



CORPORATE BUSINESS MODELLING LANGUAGE (CBML)

BASELINE (VERSION 4.2)

10TH MARCH 2009

D/D CBM(A)/7/12/4

APPROVED FOR RELEASE

Intentionally Left Blank

CONTENTS

1	What is CBML?	1-1
	Definition	1-1
	Impact of Defining Information in CBML	1-1
	Different Perceptions of Information	1-2
	Using CBML for Reconciliation	1-2
	Conventions Used Within This Document	1-2
2	The Language Structure of CBML	2-1
	General Concepts	2-1
	Differentiation of Real-World Entities and Real-World Substances	2-1
	Defining Real-World Entities and Real-World Substances	2-1
	Information about Entity Classes and Substance Classes	2-1
	Categorisation Information for Entity Classes and Substance Classes	2-2
	Intrinsic Information for Entity Classes and Substance Classes	2-2
	Constraints on Specialisation Structure and Categorisation	2-2
	CBML Element Definition	2-3
3	Classes and Specialisation Structures	3-1
	Terminology	3-1
	Purpose and Use	3-1
	General Description	3-1
	Usage Rules in Classes	3-3
4	Categories and Categorisation Structures	4-1
	Purpose and Use	4-1
	General Description	4-1
	Categorising Sets	4-2
	Category Division	4-3
5	Specialising a Category	5-1
	Purpose and Use	5-1
	General Description	5-1
6	Categorising of Classes	6-1
	Purpose and Use	6-1
	General Description	6-1
	Usage Rules in the Categorisation of Classes	6-3
7	Intrinsic Information in Class or Category	7-1
	Purpose and Use	7-1
	General Description	7-1
	Structure of Characteristics	7-2
	Usage Rules in Characteristics	7-3
	Uniqueness of Characteristics	7-5
8	Composition, Data Types and Constraints	8-1
	Purpose and Use	8-1
	Internal Structure of a Characteristic	8-1
	Using The Character Data Type	8-2
	Using The Numeric Data Type	8-2
	Using The Boolean Data Type	8-2
	Using The Time Data Type	8-2
	Using The Referential Data Type	8-3
	Valid Characteristic Types	8-3
	Types of referred Characteristics	8-4
	Units of Measure	8-5
	Catalogue of Permitted Values	8-6
	Composition Hierarchy	8-6
9	Defining Characteristics	9-1
	Characteristic	9-1
	Fragment	9-1
	Composition	9-1
	Representation of Characteristics	9-2
	Rules for defining characteristics	9-4
10	Grouping Characteristics, Compound Fragments and Simple Fragments	10-1

	Purpose and Use	10-1
	General Description	10-1
	Array	10-1
	The Characteristic Group	10-2
11	Defining Characteristics	11-1
	Characteristic group	11-1
	Representation of Characteristics	11-1
	Rules for defining groups	11-3
12	Common Fragment Groups	12-1
	Purpose and Use	12-1
	General Description	12-1
	Common Reference Quantity	12-1
13	Defining Optionality Rules	13-1
	OR and XOR	13-1

ANNEXES

Annex A	Glossary	A-1
Annex B	Valid Referentials	B-1

FIGURES

Figure 1-1	CBML and Information System Development.....	1-1
Figure 3-1	Specialisation structure for a class	3-2
Figure 3-2	A hierarchy of specialisation structure for classes.....	3-3
Figure 4-1	Categorising set with a scope of a class.....	4-2
Figure 4-2	Categorising set with a scope of a category	4-3
Figure 4-3	Sub-dividing structure for a category	4-4
Figure 4-4	A hierarchy of sub-categorisation structure for categories	4-5
Figure 4-5	A member of a dividing set as the scope of a categorising set.....	4-6
Figure 5-1	A hierarchy of specialisation classification structure for a category.....	5-2
Figure 6-1	Categorising class scheme to a categorising set.....	6-1
Figure 6-2	Categorising class scheme to multiple categorising sets	6-2
Figure 6-3	Categorising entity class instances to a categorising set	6-3
Figure 7-1	Schematic of the structure of a characteristic	7-2
Figure 8-1	Schematic composition structure of a characteristic.....	8-1
Figure 8-2	Schematic unit of measure hierarchy.....	8-6
Figure 9-1	Standard representation of a characteristic	9-2
Figure 9-2	Representation of a characteristic with a single simple fragment.....	9-2
Figure 9-3	Standard representation of a characteristic that consists of multiple simple fragments....	9-3
Figure 9-4	A characteristic with multiple simple fragments and a compound fragment.....	9-3
Figure 10-1	Schematic for an array as a characteristic.....	10-1
Figure 10-2	Schematic for an array as a fragment.....	10-2
Figure 11-1	Representation of a non-predefined characteristic group.....	11-1
Figure 11-2	A predefined characteristic group with common reference quantity structure.....	11-2
Figure 11-3	Example of a group	11-2
Figure 13-1	Representation of a predefined OR or XOR characteristic group.....	13-1
Figure 13-2	Example of a more complicated predefined characteristic group of OR or XOR	13-2

TABLES

Table 6-1	Class Categorisation Usage Rules	6-4
Table 7-1	Characteristics Usage Rules.....	7-4
Table 8-1	Data Types	8-2
Table 8-2	Valid Native Combinations	8-4
Table 8-3	Referred Types.....	8-5
Table 9-1	Rules For Defining Characteristics.....	9-4
Table 11-1	Rules For Defining Groups.....	11-3

1 WHAT IS CBML?

DEFINITION

1. CBML is an Information Description Language. It is a set of concepts for the specification of the various types of information by which real-world objects are described. It allows this to be done with great precision in a manner that is entirely free of influence from any present or anticipated technology.

IMPACT OF DEFINING INFORMATION IN CBML

2. The differences between defining information and the modelling of data structures are:
 - a. Information structures define the information needed to perform the required functionality of a prescribed business.
 - b. Data structures define how data needs to be organised to provide the data needed to enable functionality; usually within a prescribed information system.

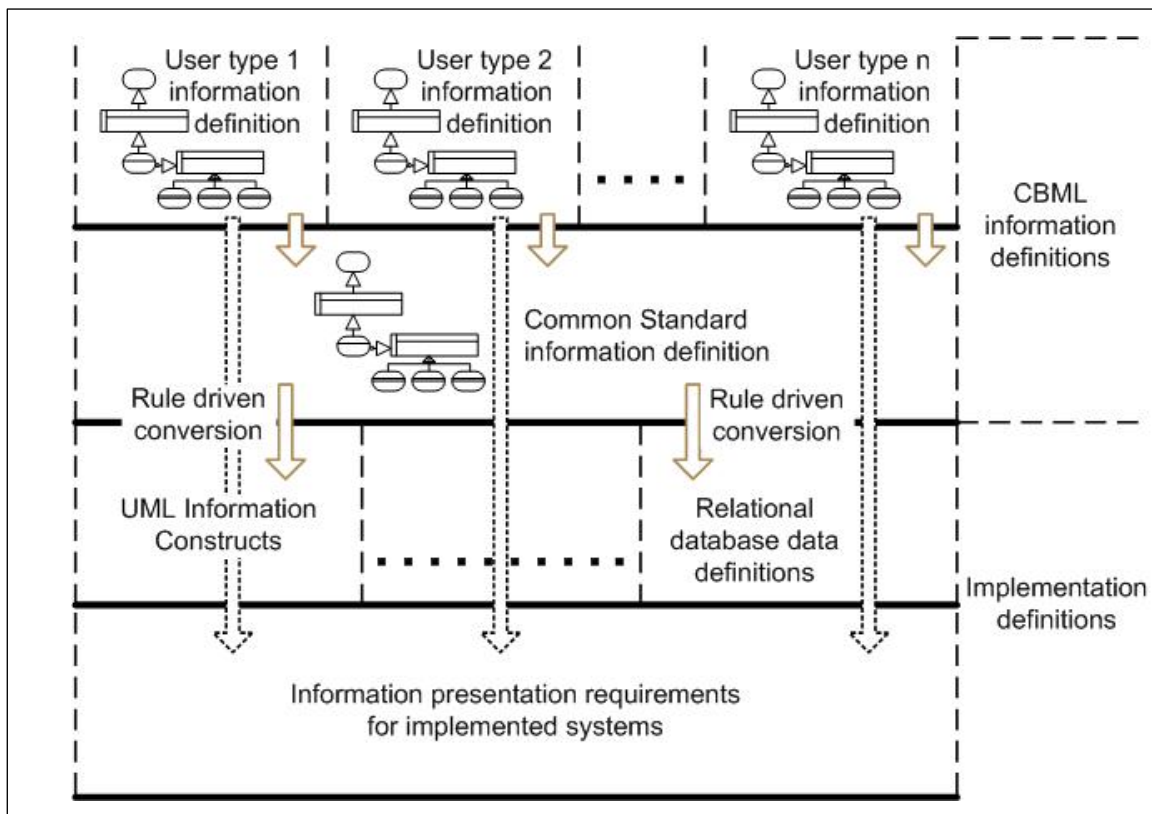


Figure 1-1 CBML and Information System Development

3. In complex businesses a number of systems are required to meet the needs of the business domains. By modelling information rather than data used in information systems the reconciliation of these diverse perceptions becomes possible.
4. The set of CBML concepts are sufficiently comprehensive to ensure that all possible information structures may be defined; and sufficiently restricted to ensure that there is one and only one way to correctly model a set of information. This presumes that the analysis of the information to be defined is consistent.

DIFFERENT PERCEPTIONS OF INFORMATION

5. The need to reconcile information leads to a requirement for an enterprise wide common understanding of information structures that is capable of representing the whole scope of the business. This is achieved by modelling information with the full rigour of the syntax rules of CBML.
6. Within different business domains there will be diverse perceptions of information that will be reflected in the Information definition for that business domain. These perceptions will be influenced by how information is used and, possibly, by the data structures employed in information systems used by the business domain.

USING CBML FOR RECONCILIATION

7. The underlying principles are:
 - a. Reconcile the business domain Information definition to the enterprise wide information structure.
 - b. Reconcile the business domain Information definition to the information system data structure.
8. This will enable the reconciliation of all Information definitions.

CONVENTIONS USED WITHIN THIS DOCUMENT

9. Within this document all CBML terms are shown in **bold type**. They are intended to be read and understood in accordance with their formal definition as given in the Glossary presented in Annex A.
10. All the diagrams in this document use an approved CBML graphical notation.

2 THE LANGUAGE STRUCTURE OF CBML

GENERAL CONCEPTS

1. CBML is a language and methodology for defining the information used by the business. Information is defined as “An expression of knowledge about a subject”.
2. A definition of information expressed in CBML is focused on the various types of **real-world entity** and **real-world substance** that the information is used to describe. **Real-world** is used to describe an entity or substance as “any identifiable thing, physical or otherwise, that may be encountered in the world”.
3. A **real-world entity** and a **real-world substance** represent something that exists in the real world. All other CBML concepts are concerned with the description of the information required to define these **real-world entities** and **real-world substances**.
4. Each **real-world entity** or **real-world substance** is defined separately from all other **real-world entities** and **real-world substances**.
5. Any association between **real-world entities** and/or **real-world substances** is defined from the perspective of the **real-world entity** or **real-world substance** being defined.

DIFFERENTIATION OF REAL-WORLD ENTITIES AND REAL-WORLD SUBSTANCES

6. **Real-world entities** and **real-world substances** are distinct and mutually exclusive concepts.
7. A **real-world entity** is something that exists in the **real-world** as an individual object, either tangible or intangible, such as “person”, “vehicle”, “invoice” etc.
8. A **real-world substance** is something that is tangible and exists in the **real-world** but never as an individual object for example “petrol”, “water”, “carbon dioxide” etc.
9. This difference between **real-world entities** and **real-world substances** causes differences in the way they are defined in CBML.

DEFINING REAL-WORLD ENTITIES AND REAL-WORLD SUBSTANCES

10. Each **real-world entity** is defined by a single **entity class** within CBML. There will be many instances of **real-world entities** defined by the same single specific **entity class**; for example there will be many individual persons as **real-world entities** defined by the single **entity class** “person”.
11. A specific **real-world substance** is defined by a specific **substance class** within CBML. There will be no **instances** defined of a **substance class**.
12. Generically both **entity class** and **substance class** are referred to by the term **class**.
13. Each **class** may be specialised through a **class specialising scheme**; **entity classes** by **entity class specialising schemes** and **substance classes** by **substance class specialising schemes**.

