figure 12 Example TP Stated Form modelling

2.3.2.5 Trade Product Unit of Use

Trade Product Unit of Use represents a marketable formulation containing active ingredient, strength and form in a single dose form or a unit of use component of a multi-component pack.

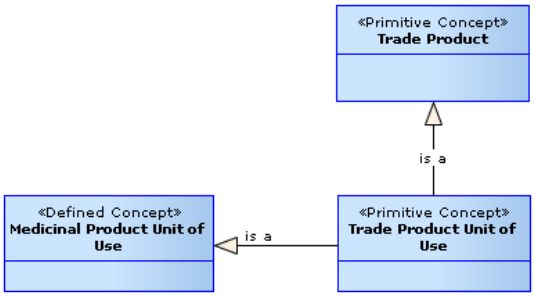


Figure 13: Trade Product Unit of Use (with direct relationships)

Each Trade Product Unit of Use concept is

- a subtype of the concept 30425011000036101 |*trade product unit of use*|.
- always a subtype of a single Trade Product, from which its 'brand' is inherited.
- always a subtype of a Medicinal Product Unit of Use, from which it inherits its
    - Ingredients
    - Ingredient strengths, including Basis of Strength Substance
    - Form
    - Unit of use

Trade Product Unit of Use concepts are not considered fully defined by these attributes as it is possible for two different Trade Product Unit of Use concepts to have the same ingredients, strength, form and brand and still be a different product. Examples of this are products with different flavours or other un-modelled characteristics.

For example the following two TPUU concepts are distinct concepts, yet have the same set of relationships in AMT v3. As a result they are declared primitive as additional defining information is not modelled in AMT v3 (refer to Section 2.2.5 for more detail on primitive versus defined concepts).

- |*Neulasta 6mg/0.6 mL injection: solution, syringe*|
- |*Neulasta with Automatic Needle Guard 6mg/0.6mL injection: solution, syringe*|

Examples of TPUUs include:

A	Amoxil 500 mg capsule: hard
B	Canesten Clotrimazole 1% cream
C	• Nexium 20 mg tablet: enteric • Klacid 500 mg tablet: film-coated • Amoxil 500 mg capsule: hard
D	Sodium Chloride (Baxter) 0.9% (9 g/1 L) injection: solution, bag

E	Panadeine Forte tablet: uncoated

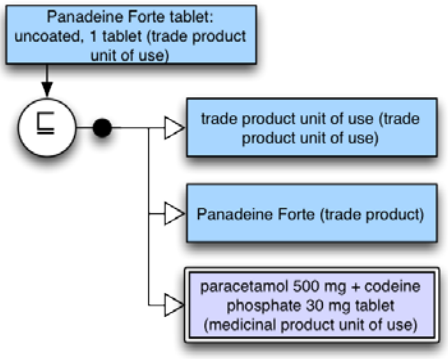


Figure 14 Example TPUU Stated Form modelling

2.3.2.6 Containered Trade Product Pack

Containered Trade Product Pack represents the marketable medicinal entity available for patient use, with details of the container type. Its attributes include the brand, the types and number of units it contains, and container type.

For details of when to use a TPP concept versus a CTPP concept, refer to Section 7.4.

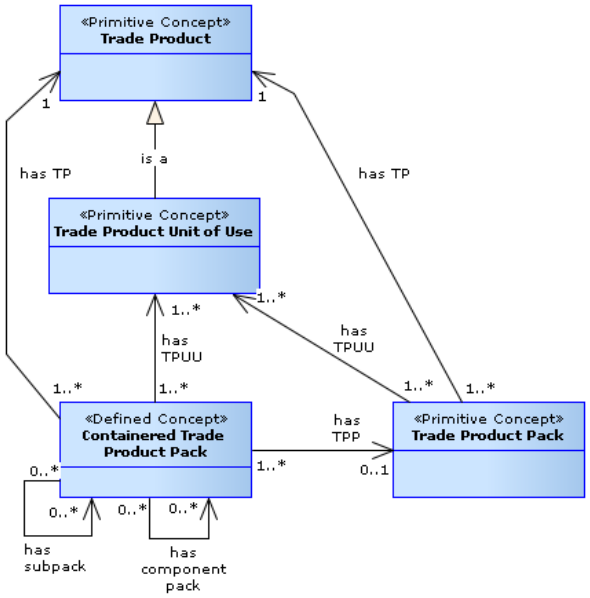


Figure 15: Containered Trade Product Pack (with direct relationships)

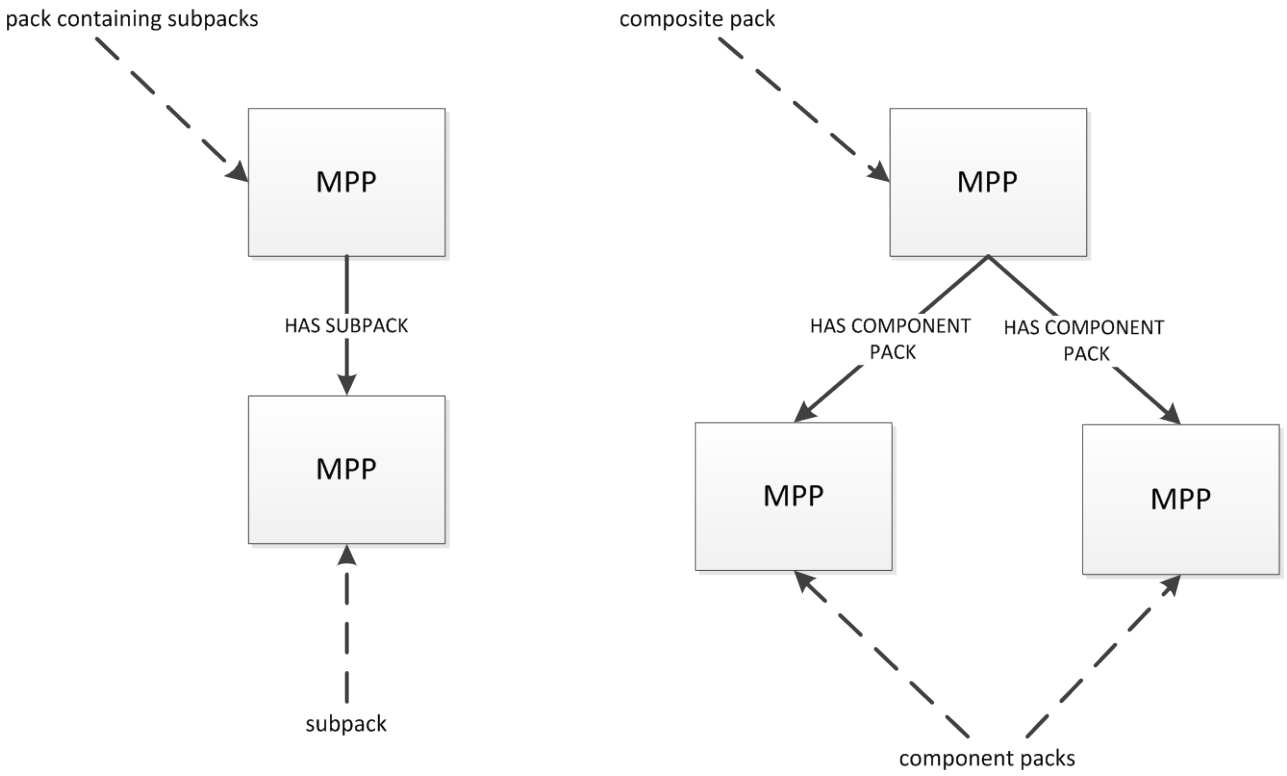
Each Containered Trade Product Pack concept

- Is a subtype of 30537011000036101 |*containered trade product pack*|.
- is always a subtype of a single Medicinal Product Pack, from which it inherits
  - one or more HAS MPUU relationships to the Medicinal Product Unit of Use concepts it contains
  - multiple HAS COMPONENT PACK relationships to Medicinal Product Pack concepts if the concept is a *composite pack*
  - one HAS SUBPACK relationship to a Medicinal Product Pack if the concept is a pack containing *subpacks*

- has one or more HAS TPUU relationships to the Trade Product Unit of Use concepts it contains, matching the HAS MPUU relationships it inherits.
- has exactly one HAS TP relationship to a Trade Product concept which represents its brand.
- has a single HAS TPP relationship to a Trade Product Pack concept if the concept is not a *component pack* or *subpack*.
- has multiple HAS COMPONENT PACK relationships to other Contained Trade Product Packs if the concept is a *composite pack*. Note that if present these relationships will correlate to and refine the HAS COMPONENT PACK relationships inherited from the parent MPP concept.
- has one HAS SUBPACK relationships to another Contained Trade Product Packs if the concept is a *pack containing subpacks.* Note that if present this relationship will correlate to and refine the HAS SUBPACK relationship inherited from the parent MPP concept.

Where

- a *composite pack* is a Contained Trade Product Pack concept that has multiple HAS COMPONENT PACK relationships to component Contained Trade Product Pack concepts. Composite packs represent the composition of multi-component kits.
- a *pack containing subpacks* is a Contained Trade Product Pack concept that has one HAS SUBPACK relationship to a subpack Contained Trade Product Pack concept. *Packs containing subpacks* represent a sequential multi-component item (currently oral contraceptives and some hormone replacement therapy products).
    - Each HAS SUBPACK relationship is quantified by the Subpack quantity reference set, which specifies the number of *subpacks* contained within each *pack containing subpacks* (see 2.3.4.4 Subpack quantity reference set).
- a *component pack* is a Contained Trade Product Pack that is the target of a HAS COMPONENT PACK relationship from a composite Contained Trade Product Pack. Component packs represent parts of a multi-component kit with separate primary packaging.
- a *subpack* is a Contained Trade Product Pack that is the target of a HAS SUBPACK relationship from a Contained Trade Product Pack. Subpacks represent a part of a sequential multi-component item (currently oral contraceptives and some hormone replacement therapy products).



The structure of CTPP sub/super and component/composite packs mirrors the MPP structure of sub/super and component/composite packs respectively. For example:

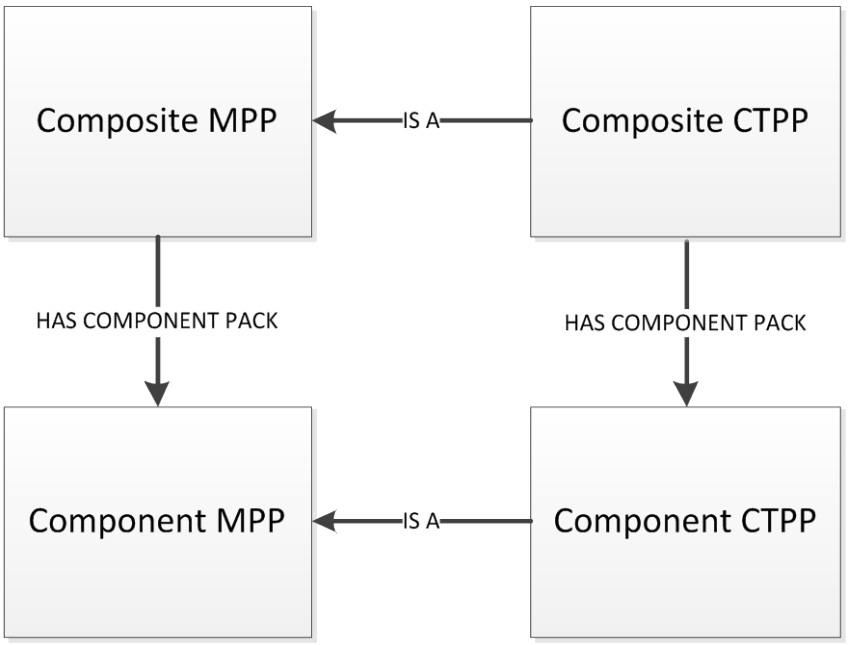


Figure 16: Mirrored structure of MPP and CTPP composite packs

Additionally the following rules apply to the HAS TPP relationship between CTPP and TPP

- All *composite* CTPPs (which have multiple HAS COMPONENT PACK relationship/s to *component* CTPPs) have one HAS TPP relationship to a TPP concept
- All CTPPs containing *subpacks* (which have one HAS SUBPACK relationship to a *subpack* CTPPs) have one HAS TPP relationship to a TPP concept

- Most CTPPs (single component), which do not contain other packs and are not contained within other packs (are not the source or target of a HAS COMPONENT PACK or HAS SUBPACK relationship) have one HAS TPP relationship to a TPP concept. However there are some exceptions, most of which represent injections with diluents.
- Component CTPP concepts generally do not have a HAS TPP relationship to a TPP concept. There are some exceptions – any component CTPP which is also available individually outside of the composite pack/s it is in will also have a single HAS TPP relationship to a TPP.
- All subpack CTPP concepts have one HAS TPP relationship to a TPP concept

Examples of CTPPs include:

A	Amoxil 500 mg capsule: hard, 20, blister pack
B	Canesten Clotrimazole 1% cream, 20 g, tube
C	• Nexium Hp7, 1 pack, composite pack • Amoxil 500 mg capsule: hard, 28, blister pack • Klacid 500 mg tablet: film-coated, 14, blister pack • Nexium 20 mg tablet: enteric, 14, blister pack
D	Sodium Chloride (Baxter) 0.9% (9 g/ 1 L) injection: solution, 1 x 1 L bag AHB 1324
E	Panadeine Forte tablet: uncoated, 20, blister pack

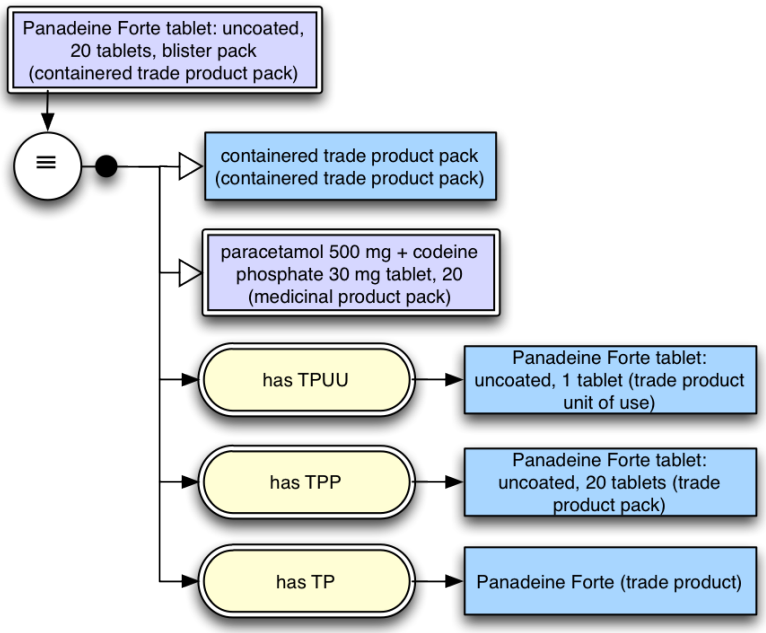


Figure 17 Example CTPP Stated Form modelling

Examples of component pack CTPPs are:

- Nexium 20 mg tablet: enteric-coated, 14 tablets, blister pack
- Klacid 500 mg tablet: film-coated, 14 tablets, blister pack
- Amoxil 500 mg capsule: hard, 28 capsules

The above are component packs of the composite pack:

- Nexium Hp7, 1 pack, composite pack

An example of a subpack CTPP is:

- Trifeme, 28 tablets, blister pack

The above concept is a subpack of the CTPP:

- Trifeme, 112 tablets [4 x 28 tablets]

For a full definition of subpacks and component packs and how they are differentiated please refer to Appendix A.

2.3.2.7 Trade Product Pack

Trade Product Pack (TPP) represents the marketable medicinal entity available for patient use devoid of container type.

For details of when to use a TPP concept versus a CTPP concept, refer to Section 7.4.

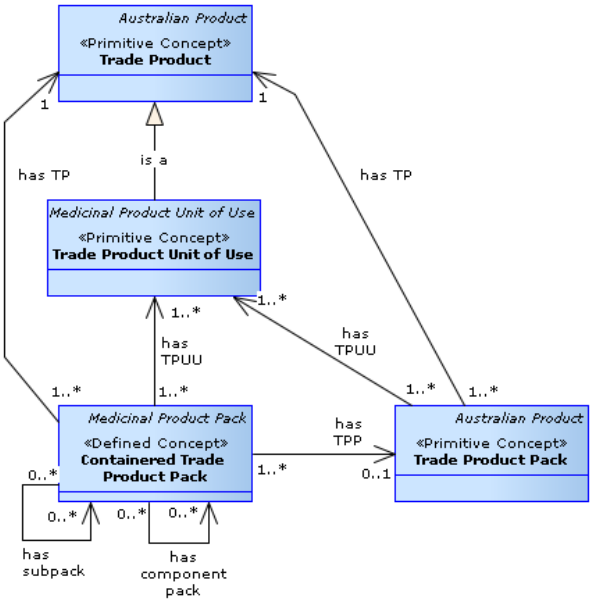


Figure 18: Trade Product Pack (with direct relationships)

Each Trade Product Pack concept

- Is a subtype of the concept 30404011000036106 |*trade product pack*|.
- has exactly one HAS TP relationship to a Trade Product which represents its brand.
- has one or more HAS TPUU relationships to the Trade Product Unit of Use concepts of which this Trade Product Pack represents the collection.

Examples of Trade Product Packs include:

A	Amoxil 500 mg capsule: hard, 20
B	Canesten Clotrimazole 1% cream, 20 g
C	Nexium Hp7, 1 pack Note there is no TPP for • Amoxil 500 mg capsule: hard, 28, blister pack

	• Nexium 20 mg tablet: enteric, 14, blister pack However there is a TPP for Klacid 500 mg tablet: film-coated, 14, blister pack
D	Sodium Chloride (Baxter) 0.9% (9 g/1 L) injection: solution, 1 x 1 L bag
E	Panadeine Forte tablet: uncoated, 20

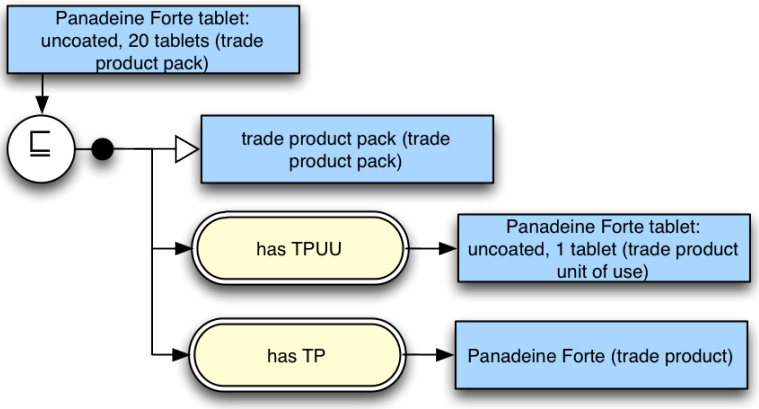


Figure 19 Example TPP Stated Form modelling

2.3.3 Other concept classes

The other concepts that exist in the model provide the necessary concepts to support the definition of the main 'notable concepts'. This section contains an overview of these concepts. More information can be found in *AMT v3 Editorial Rules* [3].

2.3.3.1 Substance

Substance concepts are primitive and identify chemical elements, compounds, and mixtures.

Substances may have an IS MODIFICATION OF relationship to another substance. If present, this means that the concept is a chemical compound incorporating the other substance. For example the substance concept |*calcium carbonate*| has an IS MODIFICATION OF relationship to the substance concept |*calcium*|. Using this relationship in reverse it is possible to find all the modified forms of a base substance.

2.3.3.2 Form

Each |fo*rm*| concept represents a dose formulation, for example, tablet, capsule or eye drop. These form concepts are used in AMT v3 to define the generic manufactured dose form of medicines concepts.

Form concepts may also be subtypes of (have IS A relationships to) other form concepts. This creates a hierarchy of form concepts, with more specific types of form towards the bottom of the hierarchy. The top level form concepts have an IS A relationship to the AMT concept |*form*|.

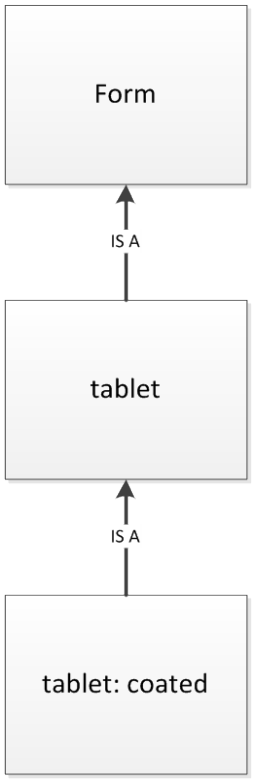


Figure 20: Example form hierarchy

2.3.3.3 Unit of measure

Under the concept |*calcium*|, AMT contains a hierarchy of concepts which represent units used to measure quantities in AMT – for example milligram.

Units of measure are sourced from the TGA data for the medications included in AMT and are defined in the TGA Approved Terminology for Medicines[9].

This hierarchy also includes units of measure known as 'composite units of measure'. A composite unit of measure is used to express a concentration based unit, expressed using a numerator and denominator unit of measure. For example |*mg/mL*|, where 'milligram' is the numerator unit and 'millilitre' is the denominator unit.

Composite unit of measure concepts will have:

- exactly one HAS NUMERATOR UNIT relationship to a non-composite unit of measure concept and
- exactly one HAS DENOMINATOR UNIT relationship to a non-composite unit of measure concept

Non-composite unit of measure concepts do not have HAS NUMERATOR UNIT or HAS DENOMINATOR UNIT relationships.

2.3.3.4 Unit of Use

The unit of use describes a discrete unit dose form (e.g. tablet, capsule) or a continuous form where a consistent physically measurable unit or sub-unit cannot be identified (e.g. cream, eye drops).

[9] TGA publishes their Approved Terminology for Medicines at http://www.tga.gov.au/industry/medicines-approved-terminology.htm.

2.3.3.5 Container type

AMT container type concepts represent the types of container that immediately cover a medicine. Examples include ampoule, bottle, blister pack, vial etc. These concepts are derived from TGA Approved Terminology for Medicines.

2.3.4 Reference sets

AMT v3 includes:

- reference sets that carry some of the semantics in the AMT model; and
- reference sets containing convenient enumeration of the 'Notable concepts'.

These reference sets specific to AMT v3 are described in this section. In addition, Section 3.1 includes a description of the other reference sets distributed with AMT v3 which form part of the RF2 representation of AMT v3.

2.3.4.1 Strength reference set

The *Strength reference set* is a 'concrete domain' reference set (refer to Section 3.2) that includes numerical datatype properties used in the definition of Medicinal Product Unit of Use concepts.

Specifically, members of the *Strength reference set* refer to HAS AUSTRALIAN BoSS relationships and add a property including:

- a value (e.g. 500);
- a reference to the unit of measure for the value (e.g. milligram/each); and
- a reference to an operator, at present only 700000051000036108 |*Equal to*|.

This has the effect of further qualifying the HAS AUSTRALIAN BoSS relationship by specifying the quantity of the ingredient included – from the example above 'equal to 500 milligram/each'.

Note that the strength of the intended active ingredient present is expressed in terms of the 'Australian BoSS', not the intended active ingredient if this differs. For example if a concept is defined with

- HAS AUSTRALIAN BoSS |*morphine sulfate*| and
- HAS INTENDED ACTIVE INGREDIENT |*morphine*| and
- strength of '10 milligram/each'

This is specifying that |*morphine*| is present, and it is present in a strength equivalent to 10 milligrams worth of |*morphine sulfate*|. Note that this is **not** saying that |*morphine sulfate*| is present, |*morphine sulfate*| is just being used to express the quantity of |*morphine*| that is present. The actual quantity of |*morphine*| present would be less than 10 milligrams due to |*morphine sulfate*|'s higher molecular weight.

Concentration or rate based strengths are represented using 'composite units of measure', such as 'milligram per millilitre' or 'milligram per hour'. Medicine concept preferred terms may still use the most preferred human representation, however the *Strength reference set* will always represent normalised strength values using a denominator of one. For example '50 mg/5 mL' might be the human readable strength, whereas the *Strength reference set* will contain '10 mg / mL'. See Section 5.4.3 for an example of how to access ingredient strength.

Most active HAS AUSTRALIAN BoSS relationships will be referenced by a single active member of the *Strength reference set* – i.e. there is a strength associated with the BoSS. There are a small number of HAS AUSTRALIAN BoSS relationships not referenced by the *Strength reference set*. These are cases where no strength exists for the ingredient such as:

- Foods/nutritional supplements (e.g. vitamins, minerals and trace elements with carbohydrate).
- Diagnostic strips (e.g. glucose indicator blood).
- Non-medicated dressings/bandages (e.g. bandage tubular; bandage retention cohesive heavy).
- Inert substances, diluents.

For details of the structure and datatypes refer to Section 3.2 Concrete domains and data type properties.

2.3.4.2 Unit of use size reference set

The *Unit of use size reference set* is a 'concrete domain' reference set (refer to Section 3.2) that specifies the quantity of a Medicinal Product Unit of Use that constitutes a 'unit of use'. This is usually the administrable dose unit (e.g. 1 tablet or 5 mL) however for continuous products (e.g. creams) a measurable administrable dose unit does not exist.

For Medicinal Product Unit of Use concepts premeasured, measured or indivisible forms, this reference set will simply indicate a single unit of this form. For example the unit of use size for a |*500 mg paracetamol tablet*| is '1 tablet'. Other examples are '1 patch' or '1 suppository'.

For concentration based products, the unit of use size is the total quantity of the medicine in the Medicinal Product Unit of Use concept. For example a |*benztropine mesylate 2 mg/2 mL injection, ampoule*| has its strength in the *Strength reference set* as '1 mg/mL' and in the *Unit of use size reference set* its value is '2 mL'.

The combination of the *Unit of use size reference set* and the *Strength reference set* provide the ability to calculate the total ingredient quantity for a unit – refer to Section 5.4.4 for an example of this. Note that there are rounding issues to consider, which are discussed further in Section 7.14.