INFORMATION ABOUT ENTITY CLASSES AND SUBSTANCE CLASSES

14. The information about **entity classes** and **substance classes** is described in two ways:
 - a. **Categorisation**: this defines the different **categories** by which an **entity class** or **substance class** may be categorised. When such a **categorisation** is in place the **entity class** or **substance class** takes on the applicability of the **characteristics** of the relevant **category**.

- b. **Characteristics:** these define the **intrinsic** types of information relevant to an **entity class** or **substance class**.

CATEGORISATION INFORMATION FOR ENTITY CLASSES AND SUBSTANCE CLASSES

15. Everything that exists in the **real-world** is categorised in some way and usually in a number of different ways.
- A “person” may be categorised by “profession”, “nationality” etc.
 - A “vehicle” may be categorised by “usage type”, “transmission type” etc.
 - “Fuel” may be categorised by “viscosity”, “flash point”, “fuel type” etc.
 - The list of possible **categorisation** criteria is endless
16. **Entity categories** are used to categorise **entity classes**.
17. **Substance categories** are used to categorise **substance classes**.
18. **Categorisations** are used to classify **entity classes** and **substance classes** in as many different ways as required.
19. **Categorisations** define the rules to determine the **categorisation** of an **entity class**, an **instance** of an **entity class** and a **substance class**.

INTRINSIC INFORMATION FOR ENTITY CLASSES AND SUBSTANCE CLASSES

20. **Intrinsic** information about **entity classes**, **substance classes** and **categories** are defined as **characteristics** in CBML.
21. When an **entity class** or **substance class** satisfies the rules for **categorisation** the **class characteristics** defined for the relevant **category** are applicable to the relevant **entity class** or **substance class** and values will be required in accordance with the **usage rules**.
22. When an **instance** of an **entity class** satisfies the rules for **categorisation** the **instance characteristics** defined for the relevant **category** are applicable to this **instance** and values will be required in accordance with the **usage rules**.
23. An actual item of information that is a value for a **characteristic** applicable to an **entity class**, a **substance class** or an **instance** of an **entity class** as described by a **characteristic** is an **information element**.
24. Each **information element** is indivisible, self-contained and cannot be modified without creating a completely new version.

CONSTRAINTS ON SPECIALISATION STRUCTURE AND CATEGORISATION

25. Each **entity class** and each **substance class** has a single **specialisation structure** within the **specialisation structure**.
26. An **instance** of an **entity class** is classified to a single **entity class** and this classification cannot be changed during the life of that **instance**.
27. Each **entity class** or **substance class** may have none, one or many **categorisations**.
28. An **instance** of an **entity class** may be classified according to the **categorisations** of its **entity class** and these classifications may change during the life of that **instance** in accordance with the rules defined for each **categorisation**.

CBML ELEMENT DEFINITION

29. The basic **element types** in CBML are:
- a. Class with its two **sub-types**:
 - (1) **Entity class.**
 - (2) **Substance class.**
 - b. **Class specialising scheme** with its two **sub-types**:
 - (1) **Entity class specialising scheme.**
 - (2) **Substance class specialising scheme.**
 - c. **Category** with its two **sub-types**:
 - (1) **Entity category.**
 - (2) **Substance category.**
 - d. **Categorising set** with its two **sub-types**:
 - (1) **Entity categorising set.**
 - (2) **Substance categorising set.**
 - e. **Category dividing set** with its two **sub-types**:
 - (1) **Entity category dividing set.**
 - (2) **Substance category dividing set.**
 - f. **Categorisation.**
 - g. **Characteristic** with its two **sub-types**:
 - (1) **Simple characteristic.**
 - (2) **Complex characteristic.**
 - h. **Characteristic group.**
30. **Elements** have a set of properties relevant to the **element type** that are explained in the sections relevant to each **element type**.
31. **Elements** of the type: **class, specialising scheme, category, categorising set, characteristic** and **characteristic group** all have:
- a. An **element name** to provide purpose and meaning for the **element**.
 - b. An **element description** for additional explanatory information.
32. A **categorisation** does not have an **element name** or an **element description** as its purpose and meaning is always the same.
33. **Specialising scheme, categorising set** and **category dividing set** have an identification with the following structure: <name of owner/scope><the text string "by"><name of the scheme or set>

Intentionally Left Blank

3 CLASSES AND SPECIALISATION STRUCTURES

TERMINOLOGY

1. In the following:
 - a. **Class** is used to represent **entity class** and **substance class**.
 - b. **Specialising scheme** is used to represent **entity class specialising scheme** and **substance class specialising scheme**.

PURPOSE AND USE

2. The purpose of **classes** and **specialising schemes** is to specify a **class** about which information is to be defined and how the **class** fits into a hierarchy of other **classes** through **specialising schemes**.
3. It is not the purpose of **specialising schemes** to define the different ways in which a **class** may be classified¹. A **specialising scheme** is used to create **sub-classes** of a **class** and these **sub-classes** are themselves things in the **real world** rather than ways of classifying things in the **real world**.

GENERAL DESCRIPTION

4. Real-world **entity classes** and real-world **substance classes** are defined in a taxonomy that is the **specialising scheme**.
5. The **specialising scheme**:
 - a. Is a single hierarchy.
 - b. Has no networks only trees.
 - c. Has no cross-classification to another node in the taxonomy.
 - d. Has no multiple inheritance from multiple parents.
6. The position of a **real-world entity class** and **real-world substance class** in its **specialising scheme** is unequivocal and permanent.
7. The **sub-classes** inherit all the **characteristics** and **categorisations** of its owning **class**.
8. There is a different and separate **specialising scheme** for **entity classes** and for **substance classes** but some limited overlap is possible as described in Section 4.
9. The **specialising scheme** of **classes** may define **class characteristics** applicable to its **members**.
10. The **specialising scheme** of **entity classes** and the **specialising scheme** of **substance classes** are the same in basic structure.
11. The **specialisation structure** in CBML is represented by the **classes** and **specialising schemes**.
12. A **class** may be sub-classified by a **specialising scheme** when the **class** is defined as the **owner** of the **specialising scheme**.

¹ Classifying a **class** in different ways is the purpose of a **categorisation rule** explained in a later section.

13. The **members** of a **specialising scheme** are mutually exclusive delineated by the **class characteristics** defined in the **specialising scheme**.
14. The **specialising scheme** will have **members** that are **classes** and also **sub-classes** of the **owner** of the **specialising scheme**.
15. This creates a hierarchy with a **class** at the top of the hierarchy and this **class** is therefore not a **member** of a **specialising scheme**.
16. A **class** at the top of the hierarchy is a **root**.
17. Each **class** that is not a **root** is a **member** of one and only one **specialising scheme**.
18. Each **class** whether or not a **root**, may or may not be the **owner** of one and only one **specialising scheme**.
19. A **specialising scheme** is shown pictorially in Figure 3-1 using an approved CBML graphical notation.

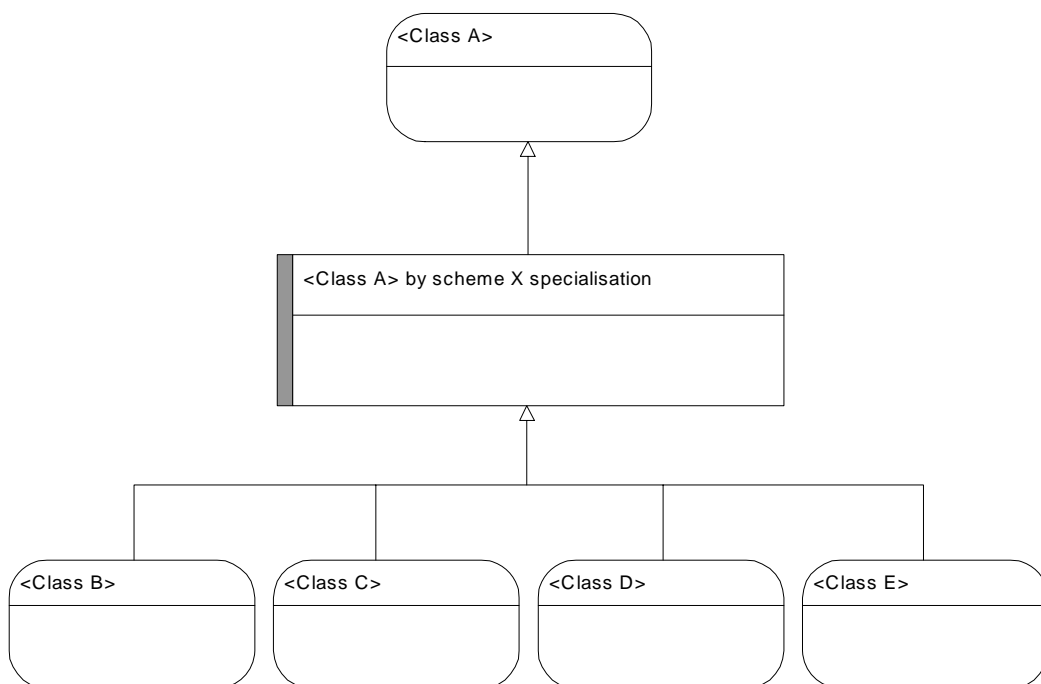


Figure 3-1 Specialisation structure for a class

20. The **class** <Class A> is sub-classified by the criteria defined by “scheme X specialisation” into **classes** <Class B>, <Class C>, <Class D> and <Class E>.
21. The **class** <Class A> is the **owner** of “scheme X specialisation” as indicated by the full title of the **specialising scheme** <Class A by scheme X specialisation>.
22. The **classes** <Class B>, <Class C>, <Class D> and <Class E> are **members** of <Class A by scheme X specialisation>.
23. The **classes** <Class B>, <Class C>, <Class D> and <Class E> are **sub-classes** of the **class** <Class A>.
24. The **classes** <Class B>, <Class C>, <Class D> and <Class E> are mutually exclusive.
25. The **class** <Class A> is not shown as being a **member** of a **specialising scheme** so within the **context** of Figure 3-1 the **class** <Class A> has the property of being a **root**.

26. A member of a **specialising scheme** may be the **owner** of one and only one **specialising scheme** creating a hierarchy as shown in Figure 3-2.

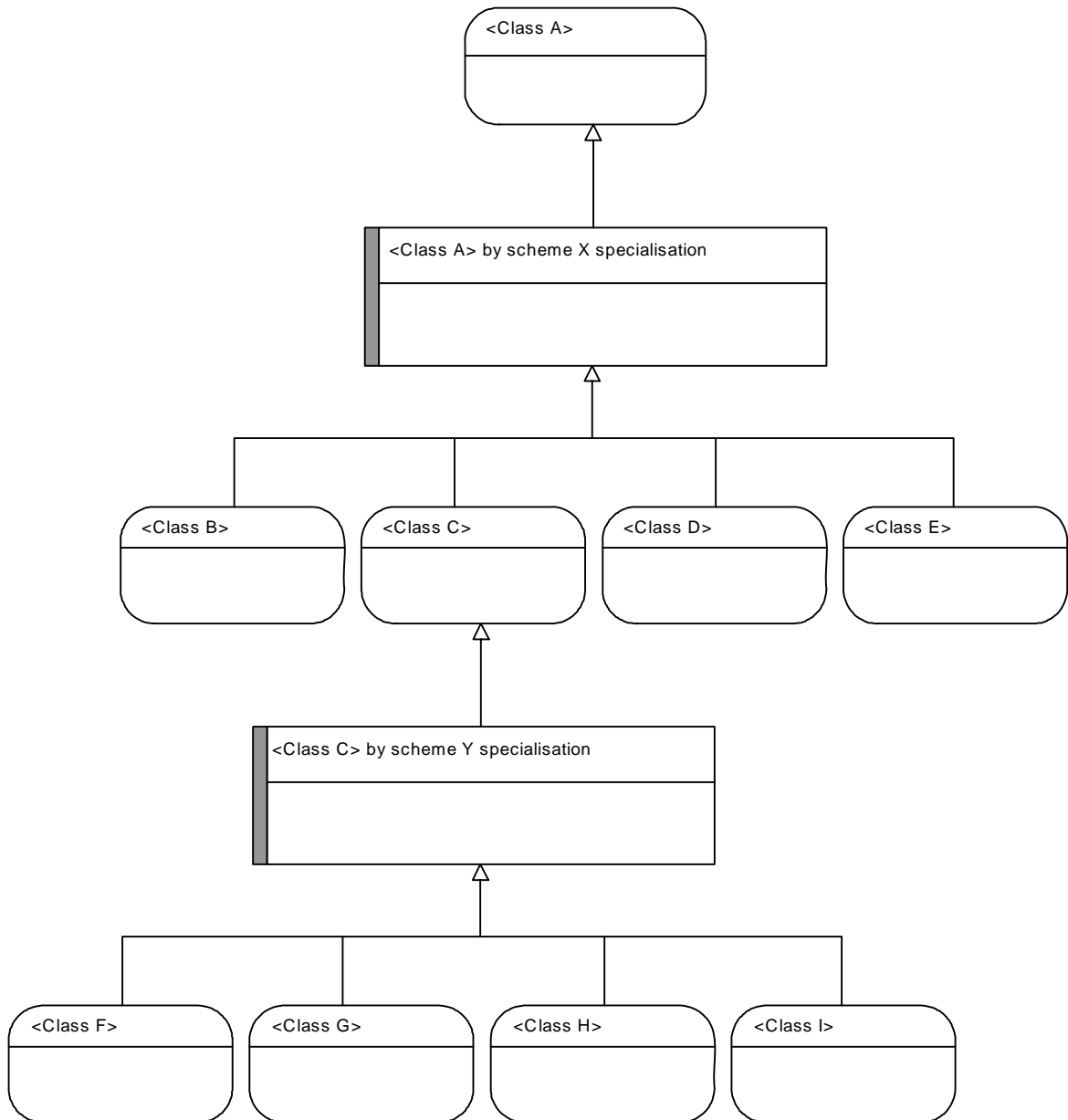


Figure 3-2 A hierarchy of specialisation structure for classes

27. At the top of this type of structure is the **class** <Class A> that has a **usage rule** of **root**.

28. No limit is set on the depth of a **specialisation structure**.

29. The **owner** of a **specialising scheme** may be a **category** as is described in Section 5.

CONSTRAINTS ON CLASSES

30. Each **class** has constraints on its use within a **CBML information definition**.

31. A **root class** asserts that the **class** is the top of a hierarchy of **classes** and **specialising schemes** and is not a **member** of a **specialising scheme**.

32. A **class** that is not a **root** asserts that the **class** is not the top of a hierarchy of **classes** and **specialising schemes** and is a **member** a of a **specialising scheme**.

4 CATEGORIES AND CATEGORISATION STRUCTURES

PURPOSE AND USE

1. The purpose of a **categorisation** is to define the different ways in which a **class** may be classified.
2. It is not the purpose of **categorisations** to define the **real-world specialisation structure** of a **class**.

GENERAL DESCRIPTION

3. Generally there is a different and separate **categorisation structure** for **entity classes** and for **substance classes**. The different **entity classes** and **substance classes** are classified by the **categorisation structure** for **entity classes** and **substance classes** respectively.
4. The following restrictions apply:
 - a. A **substance class** may not be categorised to an **entity category**.
 - b. An **entity class** may be categorised to a **substance category**.
5. The **categorisation** of **entity classes** may define both **class characteristics** and **instance characteristics** applicable to the relevant **entity class**.
6. The **categorisation** of **substance classes** may define only **class characteristics** applicable to the relevant **substance class** as there is no concept of an **instance** for a **substance class** and therefore no **instance characteristics** in the **substance class categorisation structure**.
7. The **categorisation structure** of **entity classes** and the **categorisation structure** of **substance classes** are the same in basic structure except for the existence or not of **instance characteristics**.
8. **Categorisation structure** in CBML is represented by the **categories**, **categorising sets** and **dividing sets**.
9. A **categorising set** is used to delineate a **class** or a **category**. The **class** or **category** being delineated is the **scope** of the **categorising set**.
10. A **class** or **category** may or may not be the **scope** of one **categorising set**.
11. A **class** or **category** may be the **scope** of one or more than one **categorising set**.
12. The **members** of a **categorising set** are **categorising categories**.
13. The **members** of a **categorising set** are mutually exclusive delineated by the criteria defined by the **categorising set**.
14. The **members** of a **categorising set** are delineated by the **class characteristics** defined in the **categorising set**.
15. A **dividing set** is used to define **sub-categories** of a **category**.
16. The **members** of a **dividing set** are **dividing categories**.
17. The **members** of a **dividing set** are delineated by the **class characteristics** defined in the **dividing set**.
18. Each **category** belongs to one and only one **categorising set** or **dividing set**.

19. A **category** belongs to a **categorising set** or a **dividing set** but never to both.

CATEGORISING SETS

20. When the **scope** of a **categorising set** is a **class** the **members** of the **categorising set** are used exclusively for the description of the scoping **class** including its **sub-classes**. To illustrate this a **categorising set** to categorise a **class** of “vehicle” will have **members** that are vehicles themselves; for example “front wheel drive vehicle”, “rear wheel drive vehicle”, “four wheel drive vehicle”.

21. A **class** that is the **scope** of a **categorising set** is the **apex** of the **categorisation structure** that contains the **categorising set** and all **categorising sets** and **dividing sets** within the same hierarchy.

22. A **class** as the **scope** of a **categorising set** is shown in Figure 4-1.

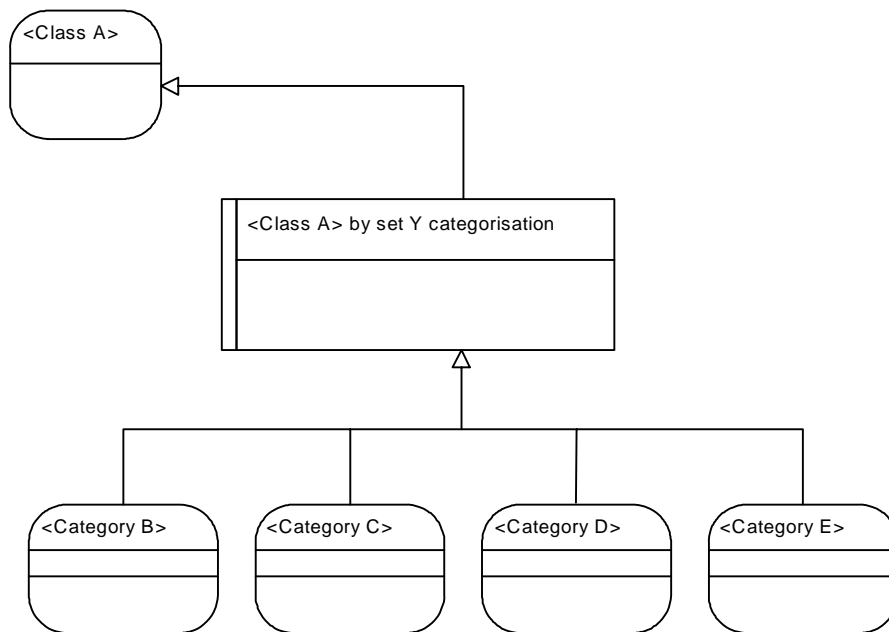


Figure 4-1 Categorising set with a scope of a class

23. In Figure 4-1 the **class** <Class A> defines the **scope** of the **categorising set** <Class A by set Y categorisation>; that is the **members** may be used to categorise <Class A> including any **sub-classes** of <Class A>. The **categorising set** <Class A by set Y categorisation> has four **categorising categories** - <Category B>, <Category C>, <Category D> and <Category E> - shown as **members**.

24. In Figure 4-1 the **class** <Class A> is the **apex** of the **categorisation structure**.

25. When the **scope** of a **categorising set** is a **category** the **members** of the **categorising set** are used exclusively for the description of the **class** that is the first **class** encountered when the hierarchy is followed up the **categorisation structure** until the **scope** of the **categorising set** is a **class**, the **apex** of the **categorisation structure**.

26. A **category** as the **scope** of a **categorising set** is shown in Figure 4-2.

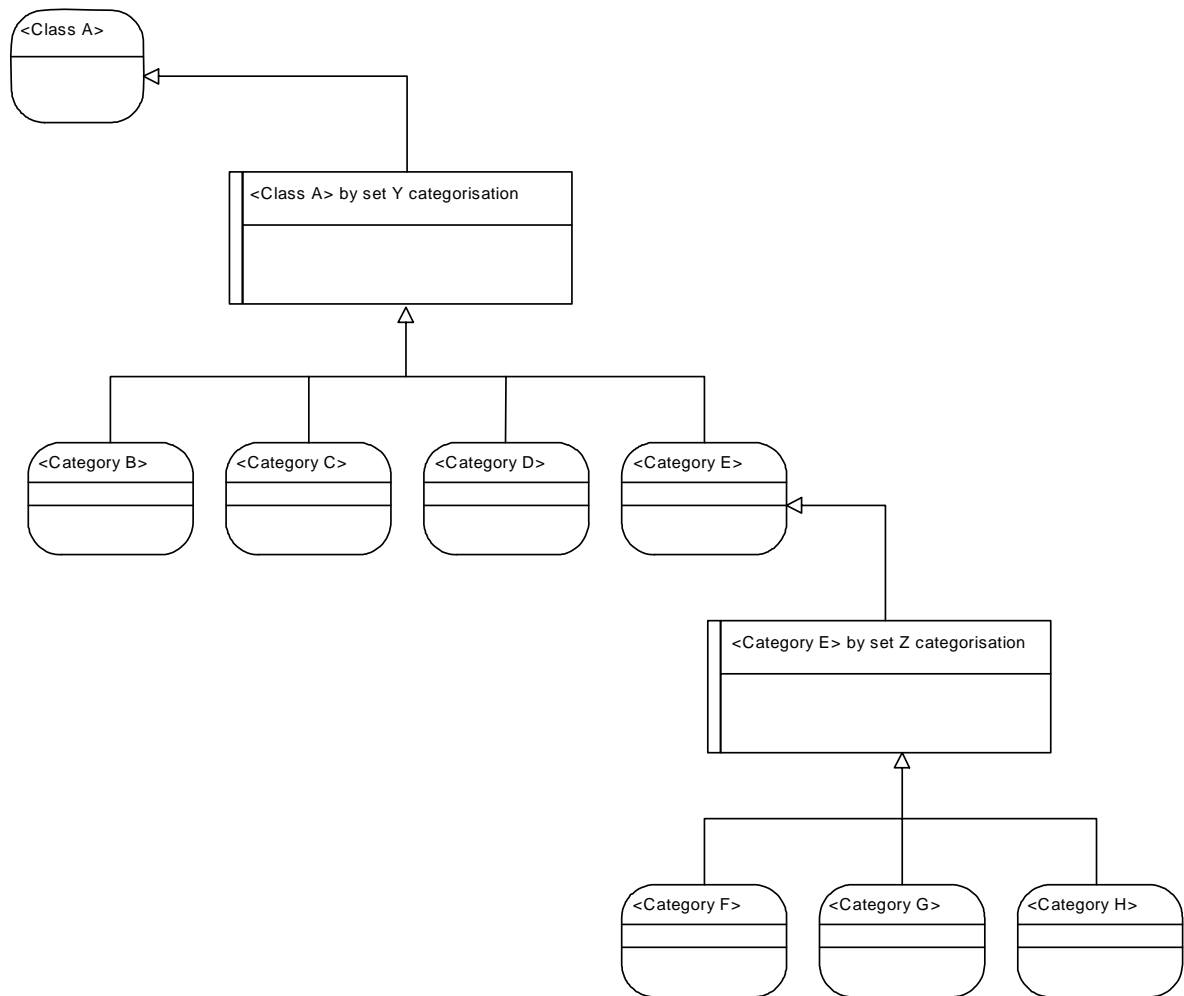


Figure 4-2 Categorising set with a scope of a category

27. In Figure 4-2 the top part of the structure is the same as Figure 4-1. The bottom half has the addition of a **categorising set** <Category E by set Z categorisation> that has a **scope** of **categorising category** <category E> and the three **categorising categories** - <Category F>, <Category G> and <Category H> - as **members** all of which define a way of categorising the **class** <Class A> including any **sub-classes** of <Class A>.

28. In Figure 4-2, as in Figure 4-1, the **class** <Class A> is the **apex** of the **categorisation structure**.

29. The top of a **categorisation structure** has a **scope** that is a **class**. All **categorising sets** that are lower in the structure will have a **scope** that is a **category**. No limit is set on the depth of a **categorisation structure**.

30. The **categorising categories** that are **members** of a **categorising set** that has a **category** as **scope** provide a finer degree of **categorisation** to that provided by the **category** that is the **scope** but the **members** do not inherit any **characteristics** or **categorisations** of the **category** that is the **scope**.

CATEGORY DIVISION

31. A **category** may be divided in a similar way to the specialisation of **classes**.

32. The **category** to be divided may be at any position in the **categorisation structure**.

33. A **dividing set** is used to divide a **category**.

34. A **category** to be divided is the **owner** of the **dividing set**.
35. All the **members** of a **dividing set** are **dividing categories**.
36. All the **members** of a **dividing set** are **sub-categories** of the **category** that is the **owner** of the **dividing set**.
37. The **sub-categories** inherit all the **characteristics** of the **owner** of the **dividing set**.
38. There is no restriction on a **category** being the **scope** of a **categorising set** and the **owner** of a **dividing set**.
39. A **category** may be the **owner** of one and only one **dividing set**.
40. The dividing of a **category** is shown in Figure 4-3.