Some products do not have an entry in the *Unit of use size reference set*. These are typically continuous form products such as solutions, creams and ointments where it is not possible to sensibly provide a unit of use size as a precise amount (for example mouthwash or eye ointment). Instead the unit of use size is not stated for these MPUU concepts, and the quantity is expressed only in the MPP packaging.

For details of the structure and datatypes refer to Section 3.2 Concrete domains and data type properties.

2.3.4.3 Unit of use quantity reference set

The *Unit of use quantity reference set* is a 'concrete domain' reference set that includes numerical datatype properties used in the definition of Medicinal Product Pack concepts.

Members of the *Unit of use quantity reference set* refer to HAS MPUU relationships between Medicinal Product Pack and Medicinal Product Unit of Use concepts and add a property including:

- An integer value (e.g. 30).
- A reference to the unit of measure (e.g. tablet).

- A reference to an operator, at present only 700000051000036108 |*Equal to*|.

This has the effect of quantifying the HAS MPUU relationship to specify how many of each type of Medicinal Product Unit of Use concepts a Medicinal Product Pack concept contains. From the example above 'equal to 30 tablets'.

Every active HAS MPUU relationship[10] between a Medicinal Product Pack concept and a Medicinal Product Unit of Use concept will be referred to by a single active member of the Unit of use quantity reference set.

For details of the structure and datatypes refer to Section 3.2 Concrete domains and data type properties.

2.3.4.4 Subpack quantity reference set

The *Subpack quantity reference set* is a 'concrete domain' reference set (refer to Section 3.2) that quantifies the HAS SUBPACK relationship a Medicinal Product Pack containing subpacks has with each 'subpack' Medicinal Product Pack. That is for each HAS SUBPACK relationship, the *Subpack quantity reference set* specifies how many of the 'subpack' Medicinal Product Packs are included in the Medicinal Product Pack concept containing the subpacks.

For example the HAS SUBPACK relationship between the 'subpack':

- levonorgestrel 50 microgram + ethinyloestradiol 30 microgram tablet [6 tablets] (&) levonorgestrel 75 microgram + ethinyloestradiol 40 microgram tablet [5 tablets] (&) levonorgestrel 125 microgram + ethinyloestradiol 30 microgram tablet [10 tablets] (&) inert substance tablet [7 tablets], 28

and the MPP:

- levonorgestrel 50 microgram + ethinyloestradiol 30 microgram tablet [24 tablets] (&) levonorgestrel 75 microgram + ethinyloestradiol 40 microgram tablet [20 tablets] (&) levonorgestrel 125 microgram + ethinyloestradiol 30 microgram tablet [40 tablets] (&) inert substance tablet [28 tablets], 112 [4 x 28 tablets]

is referred to by the *Subpack quantity reference set* with the value '4 each', indicating the parent pack contains four of the 'subpack' concept instances.

For details of the structure and datatypes refer to Section 3.2 Concrete domains and data type properties.

2.3.4.5 'Notable concept' reference sets

AMT also contains a reference set for each of the 'Notable concepts' – a total of seven reference sets. These are intended to present each notable class' concepts in pre-set lists to simplify implementations.

Each reference set references the active concepts from each of that 'Notable concept' types. For example the *Medicinal product unit of use reference set* refers to the active Medicinal Product Unit of Use concepts.

[10] The AMT v3 Beta currently has two HAS MPUU relationships that violate this rule and are not referenced in the *Unit of use quantity reference set*. These are related to "ketotifen 0.025%.(250 microgram/mL) eye drops, 2.5 mL" and "ketotifen 0.025% (250 microgram/mL) eye drops, 5 mL". This is a known issue which is scheduled to be corrected.

These reference sets are provided as a convenience and do not have any bearing on the meaning of the AMT content. Each reference set provides the ability to obtain a list of the concepts from each of the 'Notable concept' hierarchies without having to use the relationships or a transitive closure. Refer to Section 5.3.3 for other methods.

2.3.5 Relationship types

AMT v3 uses a variety of relationship types when connecting two concepts to represent different logical statements. This section covers the relationship types in use within AMT v3, where they are used and what they mean.

Note that the following sections describe the Stated Form of the AMT v3 data, as this is the form of the AMT v3 Beta. Note that expected cardinality of relationships expressed in the following sections will change in a Distribution Normal Form rendering of the same data. For more details refer to Section 3.4 Distribution Normal and Stated Forms.

2.3.5.1 Is a

116680003 |*Is a*| comes from the parent terminology, SNOMED CT. It is used to represent a supertype relationship, where the destination concept of an IS A relationship is a supertype of the source concept of an IS A relationship.

This means that the source or 'child' concept is a subtype of the 'parent' concept, and all statements that are true of the parent concept are therefore true of the child concept.

2.3.5.2 Has TPUU

30409011000036107 |*Has TPUU*| is an AMT relationship type used to indicate that a source CTPP or TPP (either concept type can be a source of this relationship type) contains the specified destination TPUU.

Note: HAS TPUU is a sub-relationship type of HAS MPUU, that is a narrower, more specific specialisation of the HAS MPUU relationship type between an MPP and MPUU.

Therefore each HAS TPUU relationship from a CTPP to a TPUU:

- specialises a HAS MPUU relationship on a parent MPP to an MPUU destination concept
- specifies a TPUU destination concept that is a subtype of the MPUU destination concept of the HAS MPUU relationship it specialises.

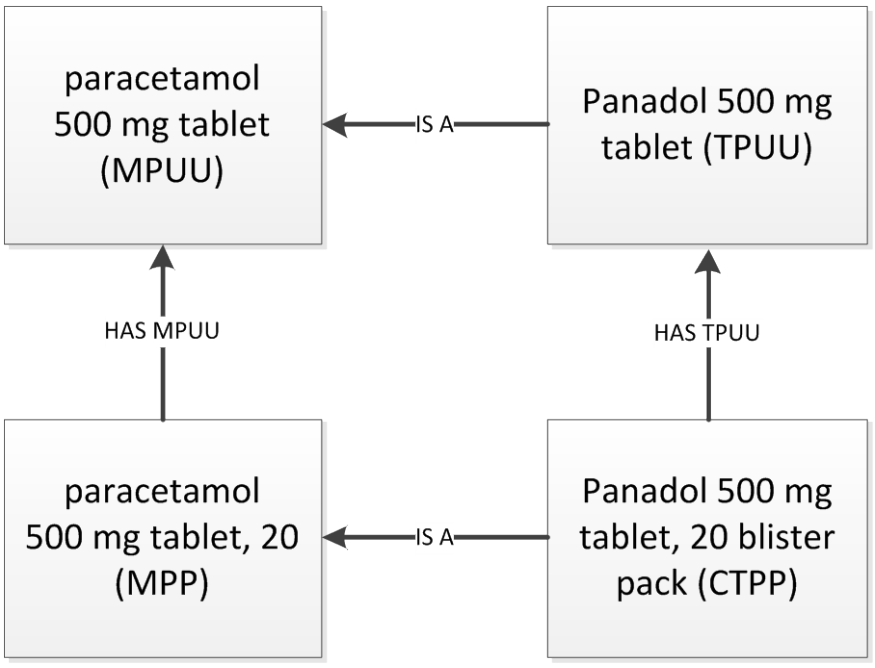


Figure 21: Example HAS MPUU *and HAS* TPUU *sub-role*

2.3.5.3 Has TP

700000101000036108 |*Has TP*| is an AMT relationship type that is used to specify the Trade Product concept representing a product's brand. Each CTPP and TPP concept is the source of a relationship of this type for this purpose.

2.3.5.4 Is modification of

30394011000036104 |*Is modification of*| is an AMT relationship type used on Substance concepts to indicate that the source Substance concept is a chemical compound including the destination Substance concept. See 2.3.3.1 Substance for more details.

This may be used to determine the moiety (base) for salt-based ingredients of AMT concepts.

2.3.5.5 Has TPP

30488011000036103 |*Has TPP*| is an AMT relationship type used to indicate the destination TPP concept is a generalisation of the source CTPP concept.

The following rules apply to the HAS TPP relationship between CTPP and TPP:

- All *composite* CTPPs (i.e. having multiple HAS COMPONENT PACK relationships to *component* CTPPs) have one HAS TPP relationship to a TPP concept.
- All CTPPs containing subpacks (i.e. having one or more HAS SUBPACK relationships to subpack CTPPs) have one HAS TPP relationship to a TPP concept.
- Most normal CTPPs, which do not contain and are not contained within other packs (are not the source or target of a HAS COMPONENT PACK or HAS SUBPACK relationship) have one HAS TPP relationship to a TPP concept. However there are some exceptions, most of which represent injections with diluents.

- Component CTPP concepts generally do not have a HAS TPP relationship to a TPP concept. There are some exceptions – any component CTPP which is also available individually outside of the composite pack(s) it is in will also have a single HAS TPP relationship to a TPP.
- All subpack CTPP concepts have one HAS TPP relationship to a TPP concept Contained Trade Product Pack for more on composite packs and subpacks.

2.3.5.6 Has container type

30465011000036106 |*Has container type*| is an AMT relationship type used to indicate a CTPP concept's container type where the source of the relationship is the CTPP concept and the destination of the relationship is the Container Type concept. Each CTPP concept has exactly one HAS CONTAINER TYPE relationship.

2.3.5.7 Has subpack

30454011000036104 |*Has subpack*| is an AMT-specific relationship type used to indicate the subpack MPP and CTPP concepts of MPP and CTPP concepts. The MPP or CTPP concept containing the subpack(s) is the source of the relationship, the 'subpack' CTPP or MPP is the destination of the relationship.

For more details on packs containing subpacks and subpacks refer to Section 2.3.2.3 Medicinal Product Pack and Section 2.3.2.6 Contained Trade Product Pack.

2.3.5.8 Has component pack

700000061000036106 |*Has component pack*| is an AMT-specific relationship type similar to HAS SUBPACK. HAS COMPONENT PACK is used to indicate where a destination MPP or CTPP concept is a component pack of an MPP or CTPP composite pack (respectively).

For more details on composite and component packs refer to Section 2.3.2.3 Medicinal Product Pack and Section 2.3.2.6 Contained Trade Product Pack.

2.3.5.9 Has intended active ingredient

700000081000036101 |*Has intended active ingredient*| is an AMT relationship used to indicate the ingredients of an MP or MPUU concept that are intended to have a therapeutic effect, i.e. ingredients that are not excipients etc.

Every MP concept will be the source of at least one relationship of this type, and the destination concept is a Substance concept that represents the ingredient. MPUU concepts have one HAS INTENDED ACTIVE INGREDIENT for each of its parent MP's HAS INTENDED ACTIVE INGREDIENT relationships with a matching destination Substance concept.

Note that in AMT v3 inert ingredients have been added to medicines using the HAS INTENDED ACTIVE INGREDIENT relationship type.

2.3.5.10 Has Australian BoSS

30364011000036101 |*has australian BoSS*| is an AMT-specific relationship type. It is used to represent the Australian Basis of Strength Substance for each HAS INTENDED ACTIVE INGREDIENT from an MPUU concept to a Substance concept. The pairing of HAS AUSTRALIAN BoSS and HAS INTENDED ACTIVE INGREDIENT relationships for each MPUU is represented using relationship groups – refer to Section 3.1.4.1 Relationship groups.

The HAS AUSTRALIAN BoSS relationship is used as part of the representation of the strength of the ingredient indicated by the paired HAS INTENDED ACTIVE INGREDIENT relationship. The other facet to the strength of the ingredient is a member of the *Strength reference set* that exists for each HAS AUSTRALIAN BoSS relationship and provides the quantity of the Substance concept targeted by the HAS AUSTRALIAN BoSS relationship. See Section 2.3.4.1 for more detail.

2.3.5.11 Has manufactured dose form

30523011000036108 |*Has manufactured dose form*| is an AMT relationship type used to indicate the manufactured dose form of an MPUU. Every MPUU concept is a source of exactly one relationship of this type, with the destination being a Form concept which indicates the MPUU's manufactured dose form.

2.3.5.12 Has unit of use

30548011000036101 |*Has unit of use*| is an AMT relationship type that indicates the Unit of Use concept that represents the unit of use for an MPUU.

Each MPUU concept has exactly one active relationship of this type, where the MPUU concept is the source and Unit of Use concept the destination of the relationship.

2.3.5.13 Has MPUU

30348011000036104 |*Has MPUU*| is an AMT relationship type used to indicate the MPUU concepts contained within an MPP concept. Each MPP concept is the source of one or more HAS MPUU relationship, where the destinations of the relationships represent the MPUU concepts contained within the MPP.

2.3.5.14 Has numerator units

700000091000036104 |*Has numerator units*| is an AMT-specific relationship type used to indicate the numerator Unit of measure concept for a composite Unit of measure concept.

For example, given the composite unit of measure |*mg/mL*|, this concept will be the source of a HAS NUMERATOR UNITS relationship to a destination |*mg*| concept.

A unit of measure concept will only be the source of a HAS NUMERATOR UNITS relationship if it is a composite unit of measure. If unit of measure concept is the source of a HAS NUMERATOR UNITS relationship, it will also be the source of a |*Has denominator units*| relationship. See also 2.3.5.15 Has denominator units.

See 2.3.3.3 Unit of measure for more details on composite units of measure.

2.3.5.15 Has denominator units

700000071000036103 |*Has denominator units*| is an AMT-specific relationship type with similar purpose to HAS NUMERATOR UNITS. It is used to indicate the denominator Unit of measure concept for a composite Unit of measure concept.

For example, given the composite unit of measure | *mg/mL* |, this concept will be the source of a HAS DENOMINATOR UNITS relationship to a destination | *mL* | concept.

A unit of measure concept will only be the source of a HAS DENOMINATOR UNITS relationship if it is a composite unit of measure. If unit of measure concept is the source of a HAS DENOMINATOR UNITS relationship, it will also be the source of *a* HAS NUMERATOR UNITS relationship. See also Section 2.3.5.14.

See Section 2.3.3.3 Unit of measure for more details on composite units of measure.

2.3.6 Relationship range and domain

The following sections show summarise the applicable domains and ranges for AMT v3 relationship types. This has been expressed using the same format used in Section 5.1 of the *SNOMED CT Technical Implementation Guide* [4].

2.3.6.1 Relationship types by domain

The following table shows the type relationships allowable for concepts broken down by the hierarchies in AMT v3. Concepts may belong to multiple hierarchies in AMT v3, and in this case the union of the relationship types allowable for each hierarchy are allowed.

Hierarchy	Relationship type
Composite unit of measure	Has numerator units
	Has denominator units
Medicinal Product	Has intended active ingredient
Substance	Is modification of
Medicinal Product Unit of Use	Has intended active ingredient
	Has Australian BoSS
	Has unit of use
	Has manufactured dose form
Medicinal Product Pack	Has MPUU
	Has subpack
	Has component pack
Trade Product Unit of Use	Has TP
Trade Product Pack	Has TP
Containered Trade Product Pack	Has TP
	Has TPUU
	Has TPP
	Has subpack
	Has component pack
	Has container type

2.3.6.2 Allowable ranges

The following table shows the possible range (values) for each relationship type.

Relationship Type	Range
Has numerator units	Unit of measure
Has denominator units	Unit of measure
Has intended active ingredient	Substance

Is modification of	Substance
Has Australian BoSS	Substance
Has unit of use	Unit of Use
Has manufactured dose form	Form
Has MPUU	Medicinal Product Unit of Use
Has subpack	Medicinal Product Unit of Use Containered Trade Product Pack
Has component pack	Medicinal Product Unit of Use Containered Trade Product Pack
Has container type	Container type
Has TP	Trade Product
Has TPP	Trade Product Pack
Has TPUU	Trade Product Unit of Use

3 Distribution form of AMT v3

AMT v3 is distributed as an RF2 release containing the following concrete domain reference sets which also form part of the semantics of the AMT content:

- *Strength reference set*
- *Unit of use size reference set*
- *Unit of use quantity reference set*
- *Subpack quantity reference set*

The following sections provide an overview of AMT v3's distribution form and format, with references to the *SNOMED CT Technical Implementation Guide* [4] where appropriate.

3.1 Overview of Release Format 2 (RF2)

The IHTSDO currently has two release formats for SNOMED CT: Release Format 1 (RF1) and Release Format 2 (RF2). RF1 is the original format SNOMED CT was released in, and has been superseded by the newer RF2 in recent years.

AMT v3 relies on features supported by RF2 but not RF1, therefore AMT v3 is distributed in RF2 only.

For the full definition of RF2 refer to the *SNOMED CT Technical Implementation Guide* [IHTSDO2012] Section 5.2 Release Format 2, however this section provides an overview of key elements of RF2.

3.1.1 Elements of an RF2 release

An RF2 release consists of four core files as depicted in Figure 22 Core RF2 files:

- Concepts
- Descriptions
- Relationships
- Identifiers[11]

Each of these files is a tab delimited UTF-8 text file with DOS line termination (<carriage return> followed by <line feed>), and together these files provide a relational distribution form for SNOMED CT components.

Each core component (Concepts, Descriptions, Relationships, and Identifiers) are distributed in their own file each with a different format. The columns for each file format are covered in more detail in the following sections, however it is important to note from Figure 22 Core RF2 files:

- Concept identifiers are foreign keys in the relationships file representing the source and destination of relationships.
- Concepts have two or more descriptions in the descriptions file. These descriptions file rows use the concept's identifier as a foreign key.
- All four files have a foreign key to a module identifier, which is a concept identifier referencing the module to which the component belongs – refer to Section 3.6 Modules.

[11] See Section 5.4.3 of *SNOMED CT Technical Implementation Guide* [4] for details of these files: http://www.ihtsdo.org/fileadmin/user_upload/doc/en_us/tig.html?t=trg2main_format.

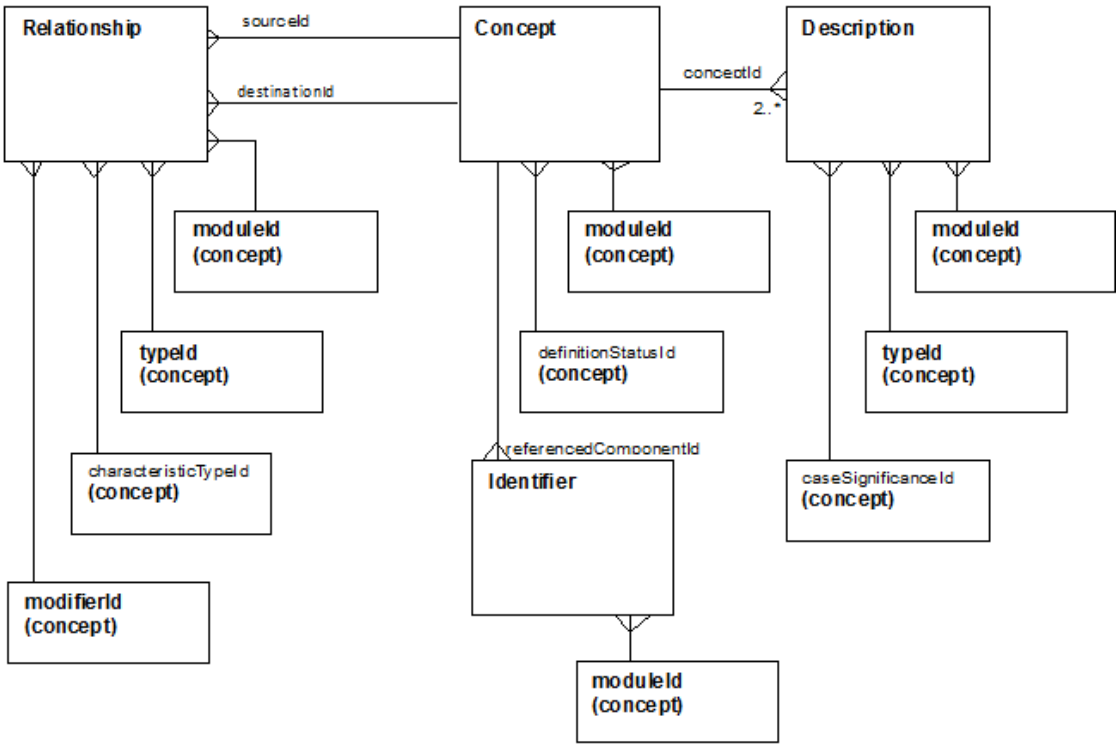


Figure 22 Core RF2 files

3.1.2 Concepts

Concepts are at the core of RF2 and AMT – refer to Section 2.2.1 Concepts for more detail on their meaning and purpose.

RF2 provides a file containing concepts which has the following format

Field	Data type	Immutable	Purpose
id	SCTID	Y	Uniquely identifies the concept.
effectiveTime	Time	N	Specifies the inclusive date at which the component version's state became the then current valid state of the component See 3.7 History Tracking Mechanism for more details.
active	Boolean	N	Specifies whether the concept 's state was active or inactive from the nominal release date specified by the effectiveTime See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	N	Identifies the concept version's module. Set to a descendant of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.

Field	Data type	Immutable	Purpose
definitionStatusId	SCTID	N	Specifies if the concept version is primitive or fully defined. Set to a child of |*Definition status*| in the metadata hierarchy. Only necessary if using the description logic definitions of the AMT v3 content, refer to Section 2.2.5 Defined and Primitive concepts

Refer to Section 5.4.3.1 Concept file of the *SNOMED CT Technical Implementation Guide* [4] for more details.

3.1.3 Descriptions

Each concept must have one, and only one, active

- Fully Specified Name
- and Preferred Term

Additionally other Synonyms may be provided for a concept.

In RF2, and hence AMT v3, there are currently only two types (refer to typeId below) of descriptions – Fully Specified Names and Synonyms. Preferred Terms are Synonym type descriptions which have been identified by a Language reference set to be preferred in a particular context. To find AMT v3 Preferred Terms it is necessary to use and combine both the Descriptions file and the Language reference set file. Refer to 3.1.7 Language reference sets for more detail on this mechanism.

Field	Data type	Immutable	Purpose
id	SCTID	Y	Uniquely identifies the description.
effectiveTime	Time	N	Specifies the inclusive date at which the component version's state became the then current valid state of the component See 3.7 History Tracking Mechanism for more details.
active	Boolean	N	Specifies whether the description 's state was active or inactive from the nominal release date specified by the effectiveTime. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	N	Identifies the description version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.

Field	Data type	Immutable	Purpose
conceptId	SCTID	Y	Identifies the concept to which this description belongs. Set to an Identifier of a concept in the |*SNOMED CT Concept*| hierarchy within the Concept file. Note that versions of descriptions and concepts don't belong to each other. Which version of any given description is combined with which version of its owning concept depends on the point in time at which they are accessed.
languageCode	String	Y	Specifies the language of the description text using the two character ISO-639-1 code. Note that this specifies a language level only, not a dialect or country code. In AMT v3 this is always 'en' for English. The *Australian english language reference set* provides the Australian preferred terms, essentially declaring them 'en-AU'.
typeId	SCTID	Y	Identifies whether the description is an FSN, Synonym or other description type. This field is set to a child of |*Description type*| in the metadata hierarchy.
term	String	N	The description version's text value, represented in UTF-8 encoding.
caseSignificanceId	SCTID	N	Identifies the concept enumeration value that represents the case significance of this description version. For example, the term may be completely case sensitive, case insensitive, initial letter case sensitive. This field will be set to a child of |*Case significance*| within the metadata hierarchy.

The *Description format reference set* (Section 3.1.9.2) specifies the lengths and formats of descriptions in the release. AMT v3 uses only 'plain text' descriptions with a maximum length of 2048 characters – note this is longer than SNOMED CT's default lengths for standard Fully Specified Names and Synonyms. See Section 7.6 Field length for more details.

Refer to Section 5.4.3.2 Descriptions file of the *SNOMED CT Technical Implementation Guide* [4] for more details.

3.1.4 Relationships

The relationships file in RF2 contains SNOMED CT relationships which provide a unidirectional connection between two concepts and a relationship type. The combination of this triplet provides a logical statement about the concept that is the 'source' of the relationship. See 2.2.3 Relationships for more detail on relationships' purpose and meaning.

Field	Data type	Immutable	Purpose
id	SCTID	Y	Uniquely identifies the relationship.
effectiveTime	Time	N	Specifies the inclusive date at which the component version's state became the then current valid state of the component See 3.7 History Tracking Mechanism for more details.
active	Boolean	N	Specifies whether the relationship 's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	N	Identifies the relationship version's module. Set to a child of |*module*| within the metadata hierarchy. See 3.6 Modules for more details.
sourceId	SCTID	Y	Identifies the source concept of the relationship version, i.e., the concept the relationship version emanates from. Set to an Identifier of a concept in the |*SNOMED CT concept*| hierarchy within the 'Concept' file.
destinationId	SCTID	Y	Identifies the concept that is the destination of the relationship version. Set to an Identifier of a concept in the |*SNOMED CT concept*| hierarchy within the 'Concept' file.
relationshipGroup	Integer	Y	Groups together relationship versions that are part of a logically associated *relationship group*. All active Relationship records with the same *relationshipGroup* number and sourceId are grouped in this way. See 3.1.4.1 Relationship groups for details relevant to AMT v3
typeId	SCTID	Y	A concept enumeration value from the metadata hierarchy that identifies the semantic type of the relationship version. For example |*Is a*|, or |*associated morphology*|. Section 2.3.5 Relationship types contains the possible values for this field in AMT v3 and their meaning
characteristicTypeId	SCTID	Y	A concept enumeration value that identifies the characteristic type of the relationship version (i.e. whether the relationship version is defining, qualifying, etc.) This field is set to a descendant of |*characteristic type*| within the metadata hierarchy.
modifierId	SCTID	Y	A concept enumeration value that identifies the type of Description Logic (DL) restriction (some, all, etc.). Set to a child of |*modifier*| within the metadata hierarchy.

Refer to Section 5.4.3.3 Relationships file of the *SNOMED CT Technical Implementation Guide* [4] for more details.

3.1.4.1 Relationship groups

SNOMED CT uses 'relationship groups' (referred to as 'role groups') to group together relationships that are relative to each other and must be read as a collective.