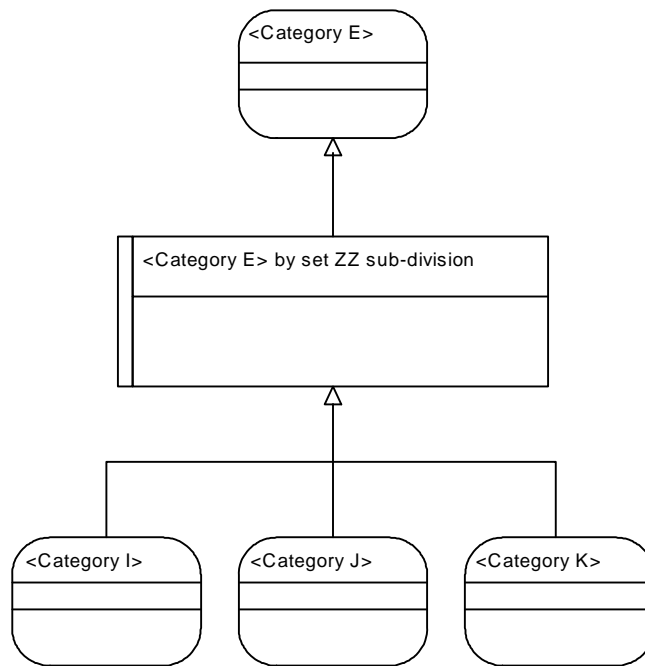


Figure 4-3 Sub-dividing structure for a category

41. In Figure 4-3 the **category** <category E> is sub-divided by the criteria defined by “set ZZ sub-division” into **dividing categories** <category I>, <category J> and <category K>.
42. The **category** <category E> is the **owner** of set “ZZ sub-division” as indicated by the full title of the **dividing set** <Category E by set ZZ sub-division>.
43. The **dividing categories** <Category I>, <Category J> and <Category K> are **members** of the **dividing set** <Category E by set ZZ sub-division>.
44. A **member** of a **dividing set** may be the **owner** of one and only one **dividing set** creating a hierarchy as shown in Figure 4-4.

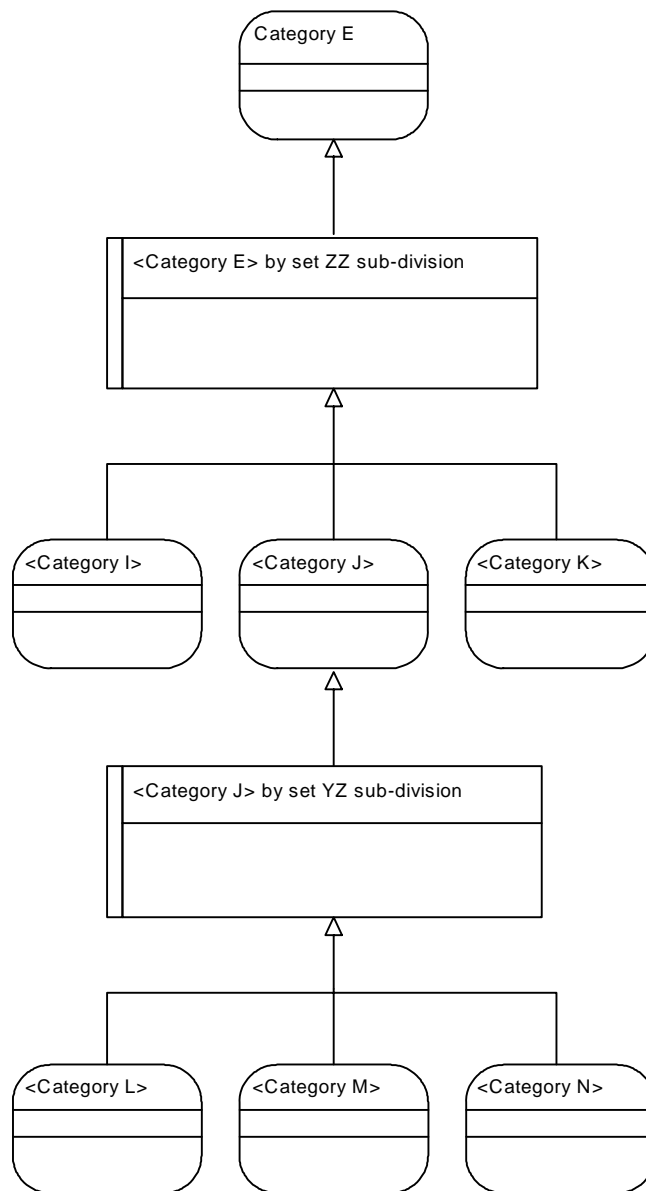


Figure 4-4 A hierarchy of sub-categorisation structure for categories

45. In Figure 4-4 the **dividing category** <category J> is sub-categorised by the criteria defined by “set YZ sub-division” into **dividing categories** <category L>, <category M> and <category N>.
46. The **dividing category** <category J> is the **owner** of **set** “YZ sub-division” as indicated by the full title of the **dividing set** <Category J by set YZ division>.
47. The **dividing categories** <Category L>, <Category M> and <Category N> are **members** of the **dividing set** <Category J by set YZ division>.
48. A **member** of a **dividing set** may be the **scope** of a **categorising set** as shown in Figure 4-5.

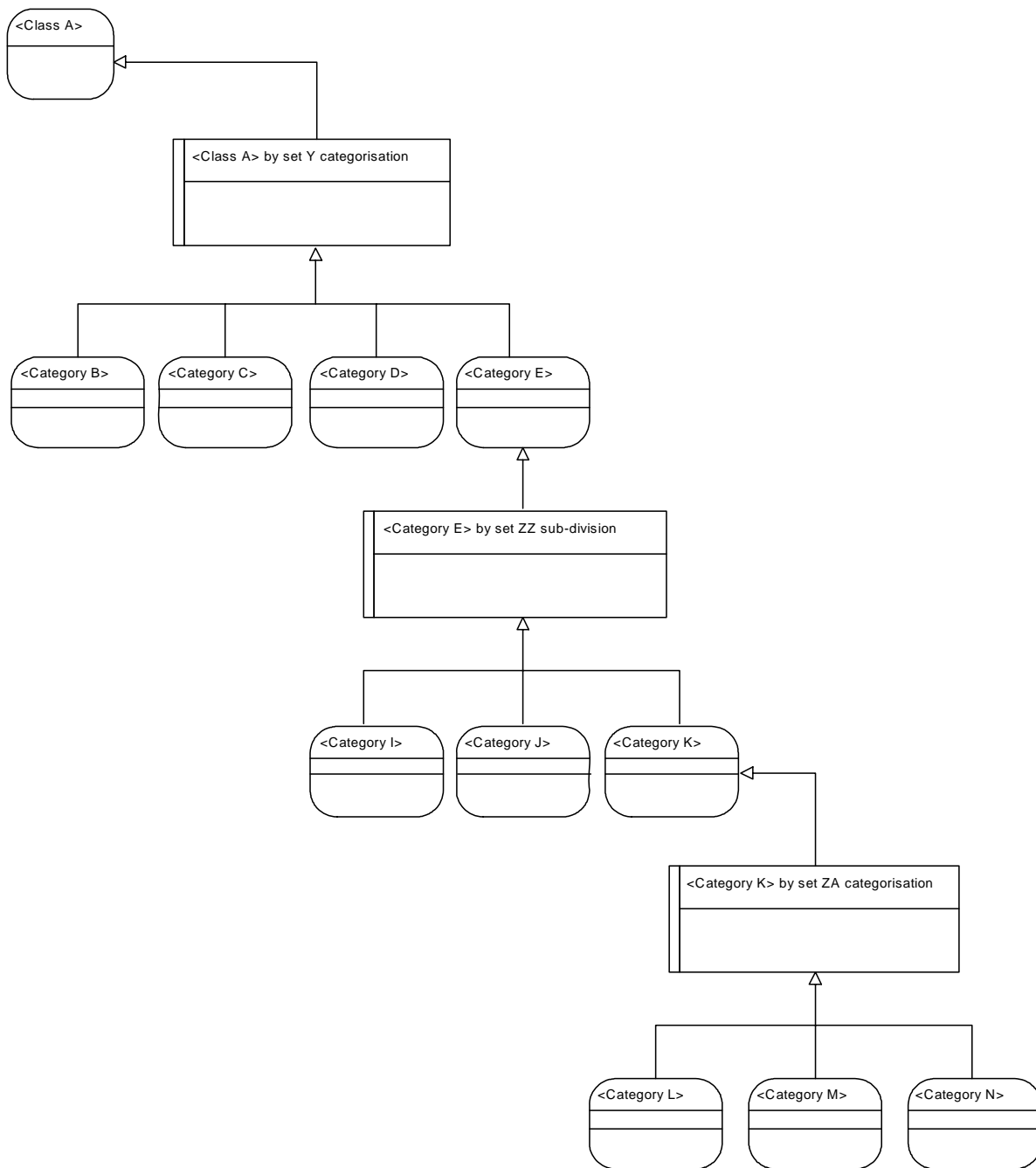


Figure 4-5 A member of a dividing set as the scope of a categorising set

49. In Figure 4-5 the **dividing category** <Category K> defines the **scope** of the **categorising set** <Category K by set ZA categorisation>; that is the **members** may be used to categorise <Class A> including any **sub-classes** of <Class A>. The **categorising set** <Category K by set ZA categorisation> has three **categorising categories** - <Category L>, <Category M> and <Category N> - shown as **members**.

5 SPECIALISING A CATEGORY

PURPOSE AND USE

1. The purpose of specialising a **category** is to define **sub-classes** of an **entity class** but the **sub-classes** are relevant only to those **instances** of the **entity class** that are categorised to the **category** that is the **owner** of the **specialising scheme**.
2. The **members** of the **specialising scheme** are **sub-classes** of the **entity class** that defines the **scope** at the top of the **categorisation structure** that is the **apex** of the **categorisation structure**.

GENERAL DESCRIPTION

3. Specialising a **category** in CBML is represented by the **category**, a **specialising scheme** and the **classes** that are **members** of the **specialising scheme**.
4. An **entity class** may be sub-classified by a **specialising scheme** that has a **category** as an **owner**.
5. A **specialising scheme** with an **entity category** as the **owner** is shown in Figure 5-1 using an approved CBML graphical notation.

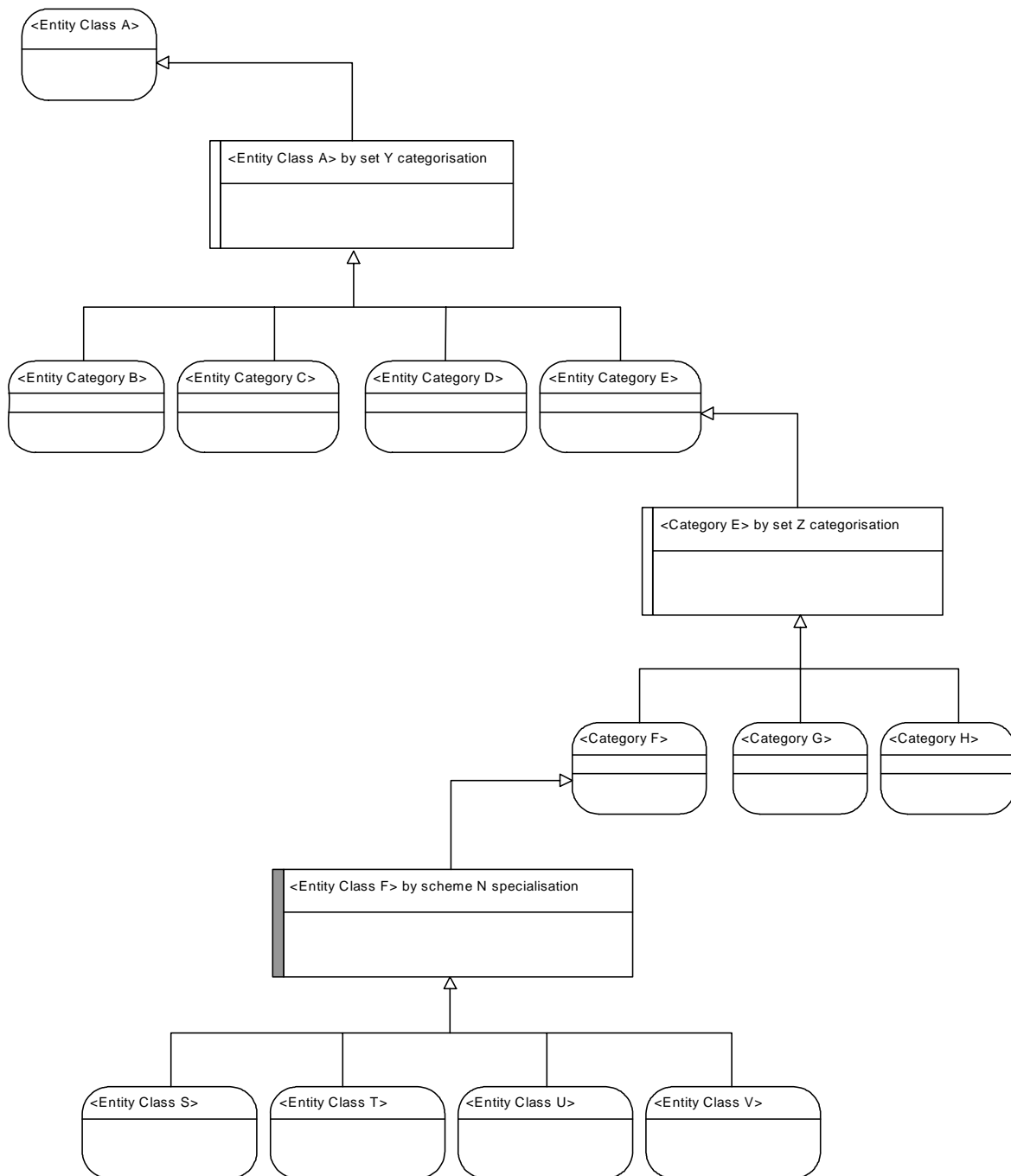


Figure 5-1 A hierarchy of specialisation classification structure for a category

6. In Figure 5-1 the **entity class** <Entity class A> is the **scope** for the **entity categorising set** <Entity class A by set Y categorisation>.
7. The **members** of the **entity categorising set** <Entity class A by set Y categorisation> are **entity categories** <Entity category B>, <Entity category C>, <Entity category D> and <Entity category E>.
8. The **entity category** <Entity category E > is the **scope** for the **entity categorising set** <Entity category E by set Z categorisation>.
9. The **members** of the **entity categorising set** <Entity category E by set Z categorisation> are **entity categories** <Entity category F>, <Entity category G> and <Entity category H>.

10. The **entity category** <Entity category F> is the **owner** of the **entity class specialising scheme** <Entity category F by scheme N specialisation>.

11. The **entity classes** <Entity class S>, <Entity class T>, <Entity class U> and <Entity class V> are **members** of the **entity class specialising scheme** <Entity category F by scheme N specialisation>.

12. The **entity classes** <Entity class S>, <Entity class T>, <Entity class U> and <Entity class V> are sub-types of the **entity class** <Entity class A> and inherit the **characteristics** and **categorisations** of **entity class** <Entity class A>.

13. The **entity classes** <Entity class S>, <Entity class T>, <Entity class U> and <Entity class V> are mutually exclusive.

14. The **category** that is the **owner** of a **specialising scheme** may be at any point in a **categorisation structure**.

Intentionally Left Blank

6 CATEGORISING OF CLASSES

PURPOSE AND USE

1. The purpose of the categorisation of **classes** is to classify **entity classes** and **substance classes** in all the ways necessary to provide all the required information about the **class**. This classification is distinct from any specialisation that is used in the specification of **classes** about which information is to be defined.
2. **Categorisations** may be applicable to a **class** or **instance** of a **class**.
3. When a **class** is categorised the **class** takes on all the features of the **category** it is categorised as. For example CBML could define a **class specialising scheme** of <vehicle> to be categorised to a **categorising set** of <vehicle by fuel type> that has as **members** “diesel powered vehicle” and “petrol powered vehicle”; indicating that a **class** of vehicle may be categorised as a “diesel powered vehicle” or a “petrol powered vehicle”. A **class** would not be categorised as “diesel powered” but as a “diesel powered vehicle”.
4. When an **instance** is categorised the **instance** takes on all the features of the **category** it is categorised as, for example CBML could define a **class** of “vehicle” to be categorised to a **categorising set** of “<vehicle> by colour” that has as **members** “red vehicle” and “black vehicle”; indicating that an **instance** of the **class** “vehicle” is either a “red vehicle” or a “black vehicle”. An **instance** would not be categorised as “red” but as a “red vehicle”.

GENERAL DESCRIPTION

5. **Categorisation** of **classes** is defined by categorising a class scheme to a **categorising set**.
6. The effect of categorising a **scheme** is that the **classes** that are **members** of the **scheme** will be categorised to a **category** that is a **member** of the **categorising set**.
7. **Categorisation** of **instances** of a **class** is defined by categorising a **class** to a **categorising set**.
8. The effect of categorising a **class** is that the **instances** of the **class** will be categorised to a **member** of the **categorising set**.
9. The hierarchy of the **class** or **class specialising scheme** that is being categorised and the hierarchy of the **categorising set** must intersect in a common **class** that is the **apex** of the **categorisation structure**.
10. The categorising of a **class** is shown in Figure 6-1.

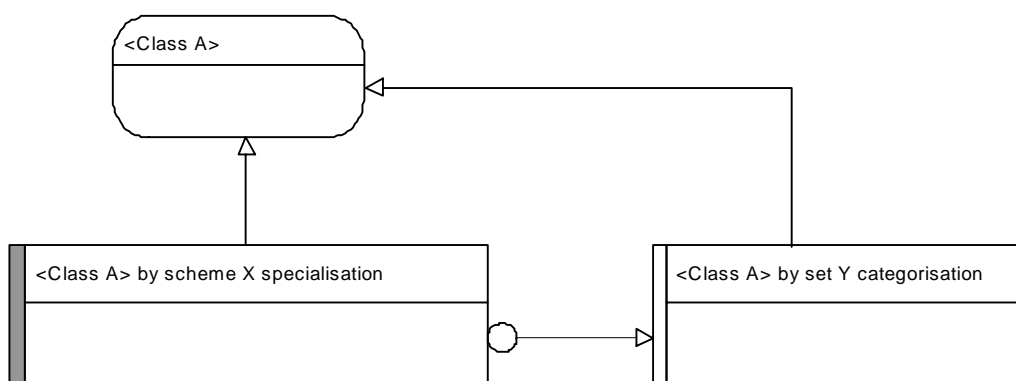


Figure 6-1 Categorising class scheme to a categorising set

11. Figure 6-1 shows a **specialising scheme** <Class A by scheme X specialisation> being categorised by the **categorising set** <Class A by set Y categorisation>.
12. This **categorisation** defines that each **member** of the **class specialising scheme** <Class A by scheme X specialisation> is categorised by a **member** of the **categorising set** <Class A by set Y categorisation>.
13. This **categorisation** is represented in Figure 6-1 by the arrow linking the **class scheme** <Class A by scheme X specialisation > to the **categorising set** <Class A by set Y categorisation>.
14. In Figure 6-1 the **class** <Class A> is in the hierarchy of the **class scheme** <Class A by scheme X specialisation > as the **owner** of the **scheme**, and in the hierarchy of the **categorising set** as the **scope** of the **categorising set** <Class A by set Y categorisation>.
15. A **class** and **instance** of a **class** may be categorised to a number of **categories** at the same time but all **categorisations** must be compatible, that is, within the hierarchies of all the **categorisations** and the hierarchy of the **class** or **class scheme** there is a common **class**. This is shown in Figure 6-2.

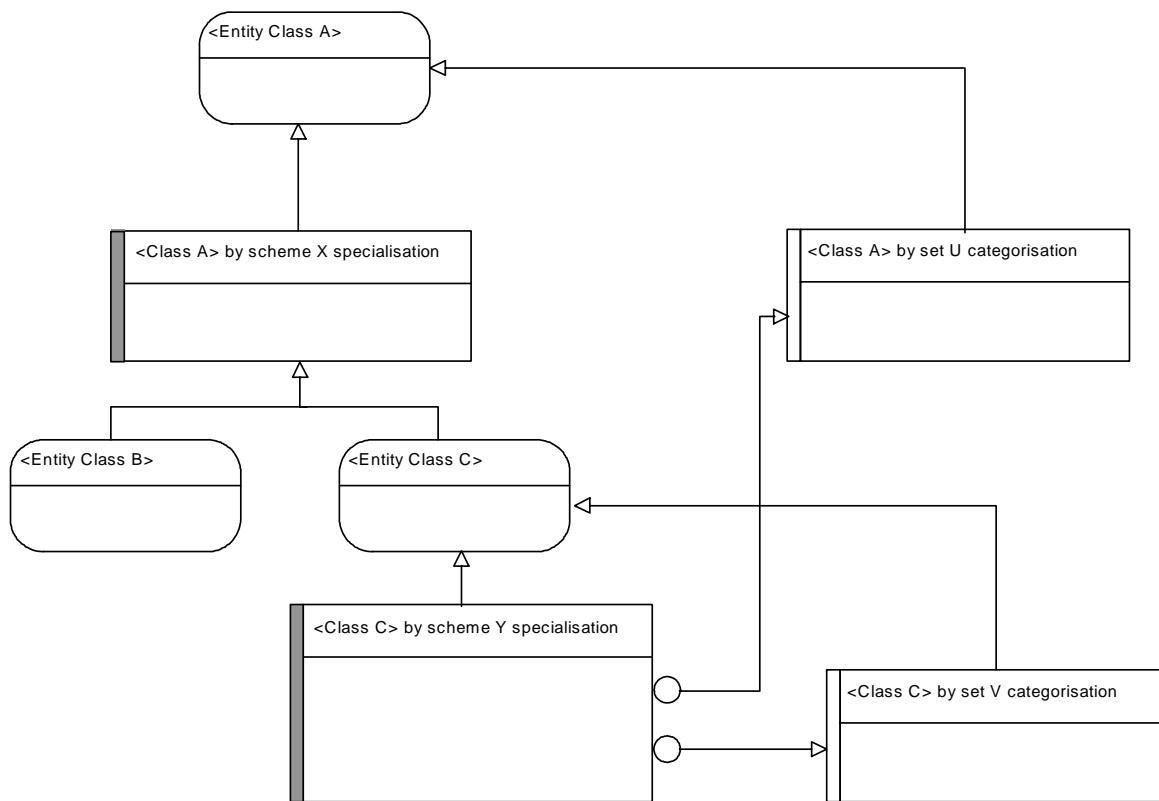


Figure 6-2 Categorising class scheme to multiple categorising sets

16. Figure 6-2 shows that each **member** of the **class scheme** <Class C by scheme Y specialisation> is categorised by a **member** of the **categorising set** <Class A by set U categorisation>, and a **member** of the **categorising set** <Class C by set V categorisation>.
17. The **categorisations** are compatible as:
 - a. The **apex** of the **categorisation structure** containing the **categorising set** <Class A by set U categorisation> is **class** <Entity class A>.
 - b. The **apex** of the **categorisation structure** containing the **categorising set** <Class C by set V categorisation> is **class** <Entity class C>.

c. The **classes** <Entity class C> and <Entity class A> share a common line of specialisation. In Figure 6-2 the **class** <Entity class C> is a sub-class of the class <Entity class A> as defined by the specialising scheme <Class A by scheme X specialisation>

18. As stated in the section “Categorisation rule in CBML” the **scope** of a **categorising set** may be defined by a **class** or a **category**.

19. The categorising of an **instance** is shown in Figure 6-3. Only **entity classes** can have the **categorisation** of an **instance** as **substance classes** do not have **instances**.