This mechanism is used in AMT v3 to group Has intended active ingredient and Has Australian BoSS relationships for MPUU concepts, and is important for multi-ingredient MPUUs.

Consider the following example Has intended active ingredient and Has Australian BoSS relationships for the MPUU concept |*paracetamol 500 mg + codeine 10 mg tablet*|:

Relationship type	Destination	Relationship Group
HAS INTENDED ACTIVE INGREDIENT	paracetamol	1234
HAS INTENDED ACTIVE INGREDIENT	codeine	9876
HAS AUSTRALIAN BoSS	paracetamol	1234
HAS AUSTRALIAN BoSS	codeine phosphate	9876

Without the 'Relationship Group' column, it is impossible for a machine to determine which HAS AUSTRALIAN BoSS relationship relates to which HAS INTENDED ACTIVE INGREDIENT relationship.

Relationship group values are:

- Zero for ungrouped, meaning all relationships with '0' as their relationship group are not grouped with any other relationship and are to be treated individually.
- Non-zero for grouped relationships. A relationship with a non-zero relationship group is grouped with other relationships with the same relationship group value and source concept id.

Relationship group values in themselves are meaningless other than the grouping effect they have. That is there is no significance to a relationship group value of 1 as opposed to 2.

Relationship group values are only meaningful within the context of relationships with the same source concept identifier.

3.1.5 Identifiers

The RF2 Identifiers file provides a mechanism to associate one or more coreferent identifiers with another RF2 component, such as a concept, description, or relationship.

AMT v3 uses this mechanism to represent the original AMT v2 identifiers for AMT v2 components that have been replaced by their SNOMED CT International and SNOMED CT-AU counterparts. In these cases the identifiers represented in AMT v3 concept, description and relationship files will reflect the SNOMED CT International and SNOMED CT-AU identifiers, and the AMT v2 identifiers will be present in the RF2 Identifiers file referencing the new component. This includes both SCTIDs and UUIDs previously published in AMT v2 for these components.

The Identifiers file format is shown in the table below.

Field	Data type	Immutable	Purpose
identifierSchemeId	SCTID	Y	Identifier of the concept enumeration value from the metadata hierarchy that represents the scheme to which the Identifier value belongs. Set to a descendant of |*Identifier scheme*| within the metadata hierarchy. In AMT v3 this will be: • 900000000000002006 |*SNOMED CT UUID*| or • 900000000000294009 |*SNOMED CT integer ID*| depending upon whether the alternateIdentifier is an AMT v2 UUID or SCTID.
alternateIdentifier	String	Y	String representation of the alternateIdentifier in its native scheme. In AMT v3 this will be an AMT v2 UUID or SCTID
effectiveTime	Time	N	Specifies the inclusive date at which the alternateIdentifier was associated with the SNOMED CT component. See 3.7 History Tracking Mechanism for more details.
active	Boolean	N	Specifies whether the association was active or inactive from the point in time specified by the effectiveTime. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	N	Identifies the source module that this association was created in. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
referencedComponentId	SCTID	Y	Uniquely identifies the SNOMED CT component with which the alternateIdentifier is associated.

Refer to Section 5.4.3.3 Identifiers file of the *SNOMED CT Technical Implementation Guide* [4] for more details.

3.1.6 Attribute value reference sets

AMT v3 is released with seven Attribute value reference sets, one for each of the 'Notable concepts' – refer to Section 2.3.4.5 'Notable concept' reference sets which explains the purpose of these specific reference sets.

These reference sets adhere to the Attribute value reference set pattern defined by the IHTSDO in the RF2 specification. Details of this pattern can be found in Section 5.5.2.5 Attribute Value Reference Set of the *SNOMED CT Technical Implementation Guide* [4], however an overview of the data structure and expected values in AMT v3 is provided below.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	Set to a child of |*Attribute value type*| in the metadata hierarchy. In AMT v3 this will identify which of the seven reference sets each row is a member of. The values used are: • 929360021000036102 |*Trade product reference set*| • 929360031000036100 |*Trade product unit of use reference set*| • 929360041000036105 |*Trade product pack reference set*| • 929360051000036108 |*Contained trade product pack reference set*| • 929360061000036106 |*Medicinal product reference set*| • 929360071000036103 |*Medicinal product unit of use reference set*| • 929360081000036101 |*Medicinal product pack reference set*|
referencedComponentId	SCTID	A reference to the SNOMED CT component being tagged with a value. In AMT v3 this will be the concept this membership row is referring to and including in the reference set.
valueId	SCTID	Set to a grandchild of |*Attribute value*|. In AMT v3 this will always be the value 32570031000036104 |*Normal member*|

3.1.7 Language reference sets

Language reference sets in RF2 are used to express local language preferences and dialectic differences across descriptions from a parent language; for example British English preferences versus United States English preferences.

RF2 also only provides two types of descriptions:

- Fully Specified Names – the true meaning of the concept.
- Synonyms – other names for a concept useful in a variety of settings.

Language reference sets in RF2 annotate RF2 Synonym descriptions with one of three mutually exclusive values to provide localisation:

Preferred Indicates that Synonyms annotated with this value are the preferred way of describing the concept (also known as the 'Preferred Term'). A Language reference set is required to have only one active 'preferred' Synonym for each concept.

Acceptable Indicates that Synonyms annotated with this value are not the preferred way of describing a concept, however are acceptable in the Language reference set's context. A Language reference set may refer to zero or more of a concept's descriptions with this value.

Not Acceptable Indicates that the Synonym is not an acceptable way to describe the concept in this Language reference set's context. Language reference sets do not reference Synonyms with this value, rather for brevity they simply do not reference Synonyms that are unacceptable.

Figure 23 shows an example taken from SNOMED CT-AU, showing how the *Australian dialect reference set* expresses that:

- 'Paracetamol' is preferred (green) and 'Acetaminophen' is not acceptable (red).
- 'Appendicectomy' is preferred (green), while 'Appendectomy' is acceptable (blue), as is 'Excision of appendix'.

That is, the Language reference set can be joined with the content of the Descriptions file to determine the preferred and acceptable Synonyms for a concept or concepts.

Note that Fully Specified Names are not referenced – they are always acceptable and preference is irrelevant.

Note also that columns have been omitted from the example for brevity.

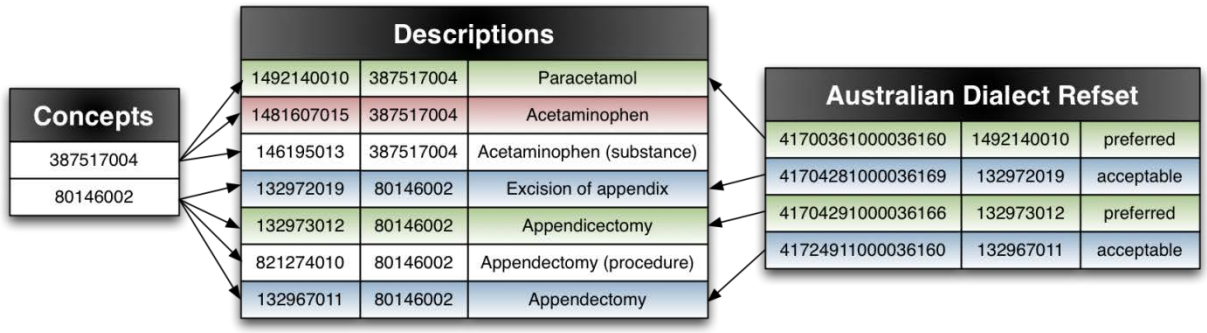


Figure 23 Example of a Language reference set

The data structure of a Language reference set is as follows.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.

Field	Data type	Purpose
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	A descendant of |*Language type*| in the metadata hierarchy. In AMT v3 this field will have the value 32570271000036106 |*Australian dialect reference set*|.
referencedComponentId	SCTID	A reference to the Description included in the Language reference set.
acceptabilityId	SCTID	A descendant of |*Acceptability*| in the metadata hierarchy. In AMT v3 this will be one of: • 900000000000548007 |*Preferred*|; or • 900000000000549004 |*Acceptable*|.

3.1.8 Association reference sets

RF2 also provides the Association reference set pattern – refer to Section 5.5.2.11 Association Reference Sets of the *SNOMED CT Technical Implementation Guide* [4].

This reference set pattern is used in AMT v3 to provide a *Historical association reference set*, 900000000000526001 |*REPLACED BY association reference set*| – refer to Section 7.4.2.3 Historical Association Reference Sets.

The *REPLACED BY association reference set* refers to retired components, and provides the identifier of the replacement component for each referenced component.

The data structure is outlined below.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.

Field	Data type	Purpose
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	A descendant of |*Association type*| in the metadata hierarchy. In AMT v3 this will be the value 900000000000526001 |*REPLACED BY association reference set*|
referencedComponentId	SCTID	A reference to the source component of the association. In the context of AMT v3 this will be the identifier of the AMT v3 concept which is now inactive and has been replaced.
targetComponentId	SCTID	A reference to the destination component of the association. In the context of AMT v3 this will be the identifier of the component replacing the retired component.

3.1.9 Metadata reference sets

AMT v3 as an RF2 release also contains a number of metadata reference sets described in the following sections. These reference sets provide metadata about the release format and contained content structure.

3.1.9.1 Module dependency reference set

The *Module dependency reference set* is used to describe the:

- modules included in the release files; and
- dependencies between the modules in the release files.

This type of reference set is described in Section 7.4.2.4, and specified in Section 5.5.2.12 of the *SNOMED CT Technical Implementation Guide* [4]. The following passage is of particular relevance here:

> *The rows in this Reference Set that originate in a given module (identified by moduleId) indicate a dependency on the module identified by the referencedComponentId. The two string values each contain dates that indicate the version of source module and the required version of the module on which it depends.* (*SNOMED CT Technical Implementation Guide* [4], Section 5.5.2.12.)

The *Module dependency reference set* from the AMT v3 Beta data has been provided in the table below as an example. The identifiers in this table have been replaced by preferred terms for readability, and some columns have been removed for brevity.

Module name	Referenced component name	Source effective time	Target effective time
SNOMED CT core	SNOMED CT model component	31/07/2012	31/07/2012
Australian Medicines Terminology module	SNOMED Clinical Terms Australian extension	31/12/2012	30/11/2012

SNOMED Clinical Terms Australian extension	SNOMED CT core	30/11/2012	31/07/2012

The data columns used in the *Module dependency reference set* are as follows.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	A reference to the |*Module dependency*| concept in the metadata hierarchy.
referencedComponentId	SCTID	A reference to the module that this module is dependent on, a descendant of |*Module*| in the metadata hierarchy.
sourceEffectiveTime	String	The effective time of the source module. This allows a specific module version to be selected as having a dependency. The effectiveTime must match exactly.
targetEffectiveTime	String	The effective time of the target module. This allows a specific module version to be selected as being the subject of a dependency. The effectiveTime must match exactly.

For more details on modules refer to Section 3.6 Modules.

3.1.9.2 Description format reference set

The *Description format reference set* is a reference set that provides format and maximum length information for description types used in a release.

AMT v3 contains a *Description format reference set* that contains the following information:

Description type	Format	Maximum length
Fully Specified Name	Plain text	2048 bytes
Synonym	Plain text	2048 bytes

If implementation cannot be achieved, please contact NCTIS by emailing terminologies@nehta.gov.au. Refer to Section 7.6 Field length for more details.

Section 5.5.2.13 Description Format Reference Set of the *SNOMED CT Technical Implementation Guide* [4] specifies Description format reference sets. The columns used in the reference set are provided below.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	Set to the |*Description format*| reference set concept in the metadata hierarchy.
referencedComponentId	SCTID	A reference to a child of |*Description type*| in the metadata hierarchy
descriptionFormat	SCTID	A reference to a child of | *Description format* | reference set attribute concept in the metadata hierarchy.
descriptionLength	Integer	The maximum length in bytes for descriptions of this description type.

3.1.9.3 Reference set descriptor

In RF2 the *Reference set descriptor* is a reference set that holds metadata used to define the following attributes of reference sets included in a release:

- the order of appearance of additional attributes (other than those mandatory for a reference set);
- the name and purpose of the additional attributes; and
- the data types for the additional attributes.

The *Reference set descriptor* is specified in Section 5.5.2.2 Reference Set Descriptor of the *SNOMED CT Technical Implementation Guide* [4].

AMT v3 provides a *Reference set descriptor reference set;* its columns are described in the table below. An example from the reference set descriptor from AMT v3 is included in Appendix B.

Field	Data type	Purpose
id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.

Field	Data type	Purpose
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SCTID	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.
refSetId	SCTID	Indicates that this is row is part of a 'reference set descriptor'. Set to the 900000000000456007 |*Reference set descriptor reference set (foundation metadata concept)*|
referencedComponentId	SCTID	Identifies the reference set (or type of reference set) that is specified by this descriptor. Set to a descendant of 900000000000455006 |*Reference set (foundation metadata concept)*| in the metadata hierarchy.
attributeDescription	SCTID	Specifies the name of an attribute that is used in the reference set to which this descriptor applies. Set to a descendant of 900000000000457003 |*Reference set attribute (foundation metadata concept)*| in the metadata hierarchy, that describes the additional attribute extending the reference set.
attributeType	SCTID	Specifies the data type of this attribute in the reference set to which this descriptor applies. Set to a descendant of 900000000000459000 |*Attribute type (foundation metadata concept)*| in the metadata hierarchy, that describes the type of the additional attribute extending the reference set.
attributeOrder	Integer	Specifies the position of this attribute in the reference set to which this descriptor applies. A zero value identifies the referencedComponentId within the reference set. Other values specify an additional attributes by its position relative to the referencedComponentId. An unsigned integer, providing an ordering for the additional attributes extending the reference set.

3.2 Concrete domains and data type properties

3.2.1 What are concrete domains?

Concrete domains and data type properties are a way of including concrete data values as defining attributes of concepts. The name “concrete domains” stems from research in the late 1970s on mathematical descriptions of the semantics of programming languages in an effort to distinguish data from semantic domains. The term “concrete domains” is commonly used when describing these capabilities of description logics.

Put simply, while the relationships in SNOMED CT (for AMT specifically the relationships listed in section 2.3.5) allow a concept to have an attribute with a concept value, data type properties provide attributes which have a concrete data value. That is an attribute value from a concrete domain such as the number “5” as opposed to a concept value such as the concept |*paracetamol*|.

The domains available are really any continuum of concrete data values, such as numbers, time, dates, colours, strings etc. OWL, for example permits the use of any XML Schema Datatype.

AMT v3 only uses the concrete domains of real numbers and integers. These are used to represent the defining numeric attributes of AMT v3 concepts, specifically:

- Strength – e.g. the concentration of an ingredient;
- Unit size – e.g. the volume of a medicine in an ampoule;
- Unit quantity (in a pack) – e.g. the number of tablets in a pack; and
- Subpack quantity – e.g. the number of subpacks in a pack.

3.2.2 Concrete domains in SNOMED CT

SNOMED CT currently has no support for concrete domains, and only supports a very limited set of description logic features known as EL+.

A proposal has been under development since 2010, defining both the semantic implications and RF2 representation of the introduction of this description logic feature into SNOMED CT’s currently limited set. This proposal is currently pending final updates and review prior to a request for Draft For Trial Use approval.

This proposal has been developed specifically in response to needs for increased expressivity to adequately represent the IHTSDO’s new Pharmacy Model, to be used to remodel SNOMED CT’s medicines content. The new IHTSDO Pharmacy Model uses concrete domains as specified to represent ingredient strengths in medicines.

While the current scope in SNOMED CT is limited to the Pharmacy Model, the proposal as specified is very general and may be applied in future in other SNOMED CT content domains.

The proposal is documented in *RF2 Specification Change Request: Addition of Concrete Domains* [2]. This document includes:

- the specification of the reference set pattern to be used (RF2 data structure);
- implication for description logic semantics;
- handling of units of measure;
- metadata additions; and
- changes to diagramming notation.

3.2.3 Concrete domains in AMT v3

AMT v3 implements concrete domains as specified in the proposal *RF2 Specification Change Request: Addition of Concrete Domains* [2]. This implementation provides:

- defining numeric attributes for AMT v3 concepts modelled within the description logic definition of the concepts; and
- type safe machine-processable access to these numeric attributes without needing to use description logic.

The following AMT v3 reference sets (which conform to the *RF2 Specification Change Request: Addition of Concrete Domains* [2]) provide these numeric attributes for AMT and are known as 'Concrete domain reference sets':

- *Strength reference set* (Section 2.3.4.1)
- *Unit of use size reference set* (Section 2.3.4.2)
- *Unit of use quantity reference set* (Section 2.3.4.3)
- *Subpack quantity reference set* (Section 2.3.4.4)

If only type safe access to these numeric attributes is required (without use of the full description logic definition) it is not necessary to read the entirety of *RF2 Specification Change Request: Addition of Concrete Domains* [2]. The information relevant to most implementations has been repeated in Section 3.2.4 below as a convenience. Note this focuses mainly on structural form of the data. The meaning of the Concrete domain reference sets included in AMT v3 is discussed in Sections 2.3.4.1 to 2.3.4.4.

For details of the description logic implications and semantics please refer to the *RF2 Specification Change Request: Addition of Concrete Domains* [2].

3.2.4 Structure

The table below shows the columns present in the Concrete Domain Reference Sets provided with AMT v3.

Field	Data type	Purpose
Id	UUID	A 128 bit unsigned integer, uniquely identifying the reference set member.
effectiveTime	Time	Specifies the inclusive date at which this change becomes effective. See 3.7 History Tracking Mechanism for more details.
active	Boolean	Specifies whether the member's state was active or inactive from the nominal release date specified by the effectiveTime field. See 3.7 History Tracking Mechanism for more details.
moduleId	SctId	Identifies the member version's module. Set to a child of |*Module*| within the metadata hierarchy. See 3.6 Modules for more details.

Field	Data type	Purpose
refSetId	SctId	Set to a child of |*Concrete domain type*| in the metadata hierarchy. In the context of AMT v3 this will either be: • 700000111000036105 |*Strength reference set*| • 700000131000036101 |*Unit of use quantity reference set*| • 700000141000036106 |*Unit of use size reference set*| • 700000121000036103 |*Subpack quantity reference set*|
referencedComponentId	SctId	The component with which the concrete value is to be associated. In the context of AMT v3 this is always a relationship identifier. Refer to Sections 2.3.4.1, 2.3.4.2, 2.3.4.3 and 2.3.4.4 for details of the specific relationship types, sources and destinations used in each case.
unitId	SctId	A child of |*Unit*| (SctId 258666001). In AMT v3 this will always be a type of AMT[12] |*Unit of measure*| concept.
operatorId	SctId	A child of |*Operator id value*| in the metadata hierarchy. Currently in AMT v3 this field will always hold the value 700000051000036108 |*Equal To*|. Other values may be introduced into AMT in future to support ranges if necessary (such as 'greater than' and 'less than').
value	String	The concrete value to be associated with the referenced component. The precise data type presented in this string is dependent upon the particular subtype of the |*Concrete domain type*| concept, which will also be reflected in the specific reference set's descriptor. For AMT v3 this value will be a double precision floating number for the *Strength reference set*, *Unit of use size reference set*, and *Unit of use quantity reference set*. For the *Subpack quantity reference set* this value will be an integer.

3.2.5 Units of measure

Units are represented as specified in Section 3.2.4 using the 'unitId' field to reference AMT Unit of measure concepts, for example 'milligram'.

Where necessary for concentration and rate based strengths, composite units of measure are used, for example 'milligrams per millilitre'[13]. Where a composite unit of measure is used, the value provided is always in terms of a denominator of one. For example '2 mg per 2 mL' is expressed in the *Strength reference set* as '1 mg / mL'. This allows much easier comparison of strength values.

[12] As distinct from being a child of the SNOMED CT |*Unit of measure*| concept.
[13] Note that composite units of measure are always used for the *Strength reference set*.

There is also a special unit of measure called ‘each’. This unit means that the value provided in the *Concrete domain reference set* is simply a count or number of the items the reference set is referring to. For example this is used in the *Unit of use quantity reference set* where MPUUs included in an MPP have a physically countable dose form, such as tablet or capsule.

These issues of unit of measure are all discussed in Section 2.3.3.3 Unit of measure, 2.3.5.14 Has numerator units, and 2.3.5.15 Has denominator units.

3.2.6 Units conversions

Currently the representation of quantities in the concrete domain specification (*RF2 Specification Change Request: Addition of Concrete Domains* [2]) does not cater for unit conversions. In order to ensure all strengths are machine comparable without unit conversion, this specification instructs that all quantities of a single concept use the same units. For example, to determine that ‘1 gram of paracetamol’ is the same as ‘1000 milligrams of paracetamol’ without unit conversion requires that |*paracetamol*| is always referred to using the same unit of measure.

Note that this normalisation of units of measure does not affect descriptions. Descriptions may represent the quantity in the most useful form to humans that is equivalent to the machine representation in the Concrete domain reference sets.

Unit conversion is possible outside of SNOMED CT’s description logic (i.e. outside a classifier), however unit conversion factors are not present in AMT or SNOMED CT data. SNOMED CT International is working towards publishing a map between SNOMED CT Units of Measure and UCUM units to resolve this issue.

AMT v3 Beta does not implement this normalisation of units, and simply follows the AMT v3 editorial rules for descriptions. Consistent units of measure are not used across all strengths of an individual ingredient in the AMT v3 Beta – e.g. according to editorial rules 500 mg of an ingredient is expressed as ‘500 mg’ and 1000 mg of the same ingredient in a different product is expressed as ‘1 g’. Unit conversions must be performed in order to compare strengths.

3.3 RF2 distribution types

RF2’s data structure and history tracking mechanism (refer to Section 3.7 History Tracking Mechanism) enables three different types of release:

Release Type	Description
Full	The files representing each type of component contain **every** version of every component ever released.
Snapshot	The files representing each type of component contain **one** version of every component released up to the time of the snapshot. The version of each component contained in a snapshot is the most recent version of that component at the time of the snapshot.
Delta	The files representing each type of component contain only component versions created since the previous release. Each component version in a delta release represents either a new component or a change to an existing component.

To demonstrate, Figure 24 Example of Full, Snapshot and Delta formats shows a sample of Full content expressed as Snapshot and Delta as well.

Concepts - Full		
138875005	1	20070131
138875005	0	20090731
138875005	1	20100131
404684003	1	20030131
404684003	1	20090731
404684003	0	20100131
162744006	1	20020131
162744006	1	20070731
3415004	1	20100131

Concepts - Snapshot		
138875005	1	20100131
404684003	0	20100131
162744006	1	20070731
3415004	1	20100131

Concepts - Delta		
138875005	1	20100131
404684003	0	20100131
3415004	1	20100131

Figure 24 Example of Full, Snapshot and Delta formats

RF1 SNOMED CT Releases and AMT v2 were always released in Snapshot format only. The AMT v3 Beta is released in Snapshot format also.

To be RF2 compliant, AMT v3 production releases will be published using the Full release type, but may also be made available in Snapshot and Delta forms.

These forms are useful in different contexts. For example the Snapshot form is easiest to query, however, data updates typically require loading a new Snapshot release. Whereas queries written against the Full form are more complex, however data updates are simplified to appending the next Delta release to the existing Full data. For more details of these considerations refer to Section 4.3 Designing for data updates.

3.4 Distribution Normal and Stated Forms

The AMT v3 Beta is published in ‘Stated Form’, however production releases will be published in ‘Distribution Normal Form’. It is therefore important to understand the following sections as any designs or prototype implementations derived from the Beta release may be impacted.

Specific impacts for analysis are listed in Section 3.4.4, however generally the AMT v3 Beta as ‘Stated Form’ does not contain all the inferable relationships for each concept. Omitted relationships may include both IS A and other relationship types.

SNOMED CT is always distributed in what is known as Distribution Normal Form, which is the definitive release. Additionally SNOMED CT is distributed by the IHTSDO in the Stated Form which was used to create the Distribution Normal Form, however the Stated Form is **not** the definitive release form.

The IHTSDO mandates that publication of SNOMED CT and extensions is in Distribution Normal Form, and distribution in other Inferred Forms or Stated Form is optional.

The following sections explain the difference between Stated Form, Inferred Forms and Distribution Normal Forms.

3.4.1 Stated and Inferred Forms

SNOMED CT and its extension terminologies (such as AMT v3) have an ontological foundation. This formalised representation of knowledge allows reasoning, and machine processes can be used to infer facts from a collection of stated facts, represented by relationships between concepts in SNOMED CT terminologies.

The collection of relationships representing the statements about each concept in the terminology can take a number of different forms, which in essence break down into two categories

Stated Form This is the set of statements authored by human authors asserting facts about the concepts in the terminology.

Inferred Forms These forms include the statements that can be inferred from the collection of statements made in the Stated Form. This is usually created by a computer using software known as a description logic classifier or reasoner.

As a simple example Figure 25 shows the addition of an inferred relationship – that a hybrid car is a type of car because it has the same definition as a petrol-driven car, with an additional relationship that the petrol-driven car doesn't have.