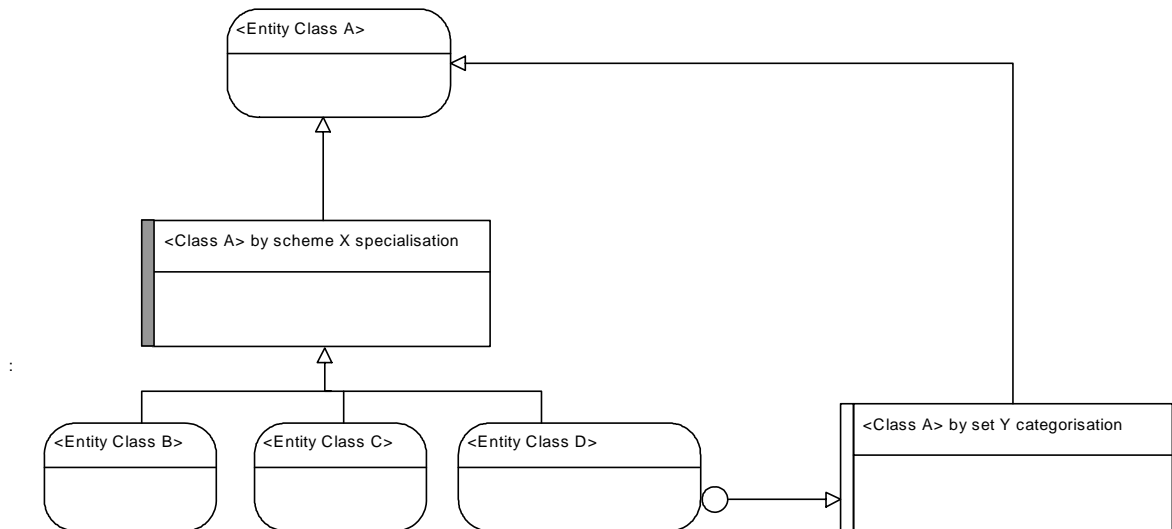


Figure 6-3 Categorising entity class instances to a categorising set

20. Figure 6-3 shows an **entity class** <Entity Class D> being categorised by the **categorising set** <Entity Class A by set Y categorisation>.

21. This **categorisation** defines that each **instance** of **entity class** <Entity Class D> is categorised by a **member** of the **categorising set** < Class A by set Y categorisation>.

22. This **categorisation** is represented in Figure 6-3 by the arrow linking the **entity class** <Entity Class D> to the **categorising set** < Class A by set Y categorisation>.

23. In Figure 6-3 the **entity class** <Entity Class A> is in the hierarchy of the **entity class** <Entity Class D> as the **owner** of the **class scheme** the **entity class** <Entity Class D> is a **member** of, and in the hierarchy of the **categorising set** as the **scope** of the **categorising set** < Class A by set Y categorisation>.

USAGE RULES IN THE CATEGORISATION OF CLASSES

24. Each **categorisation** of a **class** or **instance** has a number of rules associated with it that define constraints on its use. The **usage rules** are shown in Table 6-1.

Table 6-1 Class Categorisation Usage Rules

Usage Rule	Definition
<p>optional or mandatory</p>	<p>The categorisation of a class or instance must be defined as either optional or mandatory. Optional and mandatory have the following significance:</p> <p>optional asserts that the categorisation may or may not be defined for the class or instance;</p> <p>mandatory asserts that the categorisation must be defined for the class or instance.</p>
<p>single or multiple</p>	<p>The categorisation of a class or instance must be defined as either single or multiple. Single and multiple have the following significance:</p> <p>single asserts that the categorisation may be to one and only one member of the categorising set;</p> <p>multiple asserts that the categorisation may be to one or more members of the categorising set.</p>
<p>fixed or variable</p>	<p>The categorisation of a class or instance must be defined as either fixed or variable. Fixed and variable have the following significance:</p> <p>fixed asserts that once the member or members of the categorising set to which the class or instance is categorised have been defined then the categorisation may not be changed to another member, or other members, of the categorising set;</p> <p>variable asserts that once the member or members of the categorising set to which the class or instance is categorised have been defined then the categorisation may be changed to another member, or other members, of the categorising set.</p>
<p>time plurality</p>	<p>The categorisation of a class or instance may be defined with time plurality. Time plurality asserts that as the member or members of the categorising set to which the class or instance is categorised change with time to another member, or other members of the categorising set, then the different member or members of the categorising set may compete for relevance.</p> <p>When time plurality does not apply then there is no competition for the relevant members based on time.</p> <p>Time plurality can only apply with a usage rule of variable rather than fixed.</p>
<p>source plurality</p>	<p>The categorisation of a class or instance may be defined with source plurality. Source plurality asserts that the member or members of the categorising set to which a class or instance is categorised may vary dependent upon the source of the information that defined the member or members.</p> <p>When source plurality does not apply then there is no competition for the relevant members based on source.</p> <p>Source plurality can only apply with a usage rule of variable rather than fixed.</p>

25. The only interdependence of the **usage rules** are **source plurality** and **time plurality** with **fixed** or **variable**. There are no other dependencies between the **usage rules**.

Intentionally Left Blank

7 INTRINSIC INFORMATION IN CLASS OR CATEGORY

PURPOSE AND USE

1. The purpose of **characteristics** is to define the intrinsic types of information relevant to an **entity class** or **substance class**.

GENERAL DESCRIPTION

2. A **characteristic** describes a single piece of information known as an **information element** that has a value or set of values ascribed to it when used to describe an actual thing in the real world.
3. An **information element** is a single piece of information even though more than one value may be required to define that single piece of information.
4. The component parts of an **information element** cannot be treated independently.
5. Each **characteristic** describes either a **class** or an **instance** of a **class**¹.
 - a. If the **characteristic** describes a **class** it is a **class characteristic**.
 - b. If the **characteristic** describes an **instance** of a **class** it is an **instance characteristic**. Each **instance** of the **class** will have its own **information element**.
6. The applicability of each **characteristic** is either **intrinsic** or **contingent**.
 - a. If the **characteristic** is defined for a **class** it is an **intrinsic characteristic** and is always applicable to:
 - (1) The **class** for **class characteristics**.
 - (2) **Instances** of a **class** for **instance characteristics**.
 - b. If the **characteristic** is defined for a **category** it is a contingent **characteristic** and is applicable to:
 - (1) The **class** for **class characteristics** when the **class** is categorised to the **category** for which the **characteristic** is defined.
 - (2) An **instance** of the **class** when the **instance** is categorised to the **category** for which the **characteristic** is defined.
7. **Class characteristics** are defined in four ways:
 - a. Within a **specialising scheme** when the **class characteristics** apply to all the **members** of the **class scheme** for which the **class characteristic** was defined.
 - b. Within a **categorising set** or **dividing set** when the **class characteristics** apply to all the **members** of the **class scheme** that are categorised to one or more **members** of the **categorising set** for which the **class characteristic** was defined.
 - c. Within a specific **category** when the **class characteristics** apply to a **class** that is categorised to the **category** for which the **class characteristic** was defined.

¹ A **class** has a single **information element** for each **class characteristic** that holds a value that applies to the **class** itself rather than **instances** of the **class**; for example the design weight of a motor car of which there are many **instances**. Each **instance** of the **class** has an **information element** for each **instance characteristic** that holds a value that applies to an **instance** of the **class**; for example the registration number of an individual motor car.

- d. **Class characteristics** can be defined for a **root** entity as the **root** is at the top of the hierarchy with no owning **specialising scheme** to otherwise hold the **class**.
8. **Instance characteristics** are defined in two ways:
- a. Within an **entity class** when the **instance characteristics** apply to all the **instances** of the **entity class** for which the **instance characteristic** was defined. **Instance characteristics** may not be defined within a **substance class** as there is no concept of an **instance** for **substance classes**.
- b. Within an **entity category** when the **instance characteristics** apply to all the **instances** of the **entity class** that are categorised to the **category** for which the **instance characteristic** was defined. **Instance characteristics** may not be defined within a **substance category**, as there is no concept of an **instance** for **substance classes**.
9. Within CBML a **characteristic** is **unique** to the **class, scheme, category** or **set** to which it is attached. If similar **characteristics** are attached to different **classes, schemes, categories** or **categorising sets** then they are different **characteristics**.

STRUCTURE OF CHARACTERISTICS

10. A **characteristic** structure is a tree structure, the root of the tree is the **characteristic**, the leaves of the tree are **simple fragments** and the parts between the **characteristic** and the **simple fragments** are **compound fragments**.
11. The simplest **characteristic** consists of a single **simple fragment** but there is no limit to the complexity of a **characteristic**.
12. An **information element** that manifests a **characteristic** has an equivalent **information fragment** for each **fragment** of the equivalent **characteristic**.
13. A **fragment** is of a specific **composition**.
14. A **composition** is made up of three parts:
- a. A **data type** that is mandatory.
- b. An optional **composition**.
- c. An optional **composition constraint on value**.
15. A schematic structure of a **characteristic** is shown in Figure 7-1.

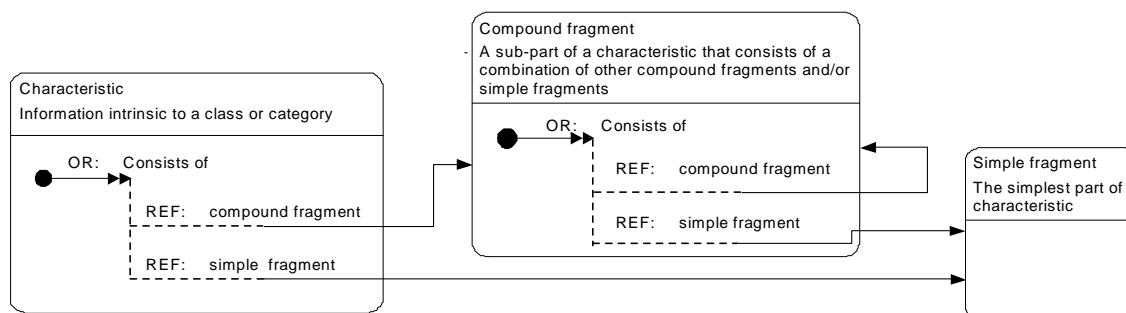


Figure 7-1 Schematic of the structure of a characteristic

16. **Characteristics** that are composed of a single **simple fragment** are termed **simple characteristics**.

17. A **characteristic** is of a particular **characteristic type**. There may be many **characteristics** of the same **characteristic type**.
18. The type of a **simple characteristic** is identified by the name of the **composition** of its single **simple fragment**.
19. **Characteristics** that are composed of more than one **fragment** are termed **complex characteristics**.
20. CBML has the following types of **complex characteristics**:
- a. **Quantity** which consists of the following **fragments**:
 - (1) An amount that has a **composition** of number that has composition rules applied to define the units of measure for the amount.
 - (2) A pointer to indicate a **class** that is the type of thing the amount refers to.
 - b. **Complex characteristic** where the user defines the **fragments** that make up the **characteristic**. The **fragments** may comprise of alternatives that may, or may not, be mutually exclusive .

USAGE RULES IN CHARACTERISTICS

21. Each **characteristic** of a **class** or **instance** has a number of rules associated with it. These are known as **usage rules** and define the constraints on its use within a **CBML information definition**.
22. The **usage rules** are shown in Table 7-1.

Table 7-1 Characteristics Usage Rules

Subject of usage rule	Usage rule explanation
optional or mandatory	<p>A characteristic must be defined as either optional or mandatory. Optional and mandatory have the following significance:</p> <p>optional asserts that the characteristic may or may not be defined for the class or instance;</p> <p>mandatory asserts that the characteristic must be defined for the class or instance.</p>
single or multiple	<p>A characteristic must be defined as either single or multiple. Single and multiple have the following significance:</p> <p>single asserts that the characteristic may have one and only one value for a simple characteristic or one set of values for a complex characteristic;</p> <p>multiple asserts that the characteristic may have more than one valid value for a simple characteristic or more than one valid set of values for a complex characteristic.</p>
fixed or variable	<p>A characteristic must be defined as either fixed or variable. Fixed and variable have the following significance:</p> <p>fixed asserts that the value for a simple characteristic or set of values for a complex characteristic once defined may not be changed;</p> <p>variable asserts that the value for a simple characteristic or set of values for a complex characteristic may be changed.</p>
time plurality	<p>A characteristic may be defined with time plurality. Time plurality asserts that as the value for a simple characteristic or set of values for a complex characteristic change with time these different values or sets of values may compete for relevance.</p> <p>When time plurality does not apply then there is no competition for the relevant values or sets of values based on time.</p> <p>Time plurality can only apply with a usage rule of variable rather than fixed.</p>
source plurality	<p>A characteristic may be defined with source plurality. Source plurality asserts that the value for a simple characteristic or set of values for a complex characteristic may vary dependent upon the source of the information that defined the value or set of values.</p> <p>When source plurality does not apply then there is no competition for the relevant values or sets of values based on source.</p> <p>Source plurality can only apply with a usage rule of variable rather than fixed.</p>

23. The only interdependence of the **usage rules** are **source plurality** and **time plurality** with **fixed** or **variable**. There are no other dependencies between the **usage rules**.

UNIQUENESS OF CHARACTERISTICS

24. A **characteristic**, or **characteristic group**, may be defined as **unique** within a uniqueness scope. The **uniqueness scope** may be another single **characteristic** or the **class, category, scheme** or **set** the **characteristic** belongs to.