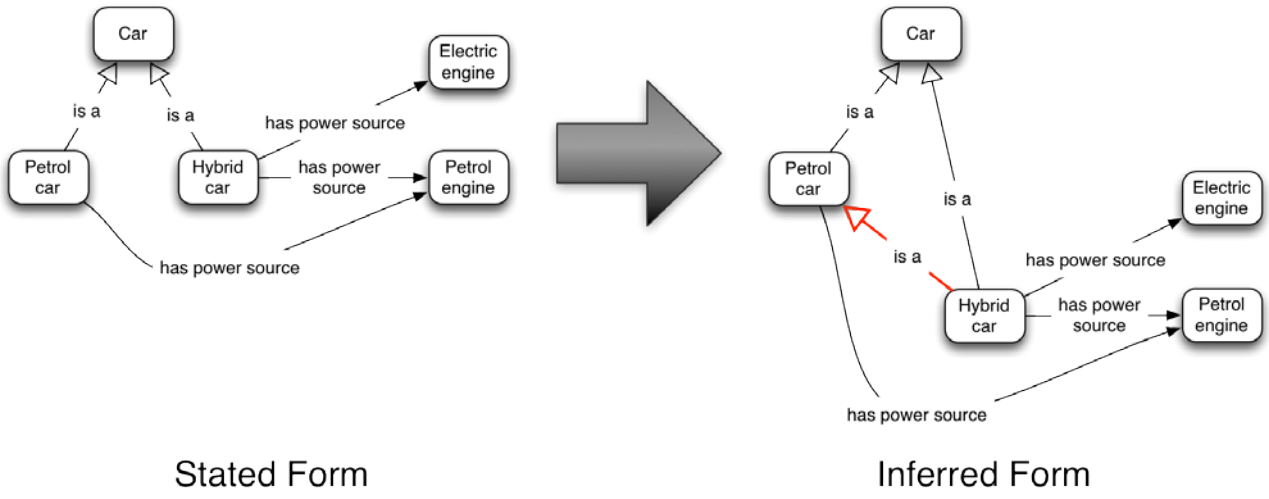


Figure 25 Adding inferred relationships

3.4.2 Normalised forms

Inferred Forms can be further broken down into 'normalised' forms, where rules are used to govern the relationships included from the entire set of stated and inferred relationships. Normal forms provide a consistent and dependable structure to code against. The two extremities of these forms are

- including every possible inferable relationship, which would include many redundant relationships; and
- reducing the relationships to the minimum possible number without losing or changing semantics.

3.4.3 Distribution Normal Form

The IHTSDO has a defined 'Distribution Normal Form' for SNOMED CT, which is an Inferred Form as discussed above.

The Distribution Normal Form is created by:

1. Using a description logic classifier on the Stated Form to determine the additional inferable relationships.
2. 'Normalising' the resulting full set of stated and inferred relationships to:
    - limit the IS A relationships to proximal supertypes (described further below); and
    - remove any redundant inferred relationships (described further below).

The IHTSDO chose this form to distribute because:

- It provides non-redundant inferred and inherited relationships, so consumers of this form do not need to calculate them (i.e. have and use a description logic classifier as in point 1 above).
- It minimises the number of IS A relationships to keep the format as compact as possible for distribution. The opposite approach would be to distribute the full transitive closure, which is very large. The benefit of a more compact form, combined with the ease of generating the transitive closure[14] means that the minimal set of IS A relationships is felt to be easier to consume. See Section 3.5 Transitive Closure for more details on what a transitive closure is, and how to create it.

The following rules are applied to create this 'normalised' form.

Proximal inferred supertypes – this rule applies to the combined stated and inferred IS A relationships, and means that only IS A relationships to 'proximal supertypes' of a concept are included. Other IS A relationships to supertypes of the concept which can be reached by traversing multiple IS A relationships are omitted. This reduces the IS A relationships to the minimal set of inferable parents, with no direct IS A relationships to ancestors, and is the opposite of the Transitive Closure described in Section 3.5.

As an example, Figure 26 shows a hierarchy of concepts containing redundant IS A relationships on the left being converted to the 'proximal supertype view' on the right.

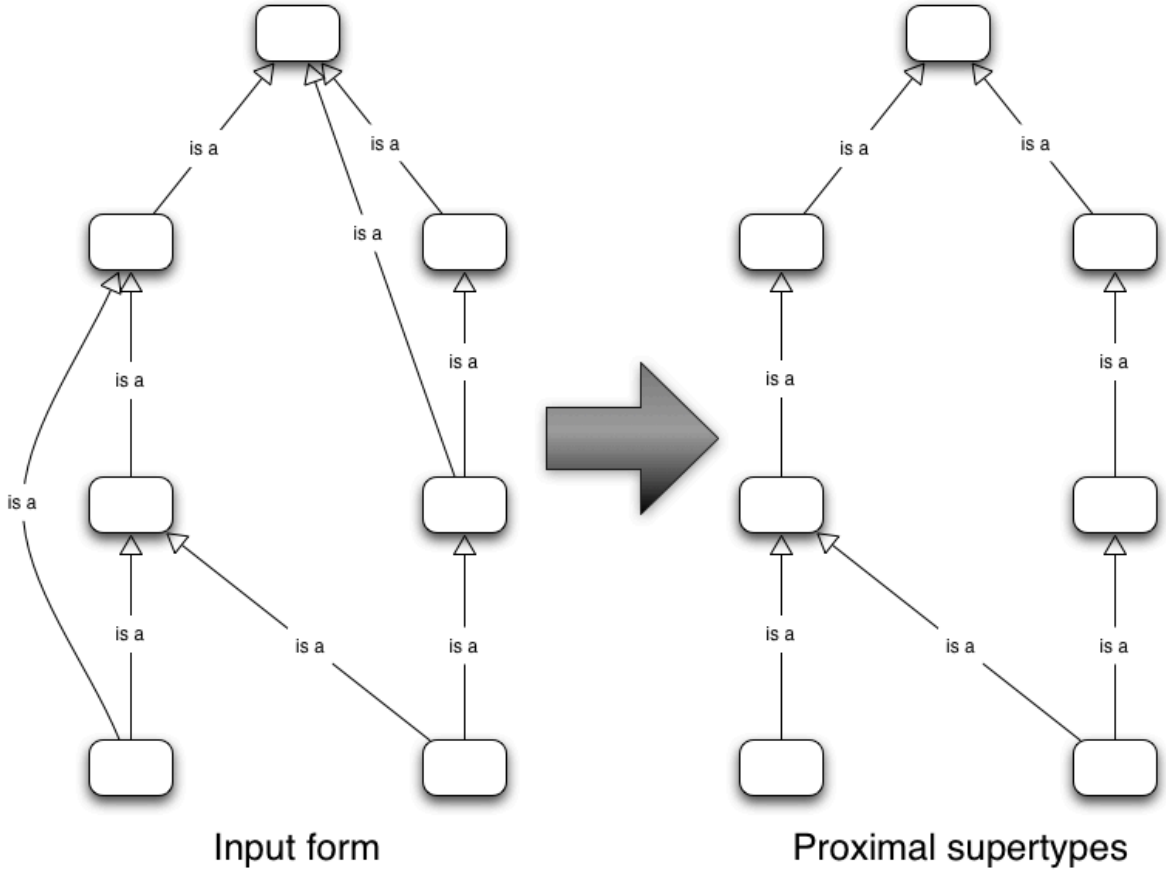


Figure 26 Example conversion to proximal supertypes for Distribution Normal Form

[14] Many algorithms exist for easily creating a transitive closure, including scripts provided with the same code accompanying this guide, and by the IHTSDO. See Section 3.5 for more details.

Non-redundant inferred defining relationships – this rule applies to all stated and inferred relationships a concept has, and means that only non-redundant defining relationships and relationship groups are included. This includes relationships inherited from supertype ancestors (not just direct parents). Relationships or relationship groups that are supertypes of other relationships or relationship groups are omitted (refer to Section 2.3.5.2 as an example of super/sub-relationships).

Figure 27 shows an example of an incomplete graph of concepts to demonstrate the 'non-redundant relationships rule'. Assuming the Stated Form on the left hand side of this diagram, all the inferred and inherited relationships are added, and then reduced to their non-redundant set. For example, note that 'rel type A' is inherited down to all of the concepts, however 'rel type B' is not.

This is because the relationship 'rel type C' from B to F is a subtype of the relationship 'rel type B' from A to E. Including a 'rel type B' relationship from B to E would therefore be redundant, however the 'rel type C' relationship to F is inherited by C.

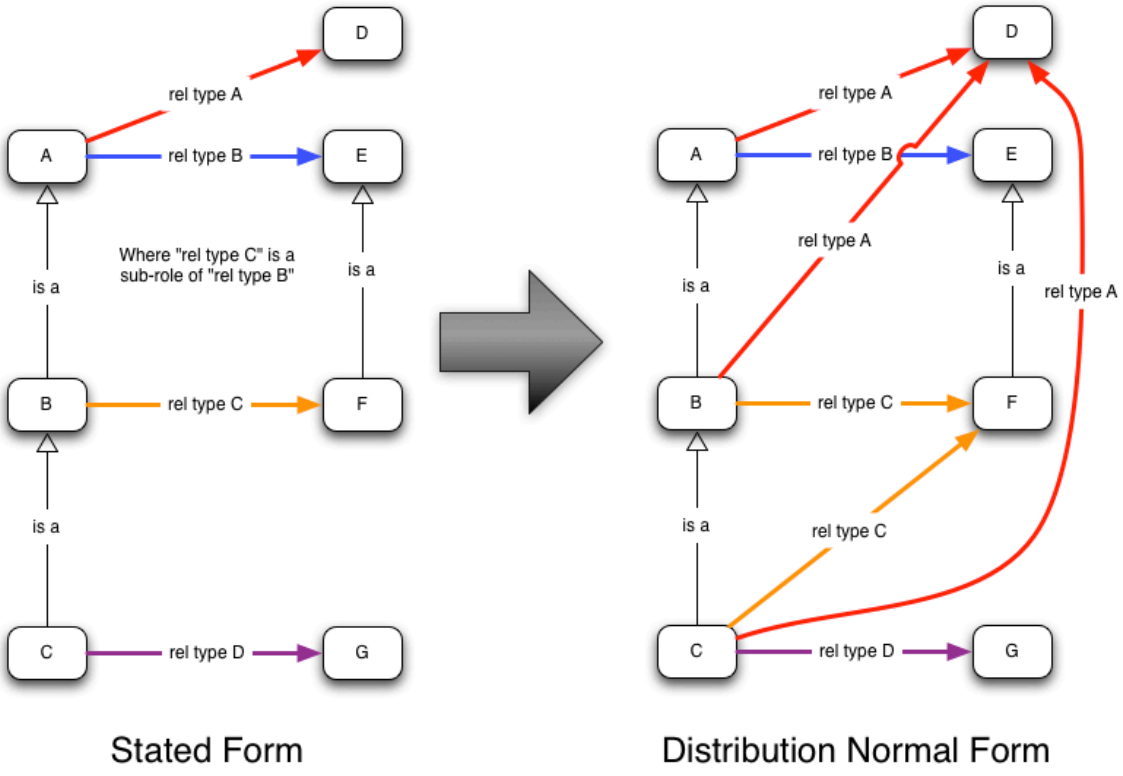


Figure 27 Conversion to non-redundant relationship view

3.4.4 Expected differences between AMT v3 Stated and Distribution Normal Forms

The AMT v3 Beta in Stated Form differs from Distribution Normal form in a number of ways. The following list provides individual examples; however these are not the limit of the differences, just examples of patterns of differences.

Missing inferred relationships – A number of inferred IS A relationships are not included in the Stated Form. An example of such a relationship is that the stated form includes the statement that

- |*paracetamol 500 mg tablet*| is a subtype of |*paracetamol (medicinal product)*|

However it does not include the statement:

- |*paracetamol 500 mg + codeine 10 mg tablet*| is a subtype of |*paracetamol product*|

This statement will be inferred and included in the Distribution Normal Form, because |*paracetamol 500 mg + codeine 10 mg tablet*| has the same HAS INTENDED ACTIVE INGREDIENT relationship to |*paracetamol (substance)*| as |*paracetamol (medicinal product)*|. This is similar to the example shown in Figure 25.

No inherited relationships – Relationships that are inherited down the hierarchy to child concepts are not included in the Stated Form. An example of this is that the Trade Product Unit of Use concept |*Panadol 500 mg tablet*| does not have the relationship HAS INTENDED ACTIVE INGREDIENT to |*paracetamol (substance)*| in the Stated Form.

In the Distribution Normal Form, this relationship will exist, inherited from the parent concept |*paracetamol 500 mg tablet*|. Therefore with the Stated Form it is necessary to navigate up to all parent concepts manually to find and use these relationships. In the Distribution Normal Form they are replicated on the child concepts for easier access. This is similar to the example shown in Figure 27.

Redundant relationships – Some relationships currently distributed in the Stated Form may become redundant in the Distribution Normal Form and therefore may be retired in future Distribution Normal Form distributions.

For example, Figure 28 shows the Stated and Distribution Normal Forms for |*paracetamol (medicinal product)*| and |*paracetamol + codeine (medicinal product)*|. Because |*paracetamol + codeine (medicinal product)*| is determined to be a subtype of |*paracetamol product*|, the directly stated IS A relationship from |*paracetamol + codeine (medicinal product)*| to |*medicinal product*| becomes redundant, and is not included in the Distribution Normal Form.

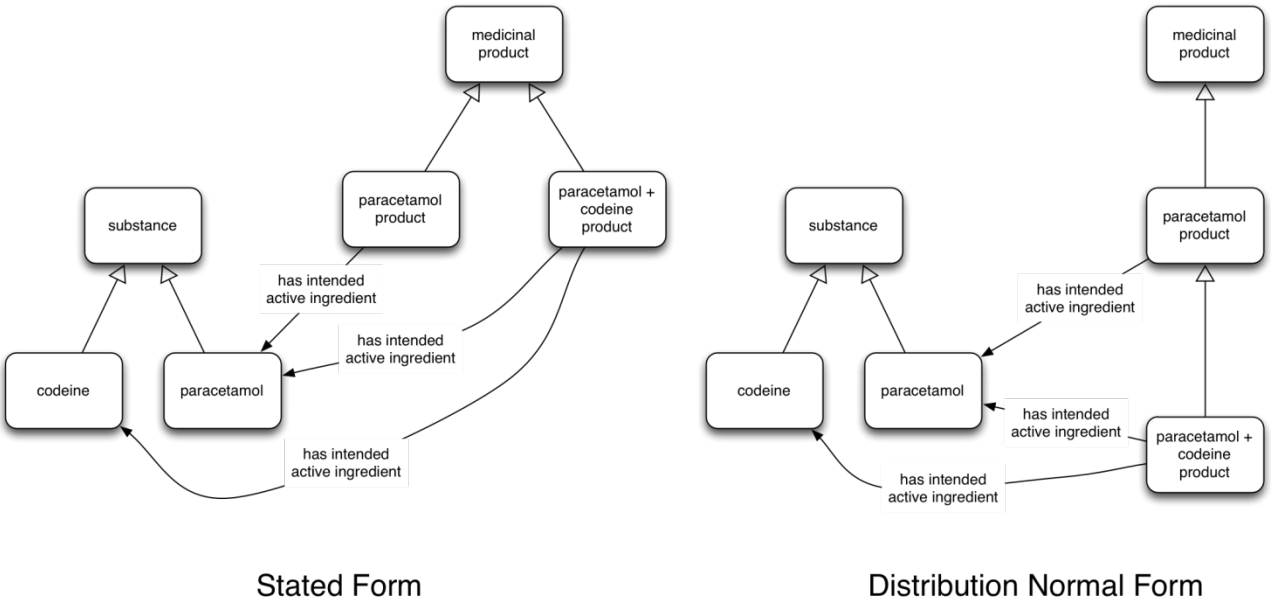


Figure 28: Comparative examples of Stated Form and Distribution Normal Form

3.5 Transitive Closure

Generation and use of a transitive closure can be a useful tool in searching terminology data, or in data retrieval/reporting on terminology encoded records.

The following sections explain what a transitive closure is, how to create one, and how to use it.

Note that generating a transitive closure on AMT v3 Beta will not produce all true subtype relationships because it is distributed in Stated Form. Transitive closure generation (as described below) is based on an initial set of IS A relationships, however the Stated Form of AMT v3 does not include all inferable IS A relationships in 'proximal supertype' form.

3.5.1 What is a Transitive Closure in SNOMED CT?

The opposite of the 'proximal inferred supertype' view included in the Distribution Normal Form (refer to Section 3.4.3), a 'transitive closure' view includes all possible true IS A relationships between concepts. This effectively adds to the 'proximal inferred supertype' view new IS A relationships for every possible multi-relationship traversal.

Best explained using a diagram, Figure 29 shows the Distribution Normal Form 'proximal supertype' view on the left turned into a transitive closure on the right.

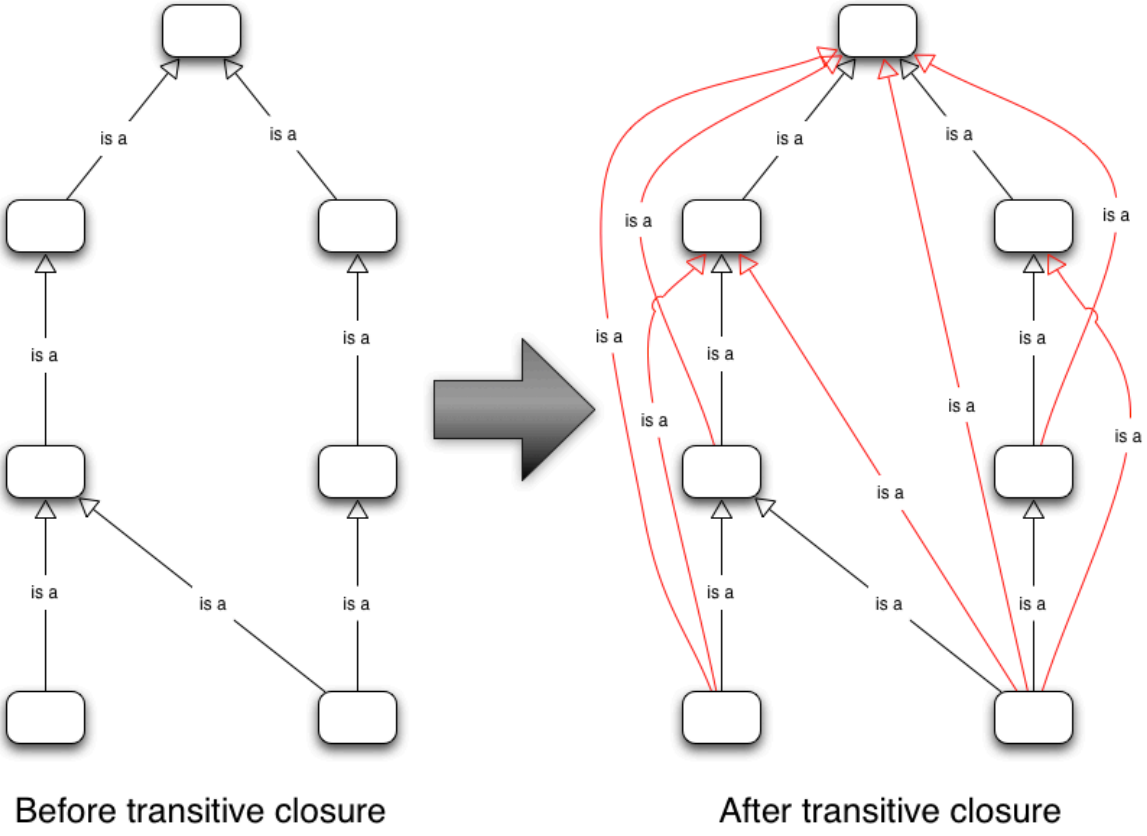


Figure 29 Example of a transitive closure

3.5.2 Purpose

In the context of SNOMED CT terminology implementation (including AMT v3) a transitive closure can be thought of as an index.

By generating all possible IS A relationships, it is possible to determine if a concept is a descendant of another concept (subsumption testing) with a single relationship. This avoids needing to traverse all possible routes through multiple relationships, which is typically slow and requires more complex code/queries.

For examples of a transitive closure being used, refer to Section 5.

3.5.3 Generation

A number of algorithms and technologies for calculating a transitive closure from RF2 relationships files exist. For example it is possible to use a relational database, or even a shell script with text file processing.

The IHTSDO provide a MySQL stored procedure implementation which generates a transitive closure into a MySQL database table. Refer to Section 7.7.5.2 Transitive closure implementation of the *SNOMED CT Technical Implementation Guide* [4]. The IHTSDO also provide some details of an in-memory representation and algorithm for fast subsumption testing in Section 7.7.5.2.3 of the same document.

Section 5 of this document also provides sample schema, load scripts and common queries for AMT v3 as an aid to understanding the distributed AMT v3 data. The sample schema and load scripts also include generation of a transitive closure using the IHTSDO's MySQL stored procedure. The accompanying sample queries demonstrate the use of the transitive closure to achieve fast subsumption testing and more compact queries.

3.6 Modules

3.6.1 What are modules?

RF2 introduced the concept of 'modules' to SNOMED CT. Put simply, a module is a way of segregating content for the purposes of identifying:

- organisational responsibility for content;
- dependencies between content; and
- groupings of content for a particular purpose or separate development/release cycle.

In RF2 each component has a 'moduleId' attribute which indicates the module to which that component belongs. The 'moduleId' can change over time to indicate its ownership has changed, or that it is being grouped with a different set of content. Changes to a module are tracked over a component's life using the same History Tracking Mechanism used for all RF2 content – refer to Section 3.7.

3.6.2 AMT v3 modules

AMT v3 contains four modules, as summarised below.

SNOMED CT Identifier	Name	Description
900000000000012004	SNOMED CT model component	IHTSDO module used to denote all metadata content used as a foundation for RF2 and SNOMED CT – i.e. non-clinical content.
900000000000207008	SNOMED CT core	IHTSDO module used to denote all SNOMED CT clinical content.
32506021000036107	SNOMED CT Australian extension	NEHTA module used to denote SNOMED CT-AU content.
900062011000036108	Australian Medicines Terminology module	NEHTA module used to denote AMT specific content.

Modules have dependencies expressed in the *Module dependency reference set* (refer to Section 3.1.9.1). Figure 32 below expresses the dependencies between the modules included in AMT v3 as expressed in AMT v3's module dependency reference set.

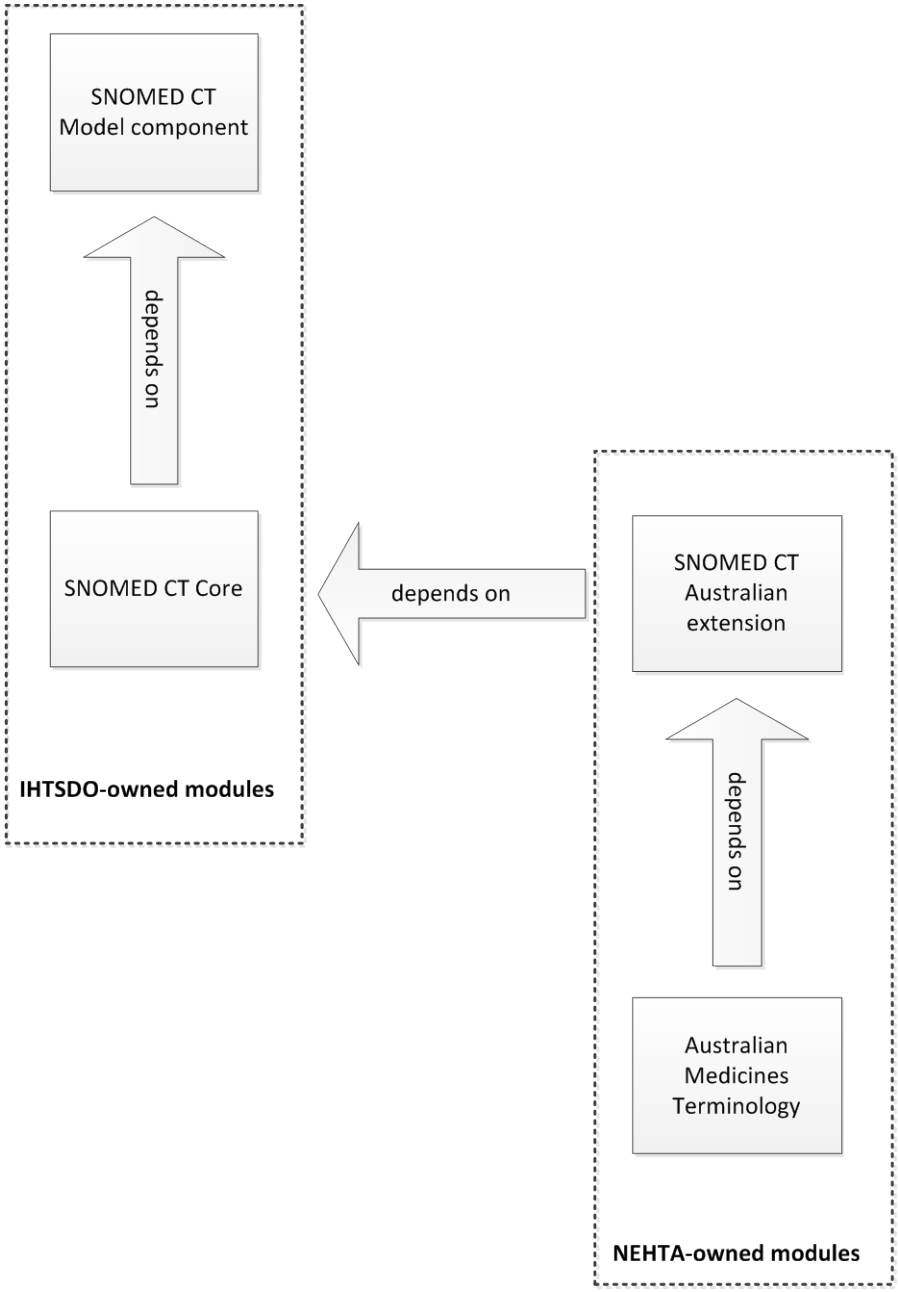


Figure 30: AMT v3 module dependencies

3.7 History Tracking Mechanism

The history tracking mechanism used in RF2 is described in 5.4.1.6 History Mechanism of the *SNOMED CT Technical Implementation Guide* [4]. The following sections provide an overview of this description.

3.7.1 Append-only data model

In short, RF2 provides a 'log-style' append-only data model, which provides full traceability of change. A row in any of the RF2 files represents a version of a component, and once released is never modified.

RF2 supports three types of release – Full, Snapshot, and Delta – refer to Section 3.3 RF2 distribution types. The Full type provides all of the versions (rows) ever released, whereas the Snapshot type provides just the most recent versions at the point in time of the Snapshot. The Delta format provides only versions that have been created since a previous release.

Regardless of the distribution type being viewed, it must be remembered that all of these versions are still valid historical states of the components, however the Delta and Snapshot types omit some for brevity and/or convenience.

3.7.2 Modifying a component

As a version of a component cannot be modified, to modify the state of a component it is necessary to add a new version of the component with the desired change.

The new version of the component supersedes the old version of the component as the component's state from the date/time of the new version's 'effectiveTime' field. Versions (rows) are therefore 'effective' from the point in time specified in their 'effectiveTime' field, to the point in time specified in the next chronological version's 'effectiveTime'.

Only one version (row) of a component is effective at any given point in time.

3.7.3 Immutable attributes

Components have mutable and immutable attributes. The mutability of an attribute is specified in Section 3.1 where the columns for each file are specified.

If an attribute is considered mutable it may appear in future versions of a component with a different value to the current and previous versions. If an attribute is considered immutable new versions are not permitted to modify the value of that attribute from the value in existing versions of the component.

That is, immutable attributes have a value set in the first version of a component and may never change for the entire life of the component. Mutable attributes may change value over time for a given component.

3.7.4 Inactive components

When a component is found to be no longer relevant, or not recommended for use, it is 'inactivated' (i.e. retired), but not removed. This is undertaken by creating a new version of the component and setting the 'active' field of the new version to 'false'. This indicates that as of the 'effectiveTime' value of the new version, the component is inactive.

Inactivation of a component is done for many reasons. For example:

- The component had an error in an immutable field, and the only way to correct this error is to inactivate the component and create a new one with the correct value.
- The component was a duplicate of another component, and therefore the duplicate component is inactivated.

When a component is inactivated and replaced by a new component, the component replacing it is specified in the Historical association reference sets – refer to Section 3.1.8 Association reference sets. This is intended to aid finding replacements for inactive components.

It is important to note that once created, components are not removed from the release files, only inactivated. This is to ensure that despite being inactive, components are always available into the future in case they have been used in clinical records. It is therefore important to note that when selecting components of the terminology to use, only active components should be used.

3.7.5 Semantics of identifiers

Once created, an identifier always refers to the same component, and that component does not change meaning over its lifespan.

Regardless of whether an identifier refers to an active or inactive component, the identifier (and the thing it refers to) always means the same thing.

4 Implementation advice

This section provides general advice and topics to consider when planning an implementation of AMT v3. Further, more product-specific, advice can be found in Section 7 Implementation considerations.

4.1 Mapping

AMT v3 may be integrated with an existing system by mapping either a local or 'Commercial Off The Shelf' (COTS) medicines dictionary to AMT v3.

This approach enables systems to continue to operate without large modifications, and without a change to the current user experience, while enabling use of the AMT for information exchange and data reporting/analysis. Using COTS medicines dictionaries mapped to the AMT also introduces features provided by such medicines dictionaries not provided natively within the AMT.

However, maps can be expensive to produce and maintain over time, particularly if both source and target of the map are living, changing products. Consideration should be given to the release and update cycles of both the COTS and AMT terminologies.

Please refer to *AMT Mapping Guidelines* [5] and *AMT Mapping Requirements* [6] if considering a mapped implementation.

4.2 Native implementation

An alternative to mapping AMT v3 to a local or COTS medicines dictionary is to directly implement AMT v3.

One of the first technical considerations when implementing AMT v3, or any SNOMED CT terminology, is how to store and retrieve the terminology as reference data[15].

The obvious aims to storing terminology as reference data are:

- to enable searching for values when entering transactional data;
- to render transactional data containing fields encoded with terminology; and
- to report across transactional data using the terminology reference data to group and filter.

There are two broad approaches that can be taken in this respect:

- custom schema and application code for the implementation; or
- use of an external terminology server.

Both have advantages and disadvantages that must be weighed when planning an implementation.

4.2.1 Embedded implementation

One option to integrate AMT v3 with an application is to embed AMT v3 terminology data in application reference data, as illustrated in Figure 32. This may be achieved using mappings to existing application reference data sources, or manipulating AMT v3 data to a form useful for the application.

[15] Reference data is data in a system that is static (aside from occasional revisions) and non-transactional used to support the operation of the system and the transactional data.

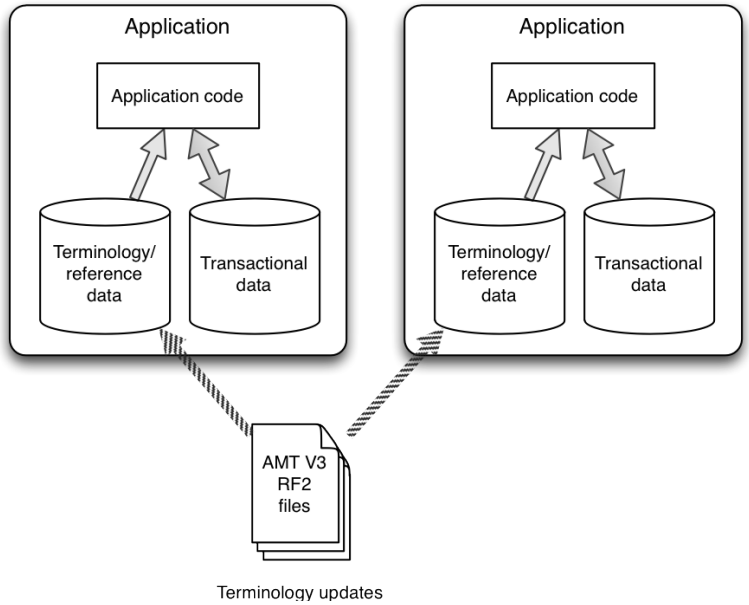


Figure 32 Example of embedding AMT v3 reference data

RF2 is a relational representation of the terminology data used to distribute AMT v3, but is not intended to be an implementation format.

In order to explain and demonstrate the distribution form and AMT v3 model, Section 5 (Sample code and scripts), is written using the release files directly translated to relational database tables. While this suits the purpose of explaining and demonstrating the distribution form, it is not likely to be the most suitable reference data schema for most applications.

Similarly, the content of the RF2 release files contains a complete tracking of all aspects of the terminology; however most implementations will only require varying subsets of this content.

Therefore identifying the subset of AMT v3 data required for the implementation will enable designing the most appropriate and efficient data structures. In the process of designing this data structure and access methods, it is important to keep in mind the process for updating this data at regular intervals – refer to Section 4.3 Designing for data updates.

Note that the data storage technology need not necessarily be a relational schema. There are potentially non-relational storage and access options which may provide as good or better performance in some circumstances. Other approaches include loading the entire terminology into memory for fast searching.

4.2.1.1 Reasoning

Reasoning with the AMT v3 data is also possible, and for this purpose a format such as OWL is likely more appropriate. The IHTSDO distribute an RF2 to OWL/KRSS Perl script for this purpose for SNOMED CT International. Unfortunately this script cannot be used for the AMT v3 RF2 as the IHTSDO script has not been updated to take account of Concrete domains and data type properties.

As part of the IHTSDO Representation of Numbers project, this Perl script will be updated for Concrete Domains and will then become usable for AMT v3 data. This script is under development at present, however is currently sufficient to convert AMT v3 to a useable OWL Functional Syntax format.

The script under development can be found in the IHTSDO's Subversion repositories at https://csfe.aceworkspace.net/svn/repos/perl-utilities/branches/concrete-domain/src/main/perl/tls2_StatedRelationshipsToOwlKRSS_Draft_INT.pl

4.2.1.2 Subsumption

Regardless of whether formal reasoning with the full description logic is performed, it is likely that some form of subsumption testing or subsumption-based querying may be useful and should be planned. For example to find out if a particular AMT concept is a subtype of another AMT concept, or to find all subtypes of an AMT concept.

Techniques similar to those described in Section 3.5 Transitive Closure are useful to achieve fast, readable queries performing subsumption testing. Section 5.3.4 demonstrates how these features can be used.

4.2.2 Terminology servers/services

Terminology servers/services may be used as an alternative to implementing storage, retrieval and maintenance of terminology within an application. Typically a terminology server is a server based application running outside a clinical application, often on a different physical machine, which manages and serves terminology to one or more applications. This is illustrated in Figure 33.

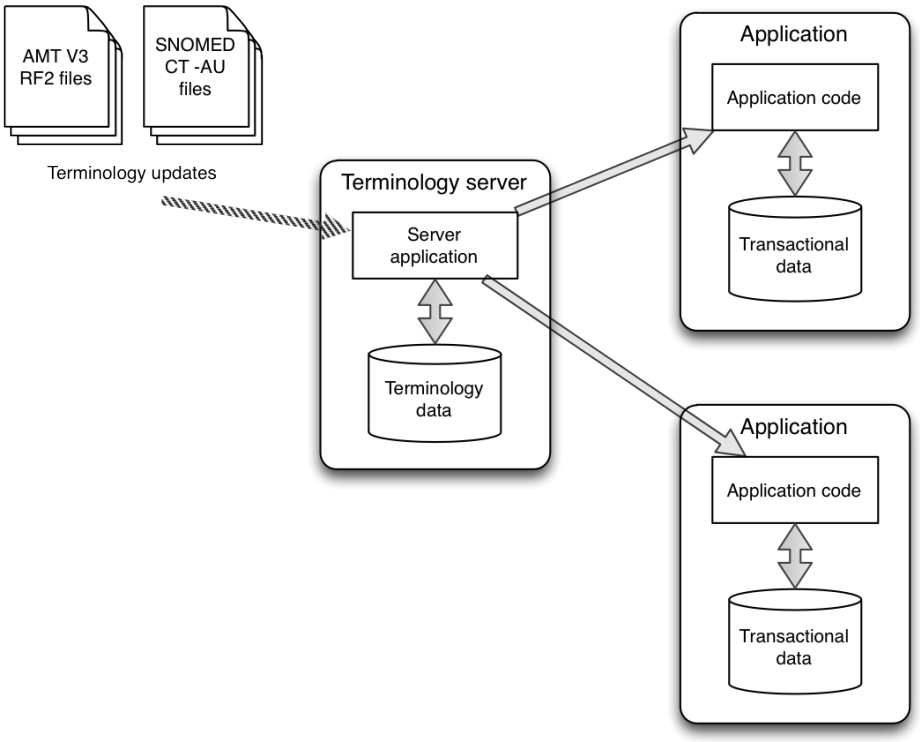


Figure 33 Example of a terminology server servicing multiple applications

One of the primary advantages of a terminology server is that they take the complexity of managing terminology out of the application itself, allowing the application developers to focus on the primary business use of the system. This can include:

- Managing multiple terminologies/code systems
- Managing multiple terminology versions
- Managing terminology updates
- Real time searching and contextual restriction
- Embeddable widgets for streamlined clinical information capture
- Subsumption testing
- Advanced reasoning

Use of a terminology server and participating applications also allows an organisation with multiple applications and/or deployments to achieve partial or complete centralised maintenance and deployment of terminology changes.

Different terminology servers provide different arrays of functionality and interfaces which may be more or less applicable to a particular organisation or implementation. Therefore it is necessary to perform a requirements analysis prior to attempting to assess and select a terminology server supplier.

4.2.3 Terminology servers versus native implementation

Bespoke terminology data structures and implementation can provide fast functionality, well targeted to the system use cases. However this approach does have disadvantages:

- More complexity in the application which might otherwise be factored out to an external system (terminology server).
- Distraction for application developers from the main purpose of the system.
- More code to maintain and test.
- Design must take into account regular terminology updates.
- Reduced ability to centrally manage terminology for multiple systems in an organisation.
- Complex reasoning features provided by terminology servers are hard to reproduce cheaply.

Terminology servers provide an alternative which, depending upon the product chosen, can eliminate one or more of the above disadvantages. We recommend considering the following characteristics when assessing terminology servers:

Cost	Depending upon the server and the implementation, terminology server costs may exceed, or be substantially cheaper than, developing custom application functionality.
External point of failure	An external terminology server, depending upon the nature of the deployment how it is used within an application (i.e. real-time access or offline with updates), may present an additional point of failure into a deployment.
Third party component	Introduction of a terminology server will include an additional third party component, requiring the usual licence and contractual agreements.

Unless implementation of terminology within an application is trivial, a terminology server is likely to be simpler and cheaper than implementing, testing and maintaining custom functionality. However this decision must be made on a case-by-case basis.

4.3 Designing for data updates

Regardless of implementation type or technologies, terminology will have to be regularly updated within an implementation. Under the terms of the Australian National Terminology Release Licence, an implementation is required to update to a new version of AMT at minimum every 180 days, however business needs may require more frequent updates.

4.3.1 Mapping

Updating to new data versions is a major ongoing cost of using a map between a local or COTS medicines dictionary and the AMT. Given the rate of change of medicines products and licence requirements, regular updates will be required.

When performing local mappings, it is necessary to carefully plan and design the process for:

- assessing changes in AMT;
- updating the map; and
- reviewing and testing the map.

Typically producers of COTS products will provide their own mapping, maintained and updated along with regular product updates.

4.3.2 Native implementation

When implementing custom, native data structures, it is important to consider implications of updates to ensure that they can easily be performed frequently.

There are two broad views that can be taken in storing terminology data, which relate closely to RF2's concept of distribution type (refer to Section 3.3):

- **Store and query a 'snapshot'**.

  This approach stores only the latest state of the terminology components implemented. Queries or code written against this form are usually simpler and faster as they don't need to consider changes to terminology components over time.

  However updating this form is more difficult, requiring dumping the existing snapshot and replacing it with a new snapshot, or attempting to update the snapshot with changed data. Rollback is similarly more complex, and it is only possible to operate with one version of the terminology at a time increasing change management challenges.

- **Store and query full release history**.

  This approach stores all versions of terminology components implemented. Queries or code written against this form are more complex, and may be slower unless carefully planned, due to the additional time dimension.

  However updating data is much simpler, as new versions of components in the terminology may simply be appended to the existing data store. Queries/code written to correctly handle change over time may be configured to retrieve data based on a given time point, representing any release point in history. Alternatively SQL views can be created which provide a Snapshot view of the Full data stored in the database based on a configured time point in order to keep queries simpler.

  This enables fast and flexible rollback mechanisms and offers options to change manage new terminology updates via configuration rather than physically loaded data.

  Note that this approach cannot be prototyped with the AMT v3 Beta release as this release is only distributed in Snapshot form.

Business processes to manage application configuration testing and deployment will also be required.

4.3.3 Terminology server updates

Typically terminology servers (refer to Section 4.2.2) will contain functionality to update to new terminology versions, and often enable configuration of terminology versions served to different applications. This is a considerable benefit of terminology servers, as this process is typically well defined and streamlined.

However business processes for planning, deploying and testing applications with new terminology versions will be required.

4.4 Searching and capturing input

Searching and navigating during data input is vital to an efficient and positive user experience. In order to achieve a good user experience there are some relatively simple steps that can be taken, as discussed in this section.

The NHS Connecting For Health also produced considerable useful documentation on this topic in their Common User Interface programme, available online at http://www.connectingforhealth.nhs.uk/systemsandservices/data/cui.

Specific to searching, the following list provides factors to consider when implementing user facing controls and backend reference data access:

Search speed

Fast search responses are critical to a productive user experience. Use and optimisation of appropriate indexes can achieve near real-time results.

Dynamic search features

Dynamic features are pre-emptive suggestions made by the system before the user has completed typing their search. Their aim is to reduce the time taken for a user to complete a search.

- Autocomplete – autocomplete is a common concept in modern software, and is especially pervasive in mobile phones, where the most likely complete word for a partially typed word can be accepted by the user with a shortcut key. Care must be taken when applying these techniques to ensure safety, and it is also very easy to frustrate users with autocomplete.
- Auto search suggestions – auto search suggestions is a feature familiar to most through modern web browsers and search engines. Through this feature a short list of probable complete search text options are provided based on the text typed by the user to that point. Search suggestions can be based on previous searches by that user or the entire user base, or on elements of the data being searched.
- Incremental search results – this feature provides search results to the user based on the text they have typed so far, but prior to them completing their search. This shows users search results being refined as they type, providing feedback to the user enabling them to alter their search. It is also likely that the result the user is looking for will be present before the user completes their search text, saving considerable time.

Ranked / ordered search results

Search results need to be ordered. However search results may be ordered by many attributes. The most obvious is alphanumeric order, which will be familiar to most users. However often ordering the results by most probable match to the user's search is much more powerful. Search engines will typically provide ranking of search results, however combining this with the most probable type of AMT concept being searched for (for example an MPUU or MPP) will achieve a better user experience.

Partial word matching	Partial word matching is implemented by most search engines, and saves users considerable time typing long words. When searching for a term containing multiple words, using multiple partial words can achieve an accurate search result with minimal typing.
Word order	Searches should be tolerant of different word orders in search text and search results. For example 'codeine paracetamol' should match 'paracetamol 500 mg + codeine 10 mg tablet'.
Stemming	Stemming is a technique used to reduce a word to its stem, base or root form. Taking a simple English example this allows 'fishing', 'fish', 'fished' and 'fisher' to be reduced to the root word 'fish'. This techniques is very useful when searching SNOMED CT which contains descriptions with many derivative forms of words, it is less likely to be effective with AMT's content.
Synonyms	When searching SNOMED CT, many more relevant search results will be achieved by searching on all acceptable synonyms, rather than just preferred terms. This enables a user's search text to match potentially alternate terms for a concept that may not be the preferred term. While very effective in SNOMED CT, this is likely to be less effective in AMT which currently contains few alternate synonyms for concepts. **Note that despite matching a synonym, the preferred term should always be rendered to the user to select.**
Structured results	When displaying search results, structuring the results as well as ordering can assist users quickly choosing their intended concept. For example grouping or flagging results to indicate what type of AMT concepts they are (for example MPUU versus MPP versus TPUU or CTPP). Results can also be further organised or categorised using the relationships within the AMT model – for example the subtype hierarchy, or using relationships to ingredients or forms.
Constraining to relevant content	One of the simplest and most effective steps in data entry for SNOMED CT and AMT is reducing the navigated/searched concepts to those relevant to the context. For example, if a user is entering data into a field intended only for trade specific concepts, returning results from other parts of the AMT model clutters search results and impedes data entry.

Similar to providing structured search results, it is possible to provide users with the ability to navigate through AMT content using concepts' relationships. For example it is possible for a user to specify an ingredient, or combination of ingredients, and then navigate through the generic medicines grouped by form and strength, down to trade products grouped by those generic medicines. Navigation of the subtype hierarchy for data entry is not recommended for SNOMED CT, due to its variable depth and breadth, however AMT's subtype hierarchy is much more suited to this approach.

Efficiency and usability of navigation versus searching controls will vary depending upon the setting and use case. Regardless of approach, user data entry should be limited to active AMT concepts and their current Preferred Terms at the time of entry.

When a user selects an AMT concept (either via navigation or searching), they should be presented with and select:

- the AMT Preferred Term for the concept; or
- a local/COTS medicines dictionary term in a mapped implementation (refer to Section 4.1).

Clinical users of applications **should not** be presented with Fully Specified Names or SNOMED CT Identifiers. These are part of the infrastructure of the terminology, and not intended for clinicians.

4.5 Recording and rendering recorded data

When a user's selection is recorded, two pieces of information should be persisted:

1. The identifier of the code selected.
    - If using AMT natively, this will be the AMT concept's SNOMED CT Identifier.
    - If using a mapped implementation, this will be the local or COTS medicines dictionary's code for the selected item.
2. The text seen and selected by the user.
    - If using AMT natively, this will be the AMT concept's Preferred Term.
    - If using a mapped implementation, this will be the local or COTS medicines dictionary's description text that was rendered to the user.

The code and text are both stored for safety, and if any doubt exists the stored text seen by the user is the definitive record.

Additionally systems should record the version of the AMT and/or medicines dictionary being used at the time the record was made. This information is valuable as an audit trail and provides a useful tool for later diagnostics.

When rendering recorded data back to users, the original text seen by the user (recorded when the selection was made as discussed above) should be rendered. The recorded code **should not** be used to look up the latest AMT Preferred Term in the loaded reference data at the time of rendering. This avoids the risk of discrepancies between the code and description over time, which however unlikely, may occur in AMT or in its implementation.

4.6 Retrieval and analytics

Since AMT is a SNOMED CT terminology, it is able to combine encoded instance/transactional data with reference data, providing a powerful combination for data retrieval and analytics. For example these features can be used to:

- Aggregate and analyse records for research, audit and service planning.
- Identify patients with specific risk factors for preventative/investigative measures, or identification of patients for clinical trials.
- Retrieve data for an individual patient to enable summary reports or decision support protocols.

The basic principle is to determine a query on AMT which yields the list of concept identifiers encompassing the data required. This set of concepts may be joined with instance data to determine the set of instance records required.

4.6.1 Subtype hierarchy

Perhaps the simplest approach to retrieving data is to retrieve all records where a field contains a subtype of an AMT concept.

For example, to find all records where the medication prescribed contained codeine, it is possible to find all the subtypes of the AMT MP concept |*codeine (medicinal product)*| using a description logic classifier or Transitive Closure (refer to Section 3.5). This list can be joined with prescription records and will include all prescriptions for codeine-containing prescriptions based on either their generic or trade name.

The same technique can be used anywhere the supertype/subtype hierarchy in AMT suits the required query.

Note that AMT v3 Beta's Stated Form does not include all inferred proximal supertype relationships, and therefore not all possible subtypes can be retrieved using a transitive closure generated from AMT v3 Beta.

4.6.2 Other concept attributes

Note: The scripts provided below and in other release materials are provided for educational purposes only, as noted in Section 1.5.

It is possible to query concepts from AMT using attributes other than their supertype/subtypes. For example it is possible to find all AMT concepts containing more than 10 milligrams of codeine in a tablet form by creating a query looking for concepts with these characteristics.

```
select
    v3_ingredient_strength.mpuuid,
    v3_ingredient_strength.mpuuterm,
    v3_ingredient_strength.bossterm,
    v3_ingredient_strength.substanceterm,
    v3_ingredient_strength.strengthvalue,
    v3_ingredient_strength.unitterm,
    toPt(hasDoseForm.destinationid)
from v3_ingredient_strength

    join rf2_ss_relationships hasDoseForm
        on hasDoseForm.sourceid = v3_ingredient_strength.mpuuid
        and hasDoseForm.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has manufactured dose form (relationship type)')
        and hasDoseForm.active = 1

where v3_ingredient_strength.strengthvalue > 10
    and v3_ingredient_strength.unitid = (
        select distinct conceptid from rf2_ss_descriptions where term = 'mg/each'
        )
    and v3_ingredient_strength.substanceid = (
        select distinct conceptid from rf2_ss_descriptions where term = 'codeine (AU substance)'
        )
    and hasDoseForm.destinationid = (
        select distinct conceptid
        from rf2_ss_descriptions where term = 'tablet dose form (AU qualifier)'
        )
group by v3_ingredient_strength.mpuuid;
```

Note that the above example assumes that the ingredients are all expressed in the same unit of measure – i.e. this query would not work for a tablet expressed as '1 gram' or '0.02 gram' even though they match the specified criteria. For more on this issue refer to Section 3.2.6.

Subsumption techniques are also quite useful when looking for concepts that have a certain type of relationship to a subtype of another concept. For example where looking for an AMT concept with 'dihydrocodeine' in any type of tablet dose form:

```
select
    v3_ingredient_strength.mpuuid,
    v3_ingredient_strength.mpuuterm,
    v3_ingredient_strength.bossterm,
    v3_ingredient_strength.substanceterm,
    v3_ingredient_strength.strengthvalue,
    v3_ingredient_strength.unitterm,
    toPt(hasDoseForm.destinationid)
from v3_ingredient_strength

    join rf2_ss_relationships hasDoseForm
       on hasDoseForm.sourceid = v3_ingredient_strength.mpuuid
       and hasDoseForm.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has manufactured dose form (relationship type)')
        and hasDoseForm.active = 1
```

```
where v3_ingredient_strength.substanceid = (
    select distinct conceptid
    from rf2_ss_descriptions where term = 'dihydrocodeine (AU substance)'
     )
    and (
        hasDoseForm.destinationid = (
            select distinct conceptid
            from rf2_ss_descriptions where term = 'tablet dose form (AU qualifier)'
            )
        or hasDoseForm.destinationid in (
            select source from v3_tc where dest = (
                select distinct conceptid
                from rf2_ss_descriptions where term = 'tablet dose form (AU qualifier)'
                )
            )
        )

group by v3_ingredient_strength.mpuuid;
```

Similarly subsumption techniques also work for sub-role relationships. For example a query could be written to find all concepts with a HAS MPUU relationship to |*paracetamol 500 mg tablet*|, which will return a list of MPP concepts. If this query is expanded to return all concepts with a HAS MPUU or subtype relationship to |*paracetamol 500 mg tablet*| or its subtypes, the result will also include CTPP concepts.

```
select
    hasMpuu.sourceid as mppid,
    toPt(hasMpuu.sourceid) as mppterm,
    hasMpuu.typeid as hasmpuu_typeid,
    toPt(hasMpuu.typeid) as hasmpuu_typeterm,
    hasMpuu.destinationid as mpuuid,
    toPt(hasMpuu.destinationid) as mpuuterm,
    subRole.sourceid as sub_sourceid,
    toPt(subRole.sourceid) as sub_sourceterm,
    subRole.typeid as sub_typeid,
    toPt(subRole.typeid) as sub_typeterm,
    subRole.destinationid as sub_destinationid,
    toPt(subRole.destinationid) as sub_destinationterm
from v3_mpp as mpp
    join rf2_ss_relationships hasMpuu
        on hasMpuu.sourceid = mpp.id
        and hasMpuu.active = 1
        and hasMpuu.typeid = (
            select distinct conceptid from rf2_ss_descriptions where term = 'has MPUU'
        )
        and hasMpuu.destinationid = (
            select distinct conceptid
            from rf2_ss_descriptions where term = 'paracetamol 500 mg tablet'
        )
    join rf2_ss_relationships subRole
        on subRole.active = 1
        and exists (
            select 1
            from v3_tc
            where dest = hasMpuu.destinationid
                and source = subRole.destinationid
        )
        and exists (
            select 1
            from v3_tc
            where dest = hasMpuu.sourceid
                and source = subRole.sourceid
        );
```

Note that this technique is made simpler by the Distribution Normal Form, which provides all inherited relationships at each concept. However the AMT v3 Beta Stated Form does not provide all inherited relationships for each concept, and this means that more complex queries are required to check each concept and its ancestor concepts for relationships. Even so, not all relationships will be accessible through ancestor concepts as not all inferred supertypes are included in the Stated Form. See Section 3.5 for more details.