25. A **characteristic** within a **characteristic group**, may be defined as **unique** within the **characteristic group**. The **uniqueness scope** is the **characteristic group**. This only applies when the **characteristic group** has a **repeating** set of values.

Intentionally Left Blank

8 COMPOSITION, DATA TYPES AND CONSTRAINTS

PURPOSE AND USE

1. **Composition** is used to describe the composition of a **simple fragment**.

INTERNAL STRUCTURE OF A CHARACTERISTIC

2. A **characteristic** is composed of one or more components known as **simple fragments**.
3. Each **simple fragment** is described by a **composition**.
4. A **composition** consists of a **data type**, an optional **composition qualifying detail** and optionally one or more **composition constraint on value**.
5. A schematic of the **composition** structure of a **characteristic** is shown in Figure 8-1.

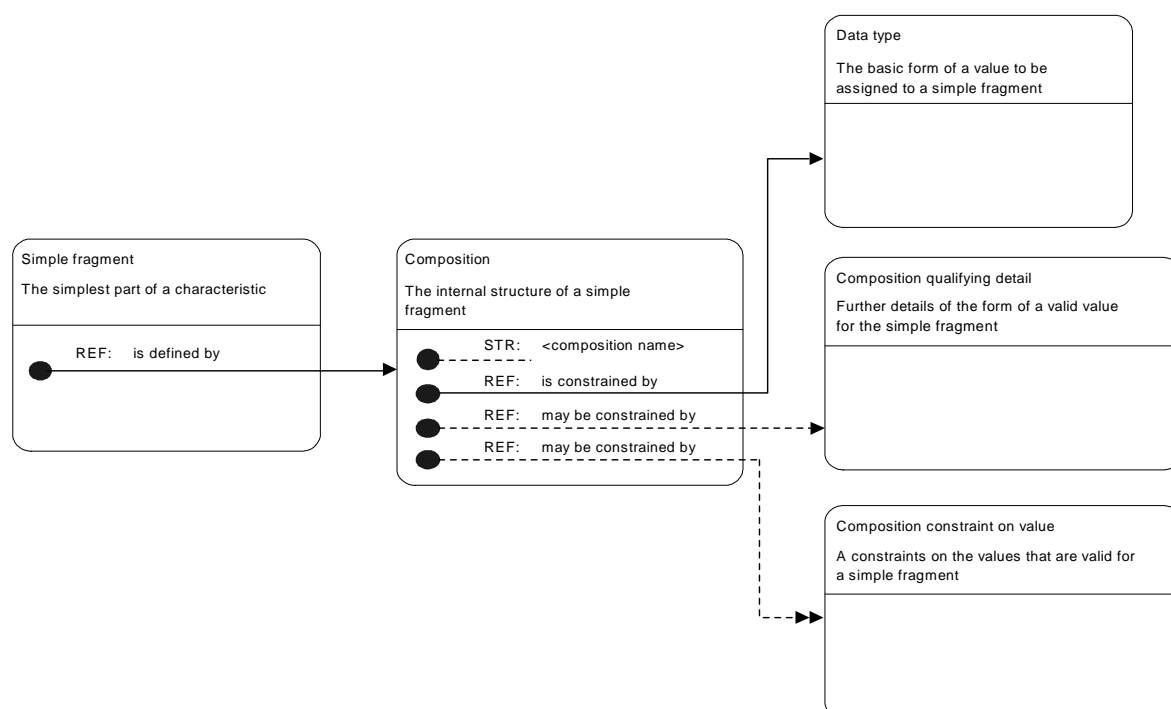


Figure 8-1 Schematic composition structure of a characteristic

6. A **composition** is solely a combination of **data type**, **composition qualifying detail**, and **composition constraint on value**; and of a name for identification.
7. There are five **data types** and these are defined in Table 8-1.

Table 8-1 Data Types

Data type	Fragment type description
character	A sequence of characters
numeric	A real number in the range of minus infinity to plus infinity including zero but excluding plus and minus infinity
boolean	A value of true or false only
time	A position or period in the continuum of time
referential	A pointer to one of the following: class, scheme, category or set

USING THE CHARACTER DATA TYPE

8. This defines a **fragment** that is a sequential set of characters of any type.
9. A **composition qualifying detail** could provide information regarding the character alphabet used or possibly the language if the **simple fragment** was text of some sort.
10. A **composition constraint on value** could provide a catalogue of permitted values.

USING THE NUMERIC DATA TYPE

11. This defines a **simple fragment** that is a real number that may have arithmetic performed upon it.
12. A **composition qualifying detail** could provide information regarding an applicable unit of measure.
13. A **composition constraint on value** could provide a minimum, a maximum or a range of values.

USING THE BOOLEAN DATA TYPE

14. This defines a **simple fragment** that has a value of “true” or “false” only.
15. The concepts of **composition qualifying detail** and **composition constraint on value** are not relevant and are not used.

USING THE TIME DATA TYPE

16. This defines a **simple fragment** that contains some form of **time**. This may be a date, a more precise time than date, a time that is repeated such as “1200 hours each day”, a period such as “1 hour, 10 minutes and 50 seconds”, “75 minutes” etc. **Time** has many representations normally as characters but it is a number that can have arithmetic performed upon it.
17. A **composition qualifying detail** could provide information regarding an applicable unit of time, the applicable calendar, a time datum, a time period of applicability¹, or other aspects of time.
18. A **composition constraint on value** could provide a minimum, a maximum, a range of values and a time limit.

¹ For example each day or week etc.

USING THE REFERENTIAL DATA TYPE

19. This **simple fragment** defines that a **class, scheme, category** or **set** refers to another **class, scheme, category** or **set**.
20. The **class, scheme, category** or **set** the **characteristic** containing the **simple fragment** is the “**native**” **class, scheme, category** or **set**.
21. The **class, scheme, category** or **set** the **simple fragment** refers to is the “**referred**” **class, scheme, category** or **set**.
22. When the **native** is a **class** the **characteristic** containing the **simple fragment** is an **instance characteristic**.
23. When the **native** is a **scheme** the **characteristic** containing the **simple fragment** is a **class characteristic**.
24. When the **native** is a **category** the **characteristic** containing the **simple fragment** may be an **instance** or a **class characteristic**.
25. When the **native** is a **set** the **characteristic** containing the **simple fragment** is a **class characteristic**.
26. Whether the **referred** is a **class, scheme, category** or **set** the thing referred to may be a **class** or an **instance**.

VALID CHARACTERISTIC TYPES

27. The valid **native** combinations of **native class, scheme, category** or **set**, and **class** or **instance characteristic** are shown in Table 8-2.

Table 8-2 Valid Native Combinations

Native type	Characteristic class or instance	Validity
All classes	class	Not valid
Entity class	instance	Valid
Substance class	instance	Not valid
Entity class specialising schemes	class and instance	Valid
Substance class specialising schemes	class	Valid
Substance class specialising schemes	instance	Not valid
Entity categories	class and instance	Valid
Substance categories	class	Valid
Substance categories	instance	Not valid
Entity class sets	class and instance	Valid
Substance class sets	class	Valid
Substance class sets	instance	Not valid

TYPES OF REFERRED CHARACTERISTICS

28. There are eight **referred** types and these are shown in Table 8-3.

Table 8-3 Referred Types

Referred type	Referred class or instance	Definition of the Referential Target
class	class	The referred class
Entity class	instance	An instance of the referred entity class
category	class	A class that is categorised to the referred category
Entity category	instance	An instance of an entity that is categorised to the referred category
scheme	class	A member of the referred scheme
Entity class specialising scheme	instance	An instance of a member of the referred scheme
set	class	A class that is categorised to the referred set
Entity set	instance	An instance of an entity class that is categorised to the referred set

29. If a **referential** is valid for the **native** then all the **referred** types are valid.
30. A complete list of possible referentials is given in Annex B.
31. The concepts of **composition qualifying detail** and **composition constraint on value** are not relevant and are not used.

UNITS OF MEASURE

32. The **units of measure** form a set of hierarchical structures.
33. Each hierarchical structure represents the **units of measure** for a particular **measurable property** such as: dimension, area, volume, speed, mass, etc.
34. The top of each hierarchical structure contains a **measurable property** that is simply a “unit to measure the defined measurable property”.
35. Each **unit of measure** may be the **owner** of a **unit of measure group** that defines a set of **unit of measure** for the defined **measurable property**.
36. A **unit of measure** may own more than one **unit of measure group** to separate for example imperial units of measure from metric units of measure.
37. A schematic **unit of measure** hierarchy is shown in Figure 8-2.

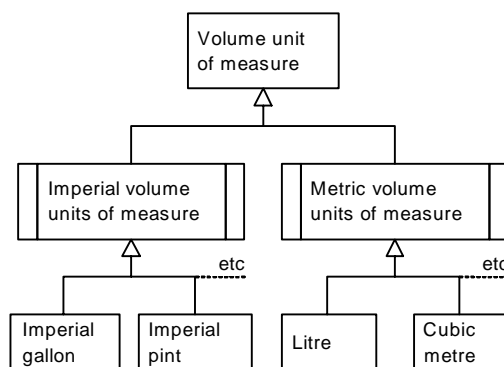


Figure 8-2 Schematic unit of measure hierarchy

PERMITTED VALUES

38. The permitted values takes the form of a list from which any value given to the **simple fragment** must be selected.

COMPOSITION HIERARCHY

39. A **composition** belongs to a hierarchy of **compositions**.

40. A **composition** that is not at the top of the hierarchy inherits the **data type**, the **composition qualifying detail** and all the **composition constraints on value** of its parent **composition**.

41. The child **composition** may further restrain the inherited **composition qualifying detail** by, for example, making the **unit of measure** more precise. This could be by changing the **unit of measure** from “imperial volume unit of measure” to “Imperial gallons”. The child **composition qualifying detail** must be contained within any limits defined for the parent **composition qualifying detail**.

42. The child **composition** may further restrain the inherited **composition constraint on value** by, for example, reducing the range of values or reducing the set of permitted values. The child **composition constraints on value** must be contained within any limits defined for the parent **composition constraints on value**.

43. CBML has defined a set of **compositions** indicating the parent and child relationships where appropriate.

44. **Compositions** may be user defined. All forms of user extensibility recognised by CBML involve the definition of **characteristics**.

45. The CBML defined **compositions** are:

Table 8-4 CBML Defined Compositions

Parent composition	Meaning	Child composition	Additional constraint
character	Only characters with an unrestrained character set	string	An unlimited number of characters in a specific sequence
		text	A body of natural language
numeric	A number in the range of minus infinity to plus infinity including zero but excluding plus and minus infinity	integer	A integer in the exclusive range of minus infinity to plus infinity including zero but excluding plus and minus infinity
time	A position in the continuum of time	Gregorian date	A day in the Gregorian calendar
		Gregorian time	A time on a day in the Gregorian calendar
		Duration	A time interval measured in years, months, weeks, days, hours, minutes, seconds and parts of a second
boolean	True or false setting	No child compositions	
referential	A pointer to a referred type.	No child compositions	

Intentionally Left Blank

9 DEFINING CHARACTERISTICS

CHARACTERISTIC

1. A **characteristic** is the definition of an **intrinsic** type of information.
2. When a **characteristic** is instantiated it represents a single piece of information.
3. A **characteristic** is made up of one or more **simple fragments** possibly through use of more **compound fragments**.

FRAGMENT

4. A **fragment** may be either:
 - a. A **simple fragment** that represents a single piece of data.
 - b. A **compound fragment** that represents more than one piece of data.
5. A **simple fragment**:
 - a. When instantiated it represents a single piece of data.
 - b. Must have a CBML **composition**.
6. A **compound fragment**:
 - a. A combination of two or more **simple fragments**.
 - b. Does not have a **composition**.

COMPOSITION

7. A **composition** consists of three parts:
 - a. **Data type.** This is the basic form of a value to be assigned to a **simple fragment**. There are five **data types** and these are shown below together with the shorthand notation used to represent them in diagrams.
 - (1) **Character** - STR
 - (2) **Numeric** - NUM
 - (3) **Boolean** - BOL
 - (4) **Time** - DTG
 - (5) **Referential** - REF
 - b. **Composition qualifying detail.** This provides further details of the form of a valid value for the **simple fragment** that could be, for example, the character alphabet or the language used for a **data type of character**, an applicable unit of measure for a **data type of numeric**, or an applicable unit time, the applicable calendar, a time datum, a time period of applicability for a **data type of time**.
 - c. **Composition constraint on value.** This is a constraint on the values that are valid for a **simple fragment** that could be, for example, a minimum, a maximum, a range of values, or a catalogue of permitted values.