4.6.3 Description logic classifiers

A description logic classifier can also be used to find subtypes and supertypes of concepts within AMT. From the AMT v3 Beta Stated Form, use of a description logic classifier is the only reliable way to achieve this. However future releases in Distribution Normal Form allow for techniques discussed in Sections 4.6.1 and 4.6.2, which are typically simpler to set up, operate and perform.

However description logic classifiers allow for the use of Post Coordinated Expressions, which are essentially a collection of logic statements, much the same as a concept definition. A post coordinated expression effectively forms an anonymous or runtime concept, the same as the concepts distributed in AMT which are known as ‘precoordinated concepts’.

Description logic classifiers are capable of classifying a post coordinated expression to determine all of the concepts that subsume the expression, and all the concepts subsumed by the expression. This provides a very powerful and flexible alternative to the technique described in Section 4.6.2, where structural queries can become quite cumbersome to express a complex set of attributes.

4.6.4 Terminology servers

Terminology servers usually provide an array of services to interrogate terminology content, such as AMT, to achieve the concepts discussed in Sections 4.6.1, 4.6.2, 4.6.3, and 4.6.5. This eliminates the need to store terminology reference data structures enabling these queries, and implementation of these queries themselves.

Particularly powerful are terminology servers incorporating description logic classifiers, enabling complex and powerful queries yet hiding most of the complexity in the server.

4.6.5 Maps

Retrieval can also be performed via maps from AMT concept identifiers to other data sources, such as PBS. As more products are mapped to AMT, AMT concept identifiers can form a common reference point and bridge to map to other types of medicines information.

5 Sample code and scripts

5.1 Description

Contained within this release is a collection of SQL scripts which are intended to provide a starting point for loading and working with AMT v3 in a relational database environment.

Note: These scripts are not a suggested implementation or a base for implementation. Their purpose is as an aid to learning AMT v3's release format and data structures. As such the database schema mirrors the release file structure.

Collectively, these scripts illustrate how to:

- create a MySQL database schema into which the AMT v3 RF2 release files can be loaded;
- create a set of useful database views and procedures to simplify querying the imported AMT v3 data; and
- write queries which utilise the AMT v3 model to retrieve the necessary data to implement the primary AMT v3 use cases.

The scripts referred to above are contained in the AMT v3 release in a file listed as *Australian Medicines Terminology v3 Technical Implementation Guide Scripts*[16]. The following table identifies each file contained in the zip file and a brief description of the purpose and content of that file.

File	Description
schema/schema.sql	Contains table and view creation scripts.
schema/importTables.sql	Contains SQL statements used to load the AMT v3 RF2 files into the created schema.
schema/createIndexes.sql	Contains SQL statements to add database indexes to the schema. These are generally added after importing the data, to maximise the performance of the import.
schema/routines.sql	Contains SQL statements to create any user-defined routines (functions and procedures). These generally consist of helper functions to shortcut repetitive coding. They will be referred to in the sample scripts.
schema/derived-model/schema.sql	Additional 'derived model' tables to facilitate query simplification and performance.
schema/derived-model/importTables.sql	Population scripts for the additional derived model table.
schema/derived-model/createIndexes.sql	Index creation scripts for derived model tables.
sql/seven-notable-concepts.sql	A simple set of queries to list all concepts in each of the seven 'notable' classes.

[16] The filename is NEHTA_1240_2013_AMT_v3_TIG_Scripts_20130204.zip.

File	Description
sql/use-case.sql	A set of more complex queries which illustrate how to utilise the AMT v3 model to satisfy prescribe and dispense use cases.

5.2 Preparation

This section assumes the reader has installed MySQL and created a database schema into which the AMT v3 release files can be loaded. If this is not the case, http://www.mysql.com provides free downloads of their MySQL Community Server and installation and configuration instructions can be found on the download site.

Following this, the six schema scripts should be executed in the order in which they appear in the above table. The scripts assume that the AMT v3 bundle and the tig_sample_code.zip file are extracted in the following directory structure:

```
<some-root-directory>/
        release-files/
                RF2Release/
        tig/
                schema/
                sql/
```

With a current working directory of <some-root-directory>, open a MySQL session[17].

The following statements will create the schema and import the RF2 release files.

```
mysql> source tig/schema/schema.sql
mysql> source tig/schema/importTables.sql
mysql> source tig/schema/createIndexes.sql
mysql> source tig/schema/routines.sql
mysql> source tig/schema/derived-model/schema.sql
mysql> source tig/schema/derived-model/importTables.sql
mysql> source tig/schema/derived-model/createIndexes.sql
```

Please note:

- The above scripts have been saved in the default MySql CLI format, using CR (carriage return) without LF (line feed). For this reason, these files will **not** display line endings correctly in some Windows (and older Mac) based programs. We specifically discourage opening these files in MS Notepad and suggest that you use an editor that honours CR line endings when displaying and writing the file content.
- The 'tig/schema/importTables.sql' script contains relative paths to the RF2 files. Depending on the operating system and version of mysql, you may need to amend these and replace with the full path. For example:
    - 'release-files/RF2Release/Snapshot/Terminology/sct2_Concept_Snapshot_AU1000036_20121231.txt'
      changes to
      'C:/Users/nehta/Downloads/release-files/RF2Release/Snapshot/Terminology/sct2_Concept_Snapshot_AU1000036_20121231.txt'

Once this has occurred, the release has been imported and the imported AMT v3 data can be queried using the sample queries contained in the sql directory.

[17] Instructions on how to open a mysql session and how to execute commands is available on the MySQL website http://www.mysql.com.

It should be noted that the queries are provided as a starting point to demonstrate a relatively simple means of importing and querying the v3 model. It is intended to be exemplary only, and should not be considered fit for any other purpose.

5.3 Notable aspects of the schema creation scripts

5.3.1 Full tables

Beyond the Beta release, a historically complete 'Full' release of AMT V3 will be loaded into the following tables:

- rf2_full_concepts
- rf2_full_descriptions
- rf2_full_relationships
- rf2_full_language_refset

As Beta will not contain the 'Full' release of AMT v3 the above tables will be loaded with the Snapshot release. The snapshot views in Section 5.3.2 will continue to work in the same manner once the 'Full' release becomes available. Additionally, the example queries are targeted at the snapshot and will likewise remain unchanged beyond Beta.

5.3.2 Snapshot views

In most circumstances, a consumer of AMT v3 will be interested in working with the 'snapshot' release, which contains only the most recent version of each released component. The snapshot can be presented as a view of the full release, as shown below:

```
CREATE VIEW rf2_ss_concepts AS
select t1.* from rf2_full_concepts t1
where t1.effectivetime = (
    select max(t2.effectivetime)
    from rf2_full_concepts t2
    where t1.id = t2.id)
```

In the statement above, the rf2_ss_concepts view (i.e. snapshot concepts view) presents the most recent version of every concept in the full release. Snapshot views are also created for the descriptions, relationships and language reference set in the schema.sql file.

5.3.3 Notable concept views

It is possible to write a query which extracts every member of a notable concept class (e.g. Medicinal Product Pack), and there are several valid ways to achieve this. One method is to find all source concepts of active relationships of type IS A which have a destination concept of 'medicinal product pack (medicinal product pack)', as shown below:

```
select sourceid
from rf2_ss_relationships
where typeid = 116680003 -- is a
and destinationid = 30513011000036104 -- medicinal product pack
and active = 1
```

While the above query returns all the MPPs when querying the 'stated form' it requires further revision to be compliant with DNF or where a hierarchy has a depth greater than one level (i.e. this query will only return direct children of |*medicinal product pack*|, and not all descendants). These deficiencies can be addressed via the transitive closure table which returns all descendants. Since this example should **only** return MPPs it then also needs to exclude concepts that are also descendants of |*containered trade product pack*|. This exclusion is required due to an IS A relationship between CTPP and MPP, as detailed in Section 2.3.2.3. Further explanation and exemplar of transitive closure can be found in Sections 3.5 and 5.3.4 respectively.

```
select source
from v3_tc
where dest = 30513011000036104 -- medicinal product pack
and source not in (
        select source
        from v3_tc
        where dest = 30537011000036101 -- containered trade product pack
)
```

To avoid hardcoding the concept identifiers, the query can be improved as follows:

```
select source
from v3_tc
where dest = (
    select distinct conceptid from rf2_ss_descriptions
    where term = 'medicinal product pack (medicinal product pack)'
    and active = 1)
and source not in (
    select source
    from v3_tc
    where dest = (
        select distinct conceptid from rf2_ss_descriptions
        where term = 'containered trade product pack (containered trade product pack)'
        and active = 1)
)
```

A second method of obtaining all concepts within a notable class is to use the corresponding notable reference set. Every child concept of a particular notable class is also given membership of its corresponding notable reference set. The component referenced by the reference set member is the child concept. This can be queried as follows:

```
select member.referencedcomponentid
from rf2_ss_cRefset member
where member.refsetid = (
    select conceptid from rf2_ss_descriptions
    where term = 'medicinal product pack reference set (foundation metadata concept)'
    and active = 1)
and member.active = 1
```

The above query uses the table rf2_ss_cRefset, which is effectively the union of all snapshot component reference set files contained in the release. The refsetid column identifies the reference set (in this case, MPP reference set), and the referencedcomponentid identifies the child concept of the notable class.

This query is the basis of seven views contained in the schema.sql file, one for each of the seven notable concept classes (e.g. v3_mpp). These views provide a shortcut mechanism for quickly accessing concepts of a particular notable class, without having to explicitly query the component refset and concepts tables. For example, using these views allows the above query to be replaced with:

```
select id from v3_mpp
```

5.3.4 Creating and using the transitive closure

A transitive closure presents an exploded view of all IS A relationships contained within the terminology. For example, if concept A IS A concept B, and concept B IS A concept C, then it can be inferred that concept A IS A concept C, even if that relationship is not explicitly stated. The transitive closure table (v3_tc) includes a row for each of these inferred relationships, as well as including all explicitly stated IS A relationships.

A procedure for creating that table is provided in routines.sql. The implementation of that procedure is database provider specific, and will not be explained further here[18]. The transitive closure can be used for subsumption testing, for example, to find all descendants of |*form (AU qualifier)*|.

The transitive closure table can be used to illustrate this:

```
select source, toFsn(source)
from v3_tc
where dest = (
    select distinct conceptid from rf2_ss_descriptions
    where term = 'form (AU qualifier)'
    and active = 1)
```

The above query returns all 231 descendants of |*form (AU qualifier)*| as opposed to just 80 children[19], had we simply looked at the destination relationships of |*form (AU qualifier)*|, shown below:

```
select sourceid, toFsn(sourceid)
from rf2_ss_relationships
where typeid = 116680003 -- is a
and destinationid = (
        select distinct conceptid from rf2_ss_descriptions
    where term =  'form (AU qualifier)'
    and active = 1
) and active = 1
```

.

Note, in order to find all concepts that specifically belong to the logical grouping of the seven notable classes, the methods outlined in Section 5.3.3 are preferred. We advise using the provided refset for the seven notable classes, to avoid unexpected inclusion of concepts across multiple classes.

For example:

|*Miochol-E (acetylcholine chloride 20 mg) solution: powder for intraocular irrigation, vial (trade product unit of use)*| -IS A→ |*medicinal product*|, via:

- |*Miochol-E (acetylcholine chloride 20 mg) solution: powder for intraocular irrigation, vial (trade product unit of use)*| - IS A→ |*acetylcholine 16.1 mg*| *acetylcholine chloride 20 mg solution: powder for intraocular irrigation, vial (medicinal product unit of use)* |
- | *acetylcholine 16.1 mg* |*acetylcholine chloride 20 mg solution: powder for intraocular irrigation, vial (medicinal product unit of use)*| – IS A→ |*acetylcholine (medicinal product)*|
- |*acetylcholine (medicinal product)*| – IS A→ |*medicinal product*|

The above example is a correct inference of IS A relationships. Note that in this case a specific Trade Product Unit of Use concept IS A Medicinal Product, as the seven notable concept hierarchies are not disjoint.

[18] Those interested in an explanation of this procedure should contact the NCTIS by emailing terminologies@nehta.gov.au.

[19] Note that the number of results returned quoted in this document is based on the AMT v3 Beta data.

5.3.5 Fully Specified Names

Every concept in the AMT v3 terminology will have one active description of type |*fully specified name (core metadata concept)*| (FSN).

Note that FSNs represent the reference point for the meaning of the concept, however are not intended to be exposed to users of a clinical system. Refer to Section 7.11 for more details.

The following query shows how this term can be extracted for a given concept:

```
select fsn.term
from rf2_ss_descriptions fsn
where fsn.typeid = (
    select conceptid from rf2_ss_descriptions
    where term = 'fully specified name (core metadata concept)'
    and active = 1)
and fsn.active = 1
and fsn.conceptid = <id of concept>
```

The above query forms the basis of a function created in routines.sql, called toFsn, which simplifies the extraction of an FSN for a concept, as shown below:

```
select toFsn(<id of concept>)
```

5.3.6 Preferred Terms

Every concept in the AMT v3 terminology will have one or more active descriptions of type |*synonym (core metadata concept)*|. Of the synonyms for each concept, one must be designated as the Australian English |*preferred*| description.

This preferred description is the synonym most suitable for use by end users of clinical systems, although other acceptable synonyms referenced in the *Australian English language reference set* may also be used.

The preferred synonym for a given concept is expressed via membership of the *Australian English language reference set*, with the referencedcomponentid column being populated with the description id of the preferred synonym, and the valueid column containing the concept id of the concept |*preferred*|. The following query shows how this reference set can be used to obtain the preferred synonym for a given concept.

```
select preferred.term
from rf2_ss_descriptions preferred

    join rf2_ss_language_refset member
        on member.referencedcomponentid = preferred.id
        and member.active = 1

where member.valueid = (
    select conceptid from rf2_ss_descriptions
    where term = 'Preferred (foundation metadata concept)'
    and active = 1)
and preferred.conceptid = <id of concept>
```

This query forms the basis of another view defined in schema.sql, called v3_ss_pts (i.e. snapshot preferred terms). This view can be used to simplify the extraction of preferred terms, as shown below:

```
select preferred.term
from v3_ss_pts preferred
where preferred.active = 1
and preferred.conceptid = <id of concept>
```

A function, toPt (see routines.sql), is also provided which further simplifies matters:

```
select toPt(<id of concept>)
```

Note: Care should be taken to avoid situations where this function is invoked in high volumes as this will negatively impact on performance. In such cases, joining to the v3_ss_pts table is advised.

5.4 Derived Model

Queries based on the basic schema used so far tend to become verbose, repetitive and difficult to maintain when extracting anything but isolated pieces of data from the AMT v3 model. For complex queries, it is often useful to create a customised schema to provide fast and efficient access to the specific data required for the scenario(s).

The following sections attempt to illustrate the AMT v3 data extraction requirements required to satisfy the core prescribe and dispense use cases. In the main, these use cases are focused on searching MPs, MPPs and TPPs. Some additional derived tables been created to demonstrate this, and is discussed before delving into the use case data queries.

5.4.1 MPP to TPP mapping

In the AMT v3 model, there is an association between MPP concepts and TPP concepts, via the CTPP concept. The following query shows how to determine the associated MPP (and its preferred term) for a given TPP, using the notable concept views discussed in Section 5.3.3.

```
select
    mpp.id as mppid,
    toPt(mpp.id) as mppterm,
    tpp.id as tppid,
    toPt(tpp.id) as tppterm
from v3_tpp tpp

    join rf2_ss_relationships hasTpp
        on hasTpp.sourceid in (
            select id from v3_ctpp)
        and hasTpp.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has tpp (relationship type)'
            and active = 1)
        and hasTpp.destinationid = tpp.id
        and hasTpp.active = 1

    join rf2_ss_relationships isaMpp
        on isaMpp.sourceid = hasTpp.sourceid
        and isaMpp.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'is a (attribute)'
            and active = 1)
        and isaMpp.destinationid in (
            select id from v3_mpp)
        and isaMpp.active = 1

    join v3_mpp mpp
        on mpp.id = isaMpp.destinationid

where tpp.id = 11132011000036100 -- Fosipril 10 mg tablet: uncoated, 30 tablets

group by mpp.id, tpp.id
```

The above query is given a known TPP, 11132011000036100 |*Fosipril 10 mg tablet: uncoated, 30 tablets*|, as its starting point. It uses a join to locate the HAS TPP relationship with a CTPP concept as its source and the known TPP *Fosipril…* as its destination. Likewise, it uses a join to locate the 'is a MPP' relationship that has the same CTPP concept as its source and an MPP concept as its destination. Finally, it joins to the v3_mpp notable view to ensure the identified MPP is active (the view only contains active concepts), and selects the MPP's id and preferred term.

The example above shows how to navigate relationships in the AMT v3 model to retrieve related pieces of information, in this case, from TPP to CTPP to MPP. However, in terms of the basic pack-based prescribing and dispensing use cases, the CTPP concept is not used[20]. Therefore, navigating the CTPP relationships (HAS TPP and 'is a MPP') to determine the TPPs for an MPP (and vice versa) would hinder query readability, maintainability and performance.

The derived model schema (schema/derived-model/schema.sql) has flattened that part of the model by creating a table (v3_mpp_to_tpp) which reflects only the MPP to TPP associations, making that data easier to query. This is a good example of how adding specialised tables to the schema can make querying the terminology less cumbersome while optimising query performance at the same time.

The insert-select statement which queries the basic schema to populate the v3_mpp_to_tpp table (schema/derived-model/importTables.sql) is based on the query structure described above, and allows that query to be rewritten as:

```
select mppid, mppterm from v3_mpp_to_tpp where tppid = 11132011000036100
```

5.4.2 Unit of Use

The extraction of the unit of use size and quantity for a given MPP is another good candidate for derived schema. In order to extract that data, a query would need to navigate the MPP HAS MPUU relationship, the MPUU HAS UNIT OF USE relationship, the *Unit of use size reference set* and the *Unit of use quantity reference set*. By writing this query once to populate a derived schema table, the task of creating business queries to extract unit of use data is greatly simplified. An example of this query is shown below:

```
select
    mpp.id as mppid,
    toPt(mpp.id) as mppterm,
    mpuu.id as mpuuid,
    toPt(mpuu.id) as mpuuterm,
    uouSize.operatorid as sizeoperatorid,
    toPt(uouSize.operatorid) as sizeoperatorterm,
    uouSize.value as sizevalue,
    uouSize.unitid as sizeunitid,
    toPt(uouSize.unitid) as sizeunitterm,
    uouQty.operatorid as quantityoperatorid,
    toPt(uouQty.operatorid) as quantityoperatorterm,
    uouQty.value as quantityvalue,
    uouQty.unitid as quantityunitid,
    toPt(uouQty.unitid) as quantityunitterm
from v3_mpp mpp

    join rf2_ss_relationships hasMpuu
        on hasMpuu.sourceid = mpp.id
        and hasMpuu.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has mpuu (relationship type)'
            and active = 1)
        and hasMpuu.active = 1

    join v3_mpuu mpuu
        on mpuu.id = hasMpuu.destinationid

    join rf2_ss_relationships hasUnitOfUse
        on hasUnitOfUse.sourceid = mpuu.id
        and hasUnitOfUse.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has unit of use (relationship type)'
            and active = 1)
        and hasUnitOfUse.active = 1

    join rf2_ss_unit_of_use_quantity_refset uouQty
        on uouQty.referencedcomponentid = hasMpuu.id
```

[20] CTPP is not used for the purposes of this example of prescribing/dispensing, however CTPP may be used in prescribing and dispensing. For more details refer to Section 7.4 CTPP versus TPP.

```
        and uouQty.active = 1

    left outer join rf2_ss_unit_of_use_size_refset uouSize
        on uouSize.referencedcomponentid = hasUnitOfUse.id
        and uouSize.active = 1

where mpp.id = 26535011000036103
-- norethisterone 1 mg + ethinyloestradiol 35 microgram tablet, 84 [4 x 21 tablets]
```

Examining this in a little more detail, for a known MPP concept (26535011000036103), the query:

- joins to the snapshot relationships table to find any HAS MPUU relationships for that MPP;
- for each HAS MPUU relationship, the query joins the destination MPUU concept to the v3_mpuu view (which only contains active MPUUs) to ensure the MPUU concept is active;
- for each active MPUU identified, the query joins to the snapshot relationships table to find the HAS UNIT OF USE relationship for that MPUU;
- the query joins to the v3_unit_of_use_quantity_refset table to find the reference set member which relates to the 'has mpuu' relationship above; and finally
- the query joins to the v3_unit_of_use_size_refset table to find the reference set member which relates to the HAS UNIT OF USE relationship above.

The insert-select statement which queries the basic schema to populate the v3_unit_of_use table (schema/derived-model/importTables.sql) is based on the query structure described above, and allows that query to be rewritten as:

```
select * from v3_unit_of_use where mppid = 26535011000036103
```

5.4.3 Ingredient Strength

The extraction of the ingredients and their respective strengths for a given MPP is another good candidate for derived schema. In order to extract that data, a query would need to navigate the MPP HAS MPUU relationship, the MPUU HAS INTENDED ACTIVE INGREDIENT relationship, the MPUU HAS AUSTRALIAN BoSS relationship, and finally the *Strength reference set*. By writing this query once to populate a derived schema table, the task of creating business queries to extract ingredient strengths is greatly simplified. An example of this query is shown below:

```
select
    mpp.id as mppid,
    toPt(mpp.id) as mppterm,
    hasMpuu.destinationid as mpuuid,
    toPt(hasMpuu.destinationid) as mpuuterm,
    hasIngredient.destinationid as substanceid,
    toPt(hasIngredient.destinationid) as substanceterm,
    hasAuBoss.destinationid as bossid,
    toPt(hasAuBoss.destinationid) as bossterm,
    strength.operatorid as operatorid,
    toPt(strength.operatorid) as operatorterm,
    strength.value as strengthvalue,
    strength.unitid as unitid,
    toPt(strength.unitid) as unitterm
from v3_mpp mpp

    join rf2_ss_relationships hasMpuu
        on hasMpuu.sourceid = mpp.id
        and hasMpuu.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has mpuu (relationship type)'
            and active = 1)
        and hasMpuu.active = 1
```

```
    join rf2_ss_relationships hasIngredient
        on hasIngredient.sourceid = hasMpuu.destinationid
        and hasIngredient.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has intended active ingredient (attribute)'
            and active = 1)
        and hasIngredient.active = 1

    join rf2_ss_relationships hasAuBoss
        on hasAuBoss.sourceid = hasMpuu.destinationid
        and hasAuBoss.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has australian boss (relationship type)'
            and active = 1)
        and hasAuBoss.relationshipgroup = hasIngredient.relationshipgroup
        and hasAuBoss.active = 1

    left outer join rf2_ss_strength_refset strength
        on strength.referencedcomponentid = hasAuBoss.id
        and strength.active = 1

where mpp.id = 26535011000036103
-- norethisterone 1 mg + ethinyloestradiol 35 microgram tablet, 84 [4 x 21 tablets]
```

Note that the strengths returned by this query are the normalised to a denominator of one, as expressed in the *Strength reference set*. To obtain the strengths as represented in the MPUU descriptions, please refer to the technique described in Section 5.4.4.

Examining this in a little more detail, for a known MPP concept (26535011000036103), the query:

- Joins to the snapshot relationships table to find any HAS MPUU relationships for that MPP.
- For each MPUU identified (as the destination of the HAS MPUU relationship), the query joins to the snapshot relationships table to find any HAS INTENDED ACTIVE INGREDIENT relationships for that MPUU.
- The query then joins to the snapshot relationships table to find the HAS AUSTRALIAN BoSS relationship from the same MPUU **with the same relationshipgroup** as the HAS INTENDED ACTIVE INGREDIENT relationship above.
  NOTE: The boss is the actual substance included in the MPUU which *may* differ from the active component of the substance. For example:
  - HAS INTENDED ACTIVE INGREDIENT = Dexamphetamine (base)
  - HAS AUSTRALIAN BoSS = Dexamphetamine sulfate (salt)
- Finally, the query joins to the v3_strength_refset table to find the strength reference set member which relates to the HAS AUSTRALIAN BoSS relationship above.

The insert-select statement, which queries the basic schema to populate the v3_ingredient_strength table (schema/derived-model/importTables.sql), is based on the query structure described above, and allows that query to be rewritten as:

```
select * from v3_ingredient_strength where mppid = 26535011000036103
```

5.4.4 Combining Strength and Unit of Use Size

The extraction of the total quantity of each ingredient contained in an MPUU is another query added to the derived schema.

This query combines the *Strength reference set* with the *Unit of use size reference set*, and may be used to determine:

- The total quantity of an ingredient in the MPUU – for example if the strength is '15 mg/mL' and the unit of use size is '20 mL' then the total ingredient quantity in the MPUU is 300 mg.
- The denormalised strength as represented in the MPUU descriptions, which are always based on the unit of use size. For example if the strength is '15 mg/mL' and the unit of use size is '20 mL' then the strength represented in the MPUU description will be '300 mg / 20 mL'.

In order to extract that data, a query would need to navigate:

- the MPUU HAS AUSTRALIAN BoSS relationship;
- the MPUU HAS UNIT OF USE relationship;
- the *Strength reference set*;
- the *Unit of use size reference set*; and
- the Composite Unit Of Measure relationships HAS NUMERATOR UNITS and HAS DENOMINATOR UNITS.

By writing this query once to populate a derived schema table, the task of creating business queries to extract ingredient strengths is greatly simplified. An example of this query is shown below:

```
select
    mpuu.id,
    toPt(mpuu.id) as mpuu,
    toPt(hasAuBoss.destinationid) as boss_substance,
    concat(strength.value, ' ', toPt(strength.unitid)) as strength,
    toPt(hasActiveIngredient.destinationid) as active_ingredient,
    concat(uouSize.value, ' ', toPt(uouSize.unitid)) as unit_of_use_size,
    concat(round(strength.value * uouSize.value, 6), ' ',
toPt(hasNumeratorUnits.destinationid)) as total_ingredient_quantity

from v3_mpuu mpuu

    join rf2_ss_relationships hasUnitOfUse
        on hasUnitOfUse.sourceid = mpuu.id
        and hasUnitOfUse.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has unit of use (relationship type)')
        and hasUnitOfUse.active = 1

    join rf2_ss_unit_of_use_size_refset uouSize
        on uouSize.referencedcomponentid = hasUnitOfUse.id
        and uouSize.active = 1

    join rf2_ss_relationships hasAuBoss
        on hasAuBoss.sourceid = mpuu.id
        and hasAuBoss.typeid in (
            select conceptid from rf2_ss_descriptions
            where term = 'has australian boss (relationship type)')
        and hasAuBoss.active = 1

    join rf2_ss_strength_refset strength
        on strength.referencedcomponentid = hasAuBoss.id
        and strength.active = 1

    join rf2_ss_relationships hasActiveIngredient
        on hasActiveIngredient.sourceid = mpuu.id
        and hasActiveIngredient.typeid in (
            select conceptid from rf2_ss_descriptions
            where term = 'has intended active ingredient (attribute)')
        and hasActiveIngredient.relationshipgroup = hasAuBoss.relationshipgroup
        and hasActiveIngredient.active = 1

    join rf2_ss_relationships hasNumeratorUnits
        on hasNumeratorUnits.sourceid = strength.unitid
        and hasNumeratorUnits.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has numerator units (attribute)')
        and hasNumeratorUnits.active = 1
```

```
        join rf2_ss_relationships hasDenominatorUnits
            on hasDenominatorUnits.sourceid = strength.unitid
            and hasDenominatorUnits.typeid = (
                select conceptid from rf2_ss_descriptions
                where term = 'has denominator units (attribute)')
            and hasDenominatorUnits.active = 1

    where mpuu.id = 22148011000036103;
```

Examining this in a little more detail, for a known MPUU concept (22148011000036103), the query:

- joins to the snapshot relationships table to find the HAS UNIT OF USE relationship for that MPUU;
- joins to the v3_unit_of_use_size table to find the *Unit of use size reference set* member which relates to the HAS UNIT OF USE relationship above;
- joins to the snapshot relationships table to find the HAS AUSTRALIAN BoSS relationship for that MPUU;
- joins to the v3_strength_refset table to find the *Strength reference set* member which relates to the HAS AUSTRALIAN BoSS relationship above;
- joins to the snapshot relationships table to find the HAS INTENDED ACTIVE INGREDIENT relationship for the MPUU, which is role-grouped to the HAS AUSTRALIAN BoSS relationship above;
- joins to the snapshot relationships table to find the HAS NUMERATOR UNITS relationship for the composite unit of measure concept identified by the *Strength reference set* member above;
- joins to the snapshot relationships table to find the HAS DENOMINATOR UNITS relationship for the composite unit of measure concept identified by the *Strength reference set* member above; and finally
- multiplies the strength value by the unit of use size value to compute the total ingredient quantity.

Both the HAS AUSTRALIAN BoSS and HAS INTENDED ACTIVE INGREDIENT relationships are required as the strength of the intended active ingredient is expressed in terms of the Basis of Strength Substance (BoSS). That is:

- the HAS INTENDED ACTIVE INGREDIENT identifies the intended ingredient in the medication; and
- the HAS AUSTRALIAN BoSS identifies the substance the strength of the ingredient is expressed in terms of.

Refer to Sections 2.3.4.1 and 2.3.5.10 for more details on strength representation and BoSS.

The insert-select statement which queries the basic schema to populate the v3_ingredient_strength table (schema/derived-model/importTables.sql) is based on the query structure described above, and allows that query to be rewritten as:

```
select * from v3_total_ingredient_quantity where mpuuid = 22148011000036103;
```

Note: There is an inconsistency in the AMT v3 Beta data with respect to the way unit of use size has been modelled, which means that for some MPUU concepts the total ingredient quantity cannot be reliably calculated. These primarily affect injections and powders.

As an example, MPUU 22420011000036103

- sodium bicarbonate 1.76 g + citrate sodium anhydrous 630 mg + citric acid 720 mg + tartaric acid 890 mg oral liquid: powder for, 4 g sachet

The unit of use size of this example is represented as ‘4 g’, whereas it should be ‘1 sachet’. As a result, the total ingredient quantity calculated for this MPUU for ingredient citric acid is actually 2880 mg, instead of 720 mg. This is expected to be rectified for the production release.

5.5 Sample queries by use case

Having introduced the basic RF2 schema and the derived model tables, the following sections look at how these tables, views and procedures can be used to extract data to fulfil the core AMT v3 use cases – prescribing and dispensing. For further details on the content of these use cases, please refer to *AMT v3 Overview and Use Cases* [1].

The queries discussed in this section are contained in sql/use-cases.sql.

5.5.1 Queries to support the Prescribing use case

The prescribing use case states that in prescribing medications, *the ‘prescribing system software searches for products based on AMT MPUU, MPP, TPUU, TPP & CTPP’*. For simplicity, the derived model used in these samples primarily focuses on prescribing by MPP and TPP. The product pack level has been chosen as it strikes a good balance between illustrating the product components, within the context of the v3 model, and targeting a level of refinement that is most intuitively prescribable. Where prescription by CTPP, MPUU or TPUU is required, these samples can be easily adapted via the relationships from, and between, MPP and TPP.

In the following example, the prescriber has decided to prescribe amoxycillin, and has entered the characters ‘a-m-o-x’ into the medications search field of their prescribing system. While this is a simplified example, a real world implementation would likely dynamically query and refine these results as the user types each character. Further consideration to performance would also be required. For the intent of this example we will illustrate a simple, transactional, text search.

The system will display a list of MPPs and TPPs for which their preferred term contains a word commencing with the characters ‘amox’ *or* they contain a substance for which its preferred term commences with the characters ‘amox’. The majority of the task can be achieved by simply querying the derived model tables v3_mpp_to_tpp and v3_ingredient_strength as follows:

```
select
        v3_mpp_to_tpp.mppid,
        v3_mpp_to_tpp.mppterm,
        v3_mpp_to_tpp.tppid,
        v3_mpp_to_tpp.tppterm
from v3_mpp_to_tpp
    join v3_ingredient_strength
        on v3_mpp_to_tpp.mppid = v3_ingredient_strength.mppid
where v3_ingredient_strength.substanceterm regexp (

    @search_term:='(^|[^a-zA-Z]+)amox' collate utf8_unicode_ci

    )
    or v3_mpp_to_tpp.mppterm regexp @search_term
    or v3_mpp_to_tpp.tppterm regexp @search_term

group by v3_mpp_to_tpp.tppid, v3_mpp_to_tpp.mppid
```

The above query makes use of the derived model mentioned above. It primarily searches the v3_ingredient_strength table as it is already populated with terms for the MPPs, MPUUs and substances within them. We additionally join to the v3_mpp_to_tpp table to provide the ability to search the preferred term of the TPP (in the case where the user has entered some trade/brand specific text). The query then performs the search on the relevant terms. You will note that the query uses a regular expression for the search criteria. This has been used as a concise form of querying any term beginning with (^) or containing a word beginning with ([^a-zA-Z]+), followed by the text 'amox'. Additionally, a user defined parameter '@search_term' has been used for brevity.

5.5.2 Queries to support the Dispensing use case

The dispensing use case states that in choosing medications, the product selections are 'based on TPP concepts', and that the dispenser 'dispenses and supplies the medication in accordance with the prescription presented by the patient'.

In the following example, the prescription is for AMT concept 12809011000036105, which is a TPP concept with preferred term 'Amoxil 250 mg capsule: hard, 20 capsules'. The prescription indicates that a generic alternative is acceptable.

The dispensing system must find the generic form of the prescribed medication (MPP), and then find all trade product packs (TPP) which are associated with that MPP[21]. Using the derived model table v3_mpp_to_tpp, a simple query can be written to present a list of appropriate TPPs for dispensing. The query goes further to also return all CTPPs that are associated with those TPPs.

```
select
    mppToTpp.tppid as tppid,
    mppToTpp.tppterm as tppterm,
    hasTpp.sourceid as ctppid,
    toPt(hasTpp.sourceid) as ctppterm
from v3_mpp_to_tpp mppToTpp

    join (
        select mppid
        from v3_mpp_to_tpp
        where tppid = 12809011000036105
        and tppterm = 'Amoxil 250 mg capsule: hard, 20 capsules'
    ) as t on t.mppid = mppToTpp.mppid

    join rf2_ss_relationships hasTpp
        on hasTpp.destinationid = mppToTpp.tppid
        and hasTpp.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has TPP (relationship type)')
        and hasTpp.active = 1
```

In the above query, the join-select finds the MPP associated with the prescribed TPP. The query matches on both the AMT SNOMED CT Identifier and the preferred term. The query is joining on both id and term to highlight the requirement for safety checks, ensuring that the id for the drug being dispensed is consistent with the term the prescriber used when selecting that drug. The outer query extracts all TPPs which map to the MPP identified by the temporary join table, and presents their corresponding CTPP(s) to the dispenser.

[21] Note that AMT v3 does not provide bio-equivalence.

5.6 Additional queries

5.6.1 Extracting dose form

An MPP forms the aggregation of one or more MPUUs, with the addition of pack quantities. Each MPUU has a |*manufactured dose form*| associated with it. While the extraction of dose form does not directly address the core use cases for v3, it *may* help decision support at the time of both prescribing and dispensing.

The query below seeks to extract each of the substances associated with the MPUUs contained within four sample MPPs. For each substance, the query returns its manufactured dose form for that MPUU.

```
select
    mpp.id,
    toPt(mpp.id),
    toPt(hasAuBoss.destinationid),
    toPt(hasDoseForm.destinationid)
from v3_mpp mpp

    inner join rf2_ss_relationships hasMpuu
        on hasMpuu.sourceid = mpp.id
        and hasMpuu.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has mpuu (relationship type)')
        and hasMpuu.destinationid in (
            select id from v3_mpuu)
        and hasMpuu.active = 1

        inner join rf2_ss_relationships hasAuBoss
        on hasAuBoss.sourceid = hasMpuu.destinationid
        and hasAuBoss.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has Australian BoSS (relationship type)'
               and active = 1)
        and hasAuBoss.active = 1

    inner join rf2_ss_relationships hasDoseForm
        on hasDoseForm.sourceid = hasMpuu.destinationid
        and hasDoseForm.typeid = (
            select conceptid from rf2_ss_descriptions
            where term = 'has manufactured dose form (relationship type)')
        and hasDoseForm.active = 1

where mpp.id in (
        26624011000036107, -- "amoxycillin 100 mg/mL oral..."
        51572011000036101, -- "goserelin 3.6 mg implant [..."
        26781011000036107, -- "peginterferon alfa-2a 135 ..."
        28051011000036109  -- "peginterferon alfa-2b 150 ..."
)
```

MPP ID	MPP PT	Substance	Dose Form
26624011000036107	amoxycillin 100 mg/mL oral liquid: powder for, 20 mL	amoxycillin	oral liquid: powder for
26781011000036107	goserelin 3.6 mg implant [1 implant] (&) bicalutamide 50 mg tablet [28 tablets], 1 pack	goserelin	implant
26781011000036107	goserelin 3.6 mg implant [1 implant] (&) bicalutamide 50 mg tablet [28 tablets], 1 pack	bicalutamide	tablet

MPP ID	MPP PT	Substance	Dose Form
28051011000036109	peginterferon alfa-2a 135 microgram/0.5 mL injection [4 x 0.5 mL prefilled syringes] (&) ribavirin 200 mg tablet [112 tablets], 1 pack	peginterferon alfa-2a	injection
28051011000036109	peginterferon alfa-2a 135 microgram/0.5 mL injection [4 x 0.5 mL prefilled syringes] (&) ribavirin 200 mg tablet [112 tablets], 1 pack	ribavirin	tablet
51572011000036101	peginterferon alfa-2b 150 microgram injection [4 x 150 microgram cartridges] (&) ribavirin 200 mg capsule [196 capsules] (&) inert substance diluent [4 x 0.5 mL cartridges], 1 pack	peginterferon alfa-2b	injection
51572011000036101	peginterferon alfa-2b 150 microgram injection [4 x 150 microgram cartridges] (&) ribavirin 200 mg capsule [196 capsules] (&) inert substance diluent [4 x 0.5 mL cartridges], 1 pack	ribavirin	capsule
51572011000036101	peginterferon alfa-2b 150 microgram injection [4 x 150 microgram cartridges] (&) ribavirin 200 mg capsule [196 capsules] (&) inert substance diluent [4 x 0.5 mL cartridges], 1 pack	inert substance	diluent

6 Testing and conformity

NEHTA already publishes compliance information in relation to mapping clinical terminologies: in the *AMT Mapping Guidelines* [5], and *AMT Mapping Requirements* [6]. Implementers of AMT should refer to these guidelines.

6.1 Testing

When implementing AMT v3 it is imperative to test whether the data implemented in the system matches the original AMT v3 data. Typically this is achieved by running procedures suitable for your application of AMT and then checking the results for a range of AMT concepts against the original AMT v3 source data loaded into the system.

6.2 Compliance and Conformance

Additional documentation is under development and will be published in the near future by NEHTA's Compliance, Conformance and Accreditation (CCA) Programme team. These publications will include

- Conformity Assessment Process – describing both compliance and conformance assessment processes;
- Compliance Requirements – including requirements on licensing, assessors and native implementation;
- Conformance Requirements; and
- Conformance Test Specification.

7 Implementation considerations

7.1 AMT's purpose

An important consideration when implementing the AMT is to recognise the AMT's intended purpose and corresponding limitations. The AMT is a terminology, not a medicines database; therefore depending upon how the AMT is being used, it may be necessary to use the AMT in conjunction with knowledge-support and decision-support data and functionality.

If this is necessary it will involve either mapping to existing medicine data files, or creating new data files using AMT as a base, and then authoring or procuring additional medicine data to support the intended use. It is not the purpose of this guide to provide specific guidance on additional knowledge and decision support data. However some additional information can be found in the *AMT Mapping Guidelines* [5].

Examples of content not provided by AMT, but provided by other medicines data files are:

- Physiological equivalence
- Adverse effects
- Counselling instructions
- Cautionary and advisory label recommendations
- Contraindications
- Dose checking
- Drug-Drug interactions
- Drug-Allergy interactions
- Drug-Food interactions
- Indications
- Normal dose ranges
- Precautions for use
- Storage or supply chain related information

7.2 Product availability

Initial inclusion of a medicine product in AMT is based on its registration by a sponsor with the TGA's Australian Register of Therapeutic Goods (ARTG). However this does not mean the product is available in the Australian supply chain.

Equally when a product is removed from the supply chain and no longer available for sale, the corresponding AMT concepts are not retired or deprecated (i.e. they remain as active).

This is because AMT is a terminology describing known concepts, not a product database. The concepts are still needed to support existing health records or eHealth messages regardless of the current day availability of the product.

Product availability information must be sourced outside AMT.

7.3 AMT and SNOMED CT substance concepts

Substance concepts used in the AMT are unique to the AMT, and are a different set of concepts than used by the International Edition of SNOMED CT.

This is partially due to the AMT's parallel development to SNOMED CT, and partially due to a concern at the time of creation of the AMT that anticipated future changes to the SNOMED CT substance hierarchy may destabilise the AMT.

Implementers of the AMT should note this separation, and use AMT substances for AMT-related activity. The NCTIS publishes a map between AMT substances and SNOMED substances, and this can be used to facilitate interoperability for allergy recording and decision support purposes.

Please also note that AMT substances do not contain non-medicine allergens, these must be sourced from SNOMED CT International.

7.4 CTPP versus TPP

When choosing whether to use CTPP or TPP concepts, or both, in an implementation it is important to consider the implementation's use cases.

Where references to trade packs are required without specifying container type, TPP concepts should be used. For example, pack-based prescribing usually requires specifying a product pack, however it is unnecessary to specify a container. Presenting users with a variety of container based variations of a pack (bottle, blister pack etc.) may frustrate users with irrelevant options and slow data entry unnecessarily.

However under some circumstances, clinicians may need to specify specific containers when prescribing. Similarly the more specific CTPP concept is required when recording a specific dispensed medication.

Therefore it is necessary to analyse system requirements before choosing when CTPP concepts, TPP concepts, or both are appropriate.

7.5 Parsing descriptions

The descriptions contained in AMT are structured according to editorial rules, as defined in the *AMT v3 Editorial Rules* [3].

While AMT's descriptions are very structured, they are not intended to be parsed into smaller components. Parsing AMT descriptions presents risk and is strongly discouraged. Required atomic data should be sourced from the appropriate source within the AMT model. For examples of extracting ingredient strengths, please refer to Section 5.4.3.

7.6 Field length

There is a maximum field length of 2,048 characters specified for the term field in AMT. However in practice the current longest AMT terms are just over 1,000 characters. Depending upon the section of AMT content being used, the longest terms actually present in AMT may be significantly shorter than this limit.

As an indication, the mean length of AMT v3 Beta release active preferred terms is 51 characters, the median is 49 characters and the mode is 56 characters. Of the AMT v3 Beta release preferred terms, 95% are less than 95 characters.

The graph below shows the distribution of AMT v3 Beta release active preferred term lengths.

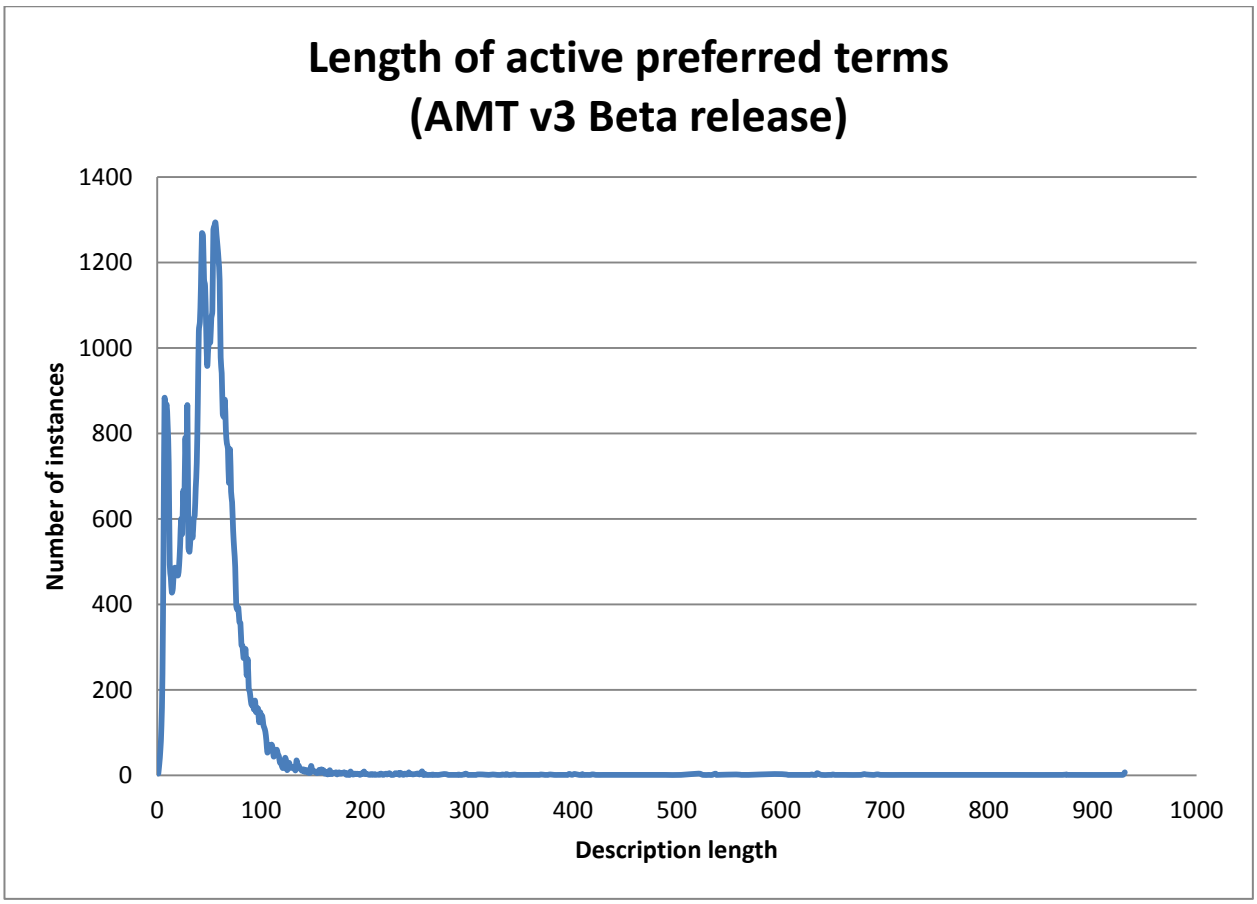


Figure 34: AMT v3 Beta release active preferred term lengths

System developers allocating less than 2,048 characters are advised to check the release files for the maximum description length for the AMT content they use on a per-release basis. This will help ensure that no truncation of AMT descriptions occurs, as truncation may be clinically unsafe.

Any field length restriction for an application that results in truncation of AMT descriptions should lead to a discussion with the NCTIS for implementation guidance and clinical safety considerations.

Assessment and potential development of clinical interface descriptions for AMT concepts is on the *AMT roadmap* [7], which will also consider label names, sort order and ingredient order.

7.7 Modifying or extending the AMT

7.7.1 Modifying the AMT

Similar to the use of SNOMED CT, AMT content may not be modified. This includes AMT concepts, descriptions, relationships and reference sets.

AMT's integrity must be maintained as distributed when it is implemented into local systems. That is relationships between AMT concepts, codes and descriptions will not be edited or distorted in use.

This restriction also precludes editing any AMT description while it remains linked to its associated AMT code. If an AMT concept is not suitable for use for any reason, it should be discarded in its entirety and a local concept used in its place.

7.7.2 Extending AMT

It is possible to extend AMT, much the same way as it is possible to extend SNOMED CT.

The simplest type of extension is to develop custom reference sets of AMT content for specific purposes. AMT is published with seven reference sets, one for each of the 'notable concept classes' which contains all active AMT content from these classes. These reference sets are provided as a convenience; however other more refined reference sets may be useful in specific implementations.

Adding synonyms to AMT concepts is another simple extension, short names for example. Although use of distributed AMT descriptions should be the default approach, in some situations alternate text may be required. In this case it is possible to add new synonyms to AMT concepts as required. Note that the Language reference set provided with AMT will need to be extended if the additional synonyms are to be considered the preferred term in this setting.

It is also possible to add additional concepts to AMT in an extension. This process is not trivial, however it allows implementers to add concepts required at a local level that are not applicable nationally.

Concepts may also be added to a local extension that are later ultimately required nationally. However it must be noted that if/when these concepts are authored in AMT, there is additional work for the local extension to identify the local and national identifiers as co-referent and cease modifications of the concept. If the final AMT modelling differs from the extension modelling, this process is further complicated. For these reasons it is usually preferable to request nationally applicable content from the NCTIS and wait for it to be present in AMT if possible.

It is recommended that users wishing to create extensions to AMT contact the NCTIS for guidance and assistance.

7.8 Medicines out of AMT's scope

The AMT is intended to cover products commonly used for human treatment in Australia. There are certain known limitations:

- Herbal medicines
- Clinical trial medicines
- Local specially-manufactured medicines
- Extemporaneous preparations
- Medical devices (although AMT currently contains some medicated devices such as bandages)

The AMT may not include all medicines or products that are required for use across all possible local care settings. Local clinical systems will still need to be able to manage local codes and descriptions, alongside their use of the AMT. This functionality is already present in local systems and drug file management.

NEHTA welcomes requests for additions or changes to AMT content. Refer to the Request Submission process as identified in the NCTIS site (https://nehta.org.au/aht/index.php).

7.9 ARTG identifiers

AMT v2 is distributed with 'ARTG id' descriptions on each CTPP concept. These descriptions provide the Australian Register of Therapeutic Goods identifier for each CTPP concept in AMT v2.

The 'ARTG id' description type has been removed in AMT v3, and therefore ARTG identifiers are not provided in AMT v3. ARTG identifiers may be added into future versions of AMT as an additional, non-defining reference set.

7.10 Sponsor information

AMT v2 has a series of 'sponsor' concepts which represent the sponsor organisations registered with medicines products on the Australian Register of Therapeutic Goods. These sponsor concepts are associated with the TP concepts in AMT v2, so it is possible to identify the sponsor for a trade medication using AMT v2.

Sponsors have been removed from the AMT v3 model as they were not required to support AMT use cases. Access to sponsor information for medications is available via the Australian Register of Therapeutic Goods.

7.11 Description types

AMT v3 contains two different types of description:

- Fully Specified Name
- Synonym

Fully Specified Names are formal descriptions of each concept and **should not** be rendered to normal clinical end users.

Synonyms are intended to be used by end users. Some Synonyms are identified in the *Australian English language reference set* as 'preferred' and others 'acceptable', which indicate their preferred use in Australia. Other language reference sets express preferences in different locations/contexts.

- 'Preferred' Synonyms (also known as Preferred Terms) are intended to be rendered to end users when selecting a concept.
- 'Acceptable' Synonyms are intended to be used when searching for a concept, but not used as the text presented as an option for a user to select when entering data.

Refer to Section 3.1.7 for more on Language reference sets.

7.12 AMT and PBS data

As of December 2012, PBS data includes AMT identifiers and preferred terms for:

- Medicinal Products;
- Medicinal Product Packs; and
- Trade Product Packs.

PBS data uses AMT Medicinal Product Packs as standardised descriptors of dose form and strength: see *Pharmaceutical Benefits Scheme (PBS)* [8].

7.13 Identifiers

Conformant to RF2, AMT v3 uses SNOMED CT Identifiers for identification of all components other than members of reference sets which are identified using UUIDs.

UUIDs previously used in AMT v2 as alternative identifiers for components (concepts, descriptions and relationships) have been carried forward into AMT v3 for backward compatibility. UUIDs for AMT v2 components can be found in the AMT v3 RF2 Identifiers file (refer to Section 3.1.5) traceable to the component. Components newly generated in AMT v3 do not have these additional UUIDs.

SNOMED CT Identifiers do have a structure, built up of a namespace, sequence, partition identifier and check digit. This is explained in Section 4.3.2 Representing SNOMED CT Identifiers of the *SNOMED CT Technical Implementation Guide* [4].

While SNOMED CT Identifiers are constructed in this manner, the structure of SNOMED CT Identifiers should not be used to infer any meaning. Elements making up SNOMED CT Identifiers are included for infrastructural purposes, such as namespaces used to avoid collisions and enable decentralised allocation of SNOMED CT Identifiers. SNOMED CT Identifiers should be treated as an opaque, unique identifier.

Finally SNOMED CT Identifiers should not be rendered to users of clinical systems. Although used in data structures and records, typical users should never see SNOMED CT Identifiers.

7.14 Sufficiency of floating point strength accuracy

When normalising values to populate the value field of the *Strength reference set* (Section 2.3.4.1), the result includes real numbers that forever repeat after the decimal point (e.g. 0.333333333...). It was determined that up to 13 significant figures are required to arrive at a sufficiently accurate strength value when calculating the non-normalised strength attribute, e.g. 16666.66666667, 33333.33333333 and 149.25373134.

For example the strength value an MPUU of ‘epoetin beta 5000 international units/0.3 mL injection, syringe’ is ‘16666.66666667’. This product has a Unit of use size value of ‘0.3’, thus recalculating non-normalised strength results in ‘5000.000000001 international units/0.3 mL’ which affords sufficient accuracy.

Note that if arithmetic is performed using AMT v3 floating point strengths, issues relating to rounding of recurring numbers may be encountered and should be taken into account. The following table of MPUUs provides examples which can be used for testing.

MPUU ID	MPUU Preferred Term	Basis of Strength Substance	Strength Value	Strength Unit
21995011000036101	epoetin beta 4000 international units/0.3 mL injection, syringe	epoetin beta	13333.33333333	international unit/mL

MPUU ID	MPUU Preferred Term	Basis of Strength Substance	Strength Value	Strength Unit
21996011000036108	epoetin beta 5000 international units/0.3 mL injection, syringe	epoetin beta	16666.66666667	international unit/mL
22082011000036102	follitropin beta 900 international units/1.08 mL injection, cartridge	follitropin beta	833.33333333	international unit/mL
23132011000036108	follitropin beta 300 international units/0.36 mL injection, cartridge	follitropin beta	833.33333333	international unit/mL
23133011000036101	follitropin beta 600 international units/0.72 mL injection, cartridge	follitropin beta	833.33333333	international unit/mL
23315011000036101	anakinra 100 mg/0.67 mL injection, syringe	anakinra	149.25373134	mg/mL
82931011000036102	epoetin beta 200 microgram/0.3 mL injection, syringe	epoetin beta	666.66666667	microgram/mL
82932011000036108	epoetin beta 100 microgram/0.3 mL injection, syringe	epoetin beta	333.33333333	microgram/mL
82935011000036100	epoetin beta 50 microgram/0.3 mL injection, syringe	epoetin beta	166.66666667	microgram/mL
933220001000036107	benzathine benzylpenicillin 1.2 million units/2.3 mL (900 mg/2.3 mL) injection, syringe	benzathine benzylpenicillin	391.30434783	mg/mL

MPUU ID	MPUU Preferred Term	Basis of Strength Substance	Strength Value	Strength Unit
23019011000036103	oestradiol 50 microgram/24 hours + norethisterone acetate 250 microgram/24 hours patch	norethisterone acetate	10.41666667	microgram/hour
45136011000036102	lignocaine hydrochloride anhydrous 2% (36 mg/1.8 mL) + adrenaline 1 in 80 000 (27.5 microgram/1.8 mL) injection, cartridge	adrenaline	15.27777778	microgram/mL

7.15 Viewer application

The AMT v3 Beta has currently been published without a viewer application to allow browsing of the data.

This is under consideration for the Production release of AMT v3.

7.16 Non-breaking spaces

Space characters between strength values and units in AMT v3 descriptions are intended to be non-breaking spaces.

In AMT v3 Production these characters will be replaced by non-breaking whitespace characters (UTF-8 0xC2 0xA0), which it will be necessary to honour in rendering these descriptions.

8 References

1. NEHTA. *AMT v3 Overview and Detailed Business Use Cases*. Sydney: NEHTA; 2013. dv003. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

2. IHTSDO. *RF2 Specification Change Request: Addition of Concrete Domains*. Copenhagen: IHTSDO; 2011. Available from: N/A.

3. NEHTA. *AMT Editorial Rules v3 Model*. Sydney: NEHTA; 2012. v005. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

4. IHTSDO. *SNOMED CT Technical Implementation Guide*. Copenhagen: IHTSDO; 2012. July 2012 release. Available from: http://www.snomed.org/doc.

5. NEHTA. *AMT Mapping Guidelines*. Sydney: NEHTA; 2012. rev001. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

6. NEHTA. *AMT Mapping Requirements*. Sydney: NEHTA; 2012. rev001. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

7. NEHTA. *AMT 2012 Survey results and development roadmap*. Sydney: NEHTA; 2012. v1.0. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

8. Pharmaceutical Benefits Scheme. *Pharmaceutical Benefits Scheme (PBS).* [Internet]. 2012 [cited 2013 Feb 22]. Available from: http://www.pbs.gov.au/info/home.

9. NEHTA. *AMT v3 Beta Release Background and Context Communique*. Sydney: NEHTA; 2013. Available from: N/A.

10. NEHTA. *AMT Implementation plan 2011-12*. Sydney: NEHTA; 2011. v1.0. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

11. NEHTA. *AMT v3 development approach for reference sets*. Sydney: NEHTA; 2013. Beta release, included in terminology release bundle. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

12. NEHTA. *NCTIS Reference set library*. Sydney: NCTIS; 2012. Release 20121130. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

13. NEHTA. *AMT v2 to v3 Migration Guide*. Sydney: NEHTA; 2012. v1.0. Available from: https://nehta.org.au/aht/index.php?option=com_docman&task=cat_view&gid=21&Itemid=40.

Appendix A Subpacks and component packs

A product pack (MPP, CTPP or TPP) always contains components (MPUUs or TPUUs) in a primary container. The primary container is the lowest-level container (non-ingestible) that immediately surrounds the medicinal product. Examples of a primary container are: blister pack, bottle, vial and cartridge.

Some products may also have a secondary container that envelops the components contained within one or more primary containers. An example of a secondary container is 'carton'. AMT does not include specific information on secondary containers but uses the container type 'composite pack' for all CTPP representing secondary containers.

The component(s) within a secondary container may:

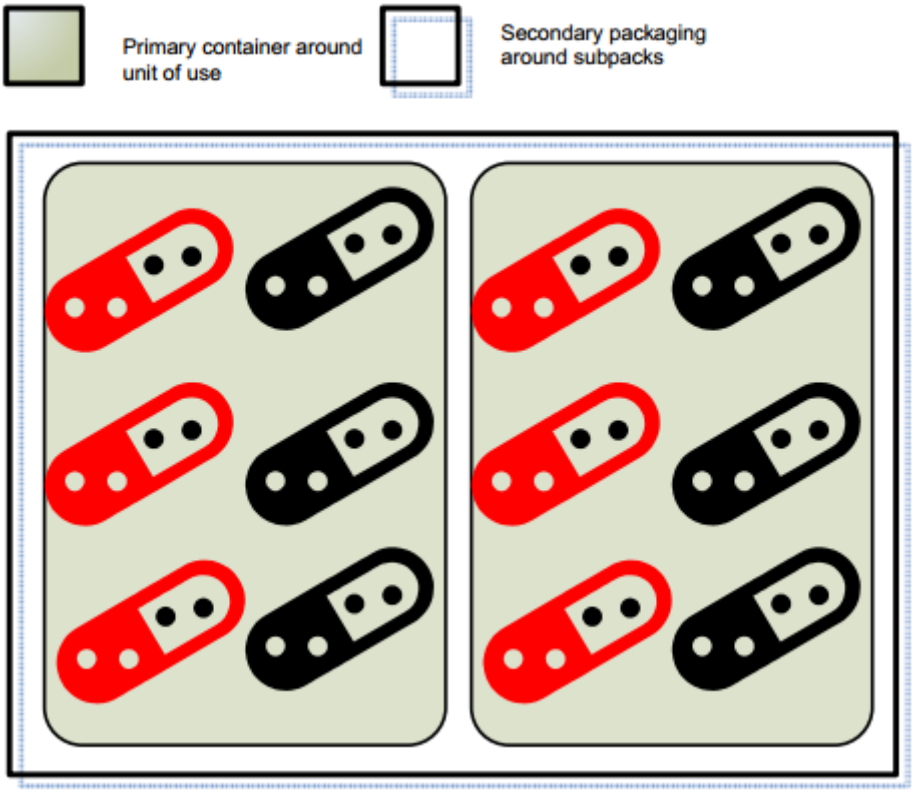
- have the same active ingredients, strength and form;
- have the same active ingredients, different strengths but similar form; or
- have different active ingredients, different strengths but similar form.

A.1 Subpacks

When there are multiple identical representations of the same component(s) within the same type of primary container, the product pack (MPP or CTPP) is said to have subpacks.

Subpacks are only represented for specific product categories. The categories currently include oral contraceptives and hormone replacement therapy products. Subpacks will be added when they are deemed to be required:

- for consistency;
- for clinical reasons; or
- when they are represented as subpacks in the Pharmaceutical Benefits Scheme (PBS).



Examples are

norethisterone 500 microgram + ethinyloestradiol 35 microgram tablet x 21

inert substance tablet x 7

28 tablet subPACK

4 x 28 PACK

and

oestradiol 2 mg tablet x 28

28 tablet subPACK

2 x 28 PACK

A.2 Component Packs

If a product pack contains multiple units of use, with each unit of use contained in a separate primary container, then the product pack is deemed to have component packs. The component packs within a composite product pack typically have different active ingredients. They may have the same form or have different forms.

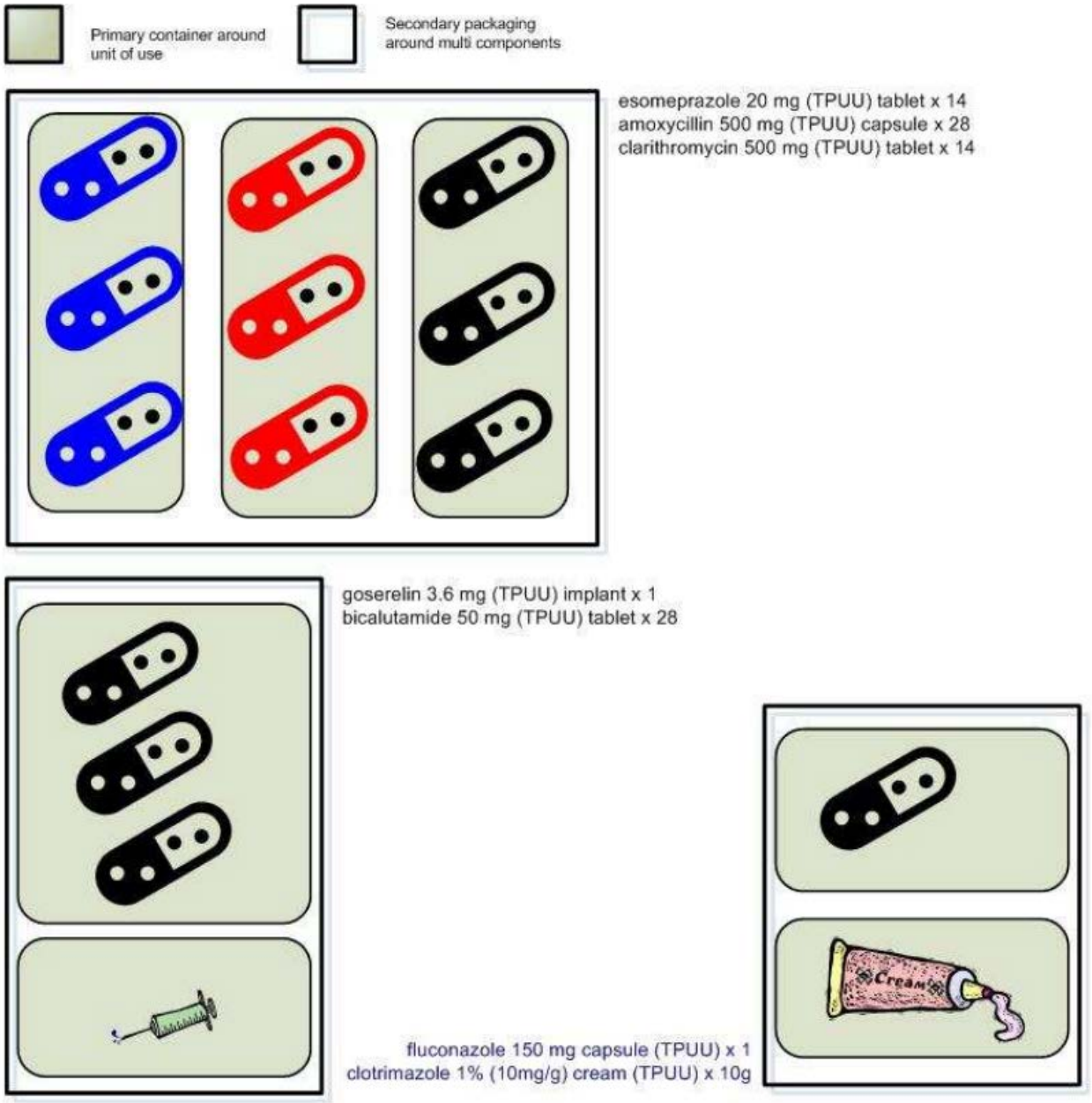
It should be noted that some component packs which are described as an MPP or CTPP may not actually be available as a TPP (i.e. a component in a component pack may only be available as part of the combined CTPP and is not available individually).

The CTPP representing the composite product pack (i.e. containing all the component packs) will have an associated generic container type of 'composite pack'. The CTPP representing a component pack will have an associated specific container type e.g. blister pack, bottle, and ampoule.

For example: Nexium Hp7 is a multi-component product. It will have four CTPPs (component packs): one to represent each of the three components and an additional CTPP to represent the overall product:

- Amoxil 500 mg capsule: hard, 28 capsules, blister pack

- Klacid 500 mg tablet: film-coated, 14 tablets, blister pack
- Nexium 20 mg tablet, 14, blister pack
- Nexium Hp7, 1 pack, composite pack (i.e. this is the overall product)



Appendix B AMT v3 Beta Reference Set Descriptor

The table below shows an example of the reference set descriptor (described in Section 3.1.9.3) delivered in the AMT v3 Beta data. For readability concept identifiers have been converted to preferred terms, and some columns have been omitted for brevity.

Module name	Referenced component name	Attribute description	Attribute type	Attribute order
Australian Medicines Terminology module	*Containered trade product pack reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Containered trade product pack reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Medicinal product pack reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Medicinal product pack reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Medicinal product reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Medicinal product reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Medicinal product unit of use reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Medicinal product unit of use reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*REPLACED BY association reference set*	Association source component	Component type	0
Australian Medicines Terminology module	*REPLACED BY association reference set*	Association target component	Component type	1
Australian Medicines Terminology module	*Strength reference set*	Referenced component	Component type	0
Australian Medicines Terminology module	*Strength reference set*	Unit Id	Concept type component	1
Australian Medicines Terminology module	*Strength reference set*	Operator id	Concept type component	2
Australian Medicines Terminology module	*Strength reference set*	Value	Floating point	3

Module name	Referenced component name	Attribute description	Attribute type	Attribute order
Australian Medicines Terminology module	*Subpack quantity reference set*	Referenced component	Component type	0
Australian Medicines Terminology module	*Subpack quantity reference set*	Unit Id	Concept type component	1
Australian Medicines Terminology module	*Subpack quantity reference set*	Operator id	Concept type component	2
Australian Medicines Terminology module	*Subpack quantity reference set*	Value	Unsigned integer	3
Australian Medicines Terminology module	*Trade product pack reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Trade product pack reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Trade product reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Trade product reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Trade product unit of use reference set*	Referenced component	Concept type component	0
Australian Medicines Terminology module	*Trade product unit of use reference set*	Member type	Concept type component	1
Australian Medicines Terminology module	*Unit of use quantity reference set*	Referenced component	Component type	0
Australian Medicines Terminology module	*Unit of use quantity reference set*	Unit Id	Concept type component	1
Australian Medicines Terminology module	*Unit of use quantity reference set*	Operator id	Concept type component	2
Australian Medicines Terminology module	*Unit of use quantity reference set*	Value	Floating point	3
Australian Medicines Terminology module	*Unit of use size reference set*	Referenced component	Component type	0
Australian Medicines Terminology module	*Unit of use size reference set*	Unit Id	Concept type component	1
Australian Medicines Terminology module	*Unit of use size reference set*	Operator id	Concept type component	2
Australian Medicines Terminology module	*Unit of use size reference set*	Value	Floating point	3

Module name	Referenced component name	Attribute description	Attribute type	Attribute order
SNOMED CT core	*Description format*	Description type component	Concept type component	0
SNOMED CT core	*Description format*	Description format	Concept type component	1
SNOMED CT core	*Description format*	Description length	Unsigned integer	2
SNOMED CT core	*Module dependency*	Dependency target	Concept type component	0
SNOMED CT core	*Module dependency*	Source effective time	Time	1
SNOMED CT core	*Module dependency*	Target effective time	Time	2
SNOMED CT core	*Reference set descriptor*	Referenced component	Concept type component	0
SNOMED CT core	*Reference set descriptor*	Attribute description	Concept type component	1
SNOMED CT core	*Reference set descriptor*	Attribute type	Concept type component	2
SNOMED CT core	*Reference set descriptor*	Attribute order	Unsigned integer	3

Appendix C Glossary

Acronym / Abbreviation	Term	Meaning
	AUSTL	Type of ARTGID assigned to 'listed' items on the Australian Register of Therapeutic Goods. Listed medicines are much lower risk self-medication products than those listed as AUSTR medicines. Listed medicines include sunscreens over SPF4 and many vitamin, mineral, herbal and homoeopathic products.
	AUSTR	Type of ARTGID assigned to 'registered' items on the Australian Register of Therapeutic Goods. Registered medicines include all prescription only medicines and many over-the-counter products such as pain relief, coughs and colds and antiseptic creams.
AMT	**Australian Medicines Terminology**	National terminology that identifies medicines used in Australia, using unique codes to deliver unambiguous, accurate and standardised names for both branded (trade) and generic (medicinal) products.
AMT v3	**Australian Medicines Terminology Version 3**	Latest version of the AMT product, consisting of a series of model and editorial rule changes from the previous versions – AMT v2.x.
ARTG	**Australian Register of Therapeutic Goods**	Australian register on which all therapeutic goods must be registered before they can be lawfully supplied in Australia. Managed by the Therapeutic Goods Administration.
BoSS	**Basis of Strength Substance**	Ingredient strengths are always represented in terms of a Basis of Strength Substance (BoSS). The BoSS may be a base, salt or modified salt, and may be the same or different to the actual ingredient being measured.
	Classifier / reasoner	A piece of software used to perform automated classification of, or reasoning about, description logic rich data such as SNOMED CT or AMT.
COTS	**Commercial Of The Shelf**	Type of product acquisition where the supplied product is commercially sold and can be procured and used for its intended purpose without need for modification or customisation.
CCA	**Compliance, Conformance and Accreditation**	NEHTA programme responsible for developing a national framework assuring systems comply with Australian specifications and demonstrate appropriate standards of interoperability, security and clinical safety in the way they handle and exchange information.
CTPP	**Containered Trade Product Pack**	A category of AMT concepts representing marketable medical entities available for patient use, with details of container type. Refer to Section 2.3.2.6.
DL	**Description Logic**	Family of formal knowledge representation languages used in artificial intelligence for formal reasoning with concepts from a domain. Description logic is the formalism that underpins SNOMED CT's and AMT's machine readable definition of the concepts they contain.

Acronym / Abbreviation	Term	Meaning
DOS	**Disk Operating System**	Name used to identify serveral closely related operating systems used in the IBM compatible PC market through the 1980s and 1990s, particularly Microsoft DOS.
DNF	**Distribution Normal Form**	A normalised inferred form specified by the IHTSDO as the mandatory distribution form for SNOMED CT and extensions. This form specifies that only proximal supertype and non-redundant inherited relationships will be included from the full set of inferred supertypes and relationships.
FSN	**Fully Specified Name**	Type of description (human readable name) given to a concept in SNOMED CT and AMT that is used as the definitive meaning of the concept.
IHTSDO	**International Health Terminology Standards Development Organisation**	International not for profit organisation which owns and administers the rights to SNOMED CT and related terminology standards.
ISO	**International Organization for Standardization**	International standards development organisation
KRSS	**Knowledge Representation System Specification**	Knowledge representation language with a Lisp-like syntax that may be used as an input format to a classifier. SNOMED CT RF1 and RF2 may be transformed into KRSS format
	Language reference set	A reference set pattern specified in the SNOMED CT RF2 specification used to indicate language preferences for SNOMED CT Synonyms in specific contexts, such as dialectic language preferences. Language reference sets are used to express which synonyms are preferred, acceptable and unacceptable from the set of synonyms distributed in the Descriptions file.
MP	**Medicinal Product**	Category of AMT concepts representing abstract formulated representations of therapeutic active ingredients used in treatment of human patients in Australia. Refer to Section 2.3.2.1.
MPP	**Medicinal Product Pack**	Category of AMT concepts representing the abstract concept of marketable medicinal entities available for patient use, devoid of brand and container type. Refer to Section 2.3.2.3.
MPUU	**Medicinal Product Unit of Use**	Category of AMT concepts representing abstract formulations containing active ingredient, strength and form in a single dose form or unit of use component of a multi-component formulation, devoid of brand. Refer to Section 2.3.2.2.
NCTIS	**National Clinical Terminology and Information Service**	Group within NEHTA responsible for development, publication and maintenance of clinical terminology and information products for Australia.
NHS	**National Health Service**	Government organisation responsible for funding and administering publically funded healthcare in the United Kingdom.

Acronym / Abbreviation	Term	Meaning
	Notable concepts	A collective term for the following concepts: • Medicinal Product (MP) • Medicinal Product Unit of Use (MPUU) • Medicinal Product Pack (MPP) • Trade Product (TP) • Trade Product Unit of Use (TPUU) • Containered Trade Product Pack (CTPP) • Trade Product Pack (TPP)
PBS	**Pharmaceutical Benefits Scheme**	Australian government subsidy system for medicines and a supporting regulatory body.
	Preferred Term	A Synonym for a concept designated within a given context as the preferred way of representing or referring to that concept. In AMT this refers to the Synonym for a concept referenced by the Australian English Language Reference Set as the preferred way of referring to that concept. Each concept has one and only one active preferred term at one time point in any given language reference set. Refer to Section 3.1.7.
	Seven notable concepts	See *Notable concepts*.
	SNOMED CT	Systematized Nomenclature of Medicine – Clinical Terms. Parent terminology of AMT v3 and SNOMED CT-AU.
SCTID	**SNOMED CT Identifier**	Identifier scheme defined for distributed components (concepts, descriptions, relationships) of a SNOMED CT or extension release.
RF1	**SNOMED CT Release Format 1**	Original SNOMED CT release file format used to since 2002 and actively being replaced by RF2.
RF2	**SNOMED CT Release Format 2**	New SNOMED CT release file format replacing RF1.
	Stated Form	The form of AMT or SNOMED CT directly stated by the authors, containing no machine inferred statements.
SQL	**Structured Query Language**	Programming language originally based on relational algebra and tuple relational calculus, designed to manage data in relational databases.
	Synonym	Type of description comfortable and natural for use by clinicians and end users. This differs from the Fully Specified Name, which by contrast is intended as the formal, definitive and unambiguous definition of a concept not intended for every day use.
SNOMED CT	**Systematized Nomenclature of Medicine – Clinical Terms**	A systematically organised, computer processable collection of medical terms, providing codes, terms, synonyms and definitions. SNOMED CT is designed to underpin clinical data recording and meaning-based retrieval and use.
TGA	**Therapeutic Goods Administration**	Australia's regulatory authority for therapeutic goods, which among other duties, maintains the Australian Register of Therapeutic Goods.
TP	**Trade Product**	Category of AMT concepts representing product brand names. Refer to Section 2.3.2.4.

Acronym / Abbreviation	Term	Meaning
TPP	**Trade Product Pack**	Category of AMT concepts representing marketable medicinal entities available for patient use devoid of container type. Refer to Section 2.3.2.7.
TPUU	**Trade Product Unit of Use**	Category of AMT concepts representing marketable formulations containing active ingredient, strength and form in a single dose form or unit of use component of a multi-component pack. Refer to Section 2.3.2.5.
UTF-8	**UCS Transform Format – 8-bit**	Variable width encoding format that can represent all Unicode characters.
UCUM	**Unified Code for Units of Measure**	Coding system for units of measure which includes a grammar for coordinating codes into complex unit expressions and conversion factors for units.
UML	**Unified Modelling Language**	Standardised general purpose modelling language aimed at object-orientated software engineering.
	Unit of use	The Unit of Use describes a discrete unit dose form (e.g. tablet, capsule) or a continuous substance where a consistent physically measurable unit or sub-unit cannot be identified (e.g. cream, eye drops).
UCS	**Universal Character Set**	Standardised set of characters defined by ISO/IEC 10646. Used as a basis for many character encodings.
UUID	**Universally Unique Identifier**	An identifier scheme used for distributed allocation of identifiers.
OWL	**Web Ontology Language**	Family of knowledge representation languages for authoring ontologies. SNOMED CT RF1 and RF2 formats may be transformed to OWL Functional or OWL XML formats for consumption by description logic reasoners.