8. The facility exists within CBML for derivatives to be created of the **data types** to reflect commonly used **composition qualifying details** and **composition constraints on value**.
9. The **boolean data type** does not have **Composition qualifying detail** or **composition constraint on value** and cannot have derivatives.

REPRESENTATION OF CHARACTERISTICS

10. The representation uses an approved CBML graphical notation.
11. Note that in the examples of graphical representation shown in Figure 9-1 to Figure 9-4:
- The choice of mandatory/optional, single/multiple, fixed/variable for the **usage rules** of the **characteristic** and the **fragment** are entirely arbitrary.
 - For **simple fragments** the REF **data type** has not been used; this is only to simplify the diagram and in all cases the REF **data type** would be valid.
12. A **characteristic** that consists of a single **simple fragment** would, to reflect the complete structure of **characteristics** and **fragments**, have the graphical representation shown in Figure 9-1.

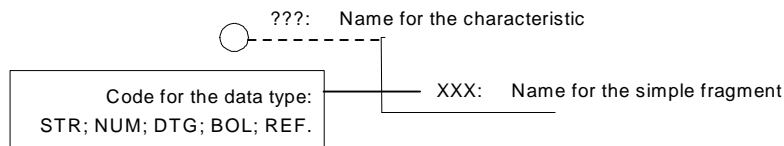


Figure 9-1 Standard representation of a characteristic

13. As the **characteristic** is fully described by the **simple fragment** then the “name of the characteristic” and the “name of the simple fragment” will be the same and, since there is only one **data type** applicable within the characteristic, the graphical representation shown in Figure 9-2 is used for a **characteristic** that consists of a **simple fragment**:

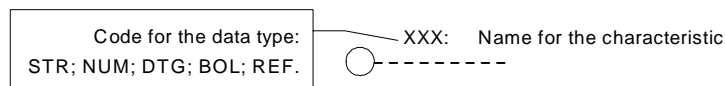


Figure 9-2 Representation of a characteristic with a single simple fragment

14. This is the only time that a **data type** can appear on a **characteristic** rather than a **fragment**.
15. **Characteristics** that consist of more than one **fragment** are termed complex.
16. Some **complex characteristics** have a predefined structure such as “Quantity”.
17. A **characteristic** that consists of multiple **simple fragments** has the graphical representation shown in Figure 9-3:

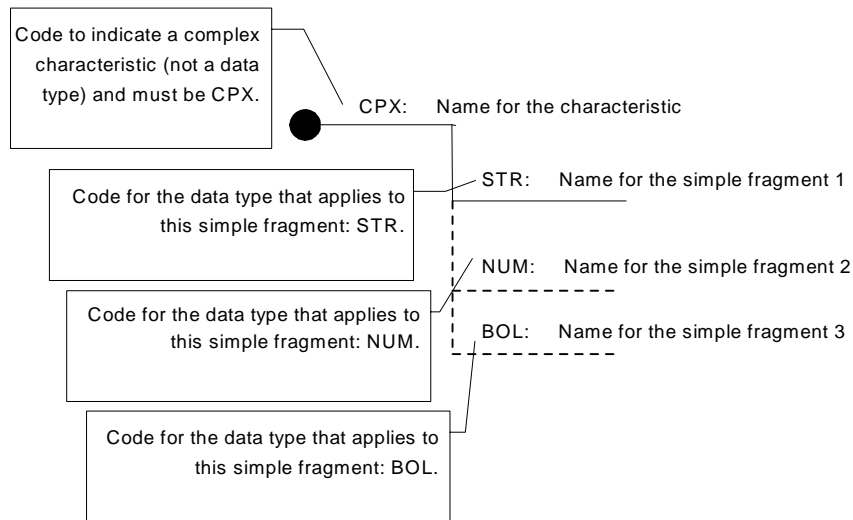


Figure 9-3 Standard representation of a characteristic that consists of multiple simple fragments

18. The CPX code can only be used to define a **characteristic**; never a **fragment**.
19. A **characteristic** that consists of multiple **simple fragments** and a **compound fragment** has the graphical representation shown in Figure 9-4:

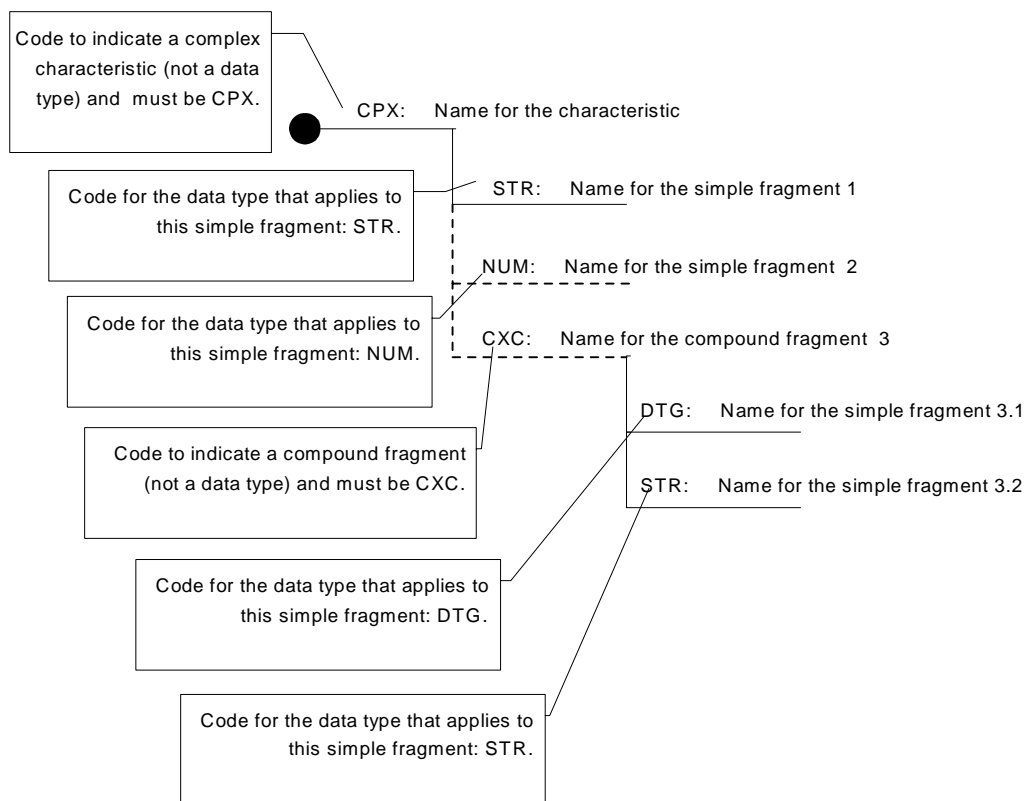


Figure 9-4 A characteristic with multiple simple fragments and a compound fragment

20. The CXC code can only be used to define a **compound fragment** never a **simple fragment**.
21. There is no limit to the depth of **complex characteristics** and **compound fragments** but the leaves of the tree are always **simple fragments**.

RULES FOR DEFINING CHARACTERISTICS

22. The rules for defining **characteristics** are shown in Table 9-1.

Table 9-1 Rules For Defining Characteristics

CBML characteristic element type	Code	Meaning of code	Constraint on use	Position in CBML
Characteristic	CPX	A characteristic consisting of more than a single fragment that is not a defined complex characteristic type	Cannot be used to define a fragment	Class, category scheme or set OR Characteristic group
Characteristic	QTY	A defined complex characteristic type	Cannot be used to define a fragment	Class, category scheme or set OR Characteristic group
Characteristic	STR NUM BOL DTG REF	Character Numeric Boolean Time Referential	Only when a characteristic consists of a single simple fragment	Class, category scheme or set OR Characteristic group
Fragment	CXC	A compound fragment	Only for fragments made up of more than one other fragments	Attached to a characteristic or other fragment
Fragment	STR NUM BOL DTG REF	Character Numeric Boolean Time Referential	Only simple fragments	Attached to a characteristic or other fragment

10 GROUPING CHARACTERISTICS, COMPOUND FRAGMENTS AND SIMPLE FRAGMENTS

PURPOSE AND USE

1. The purpose for repeating **characteristics** and **fragments** is to define how pieces of information are required to be managed collectively.
2. There are two types of grouping:
 - a. The group of repeating pieces of information need to be treated as a single **characteristic** or **fragment**.
 - b. The group of pieces of information that are separate **characteristics** but have some common behaviour.

GENERAL DESCRIPTION

3. The two types of grouping are managed by:
 - a. An **array**.
 - b. A **characteristic group**.

ARRAY

4. A **characteristic** that is an **array** consists of multiple repetitions of the same definition. The definition is the same as for other **characteristics** except for the concept of repeatability.
5. The multiple repetitions form a single **characteristic** and the equivalent **information element** will have a set of values to meet all the required repeats of the **array**.
6. As a **characteristic**, a **characteristic** that is an **array** has the same **usage rules** as other **characteristics**.
7. An **array** is shown schematically in Figure 10-1.

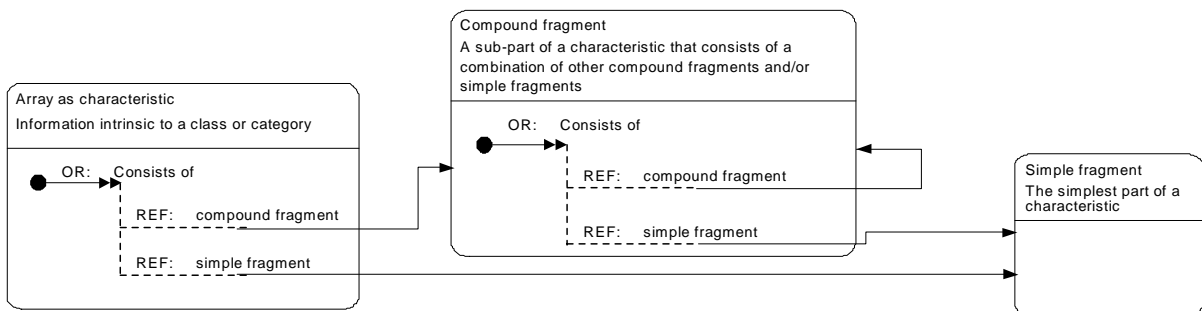


Figure 10-1 Schematic for an array as a characteristic

8. An **array** as a **characteristic** consists of a single set of unchanging **fragment** definitions.
9. An **array** as a **fragment** whose row is a set of **compound fragments** and/or **simple fragments** is shown schematically in Figure 10-2.

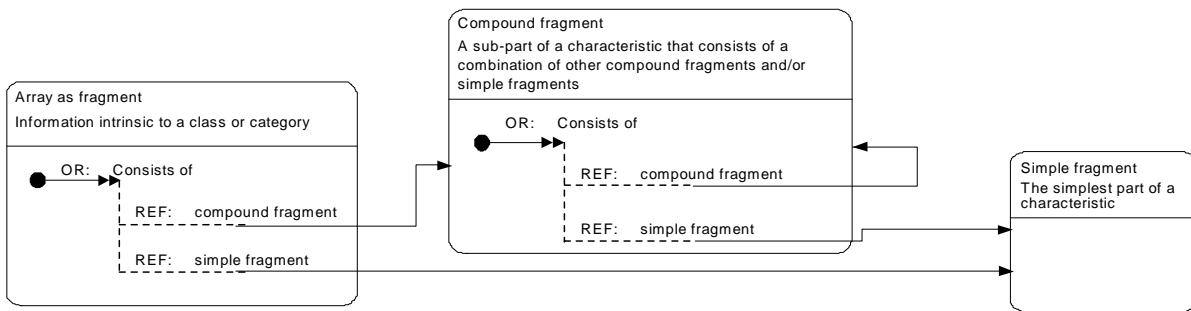


Figure 10-2 Schematic for an array as a fragment

10. An **array** whose row is a set of **compound fragments** and/or **simple fragments** consists of a single unchanging set of whose row is a set of **compound fragments** and/or **simple fragments**.
11. The “number of repeats” for the row may be defined if required.
12. A **fragment** that is an **array** consists of multiple repetitions of the same definition. A **fragment** as an **array** is applicable only to **complex characteristics**.
13. **Array** is different to **plurality** and the **array** may have **plurality** applied to it.

THE CHARACTERISTIC GROUP

14. A **characteristic group** is a collection of simple or **complex characteristics**.
15. A **characteristic group** shares the same **usage rules** as **characteristics** with the addition that the presence of each **characteristic** within the **characteristic group** is **optional** or **mandatory**.
16. When the **single** or **multiple usage rule** for a **characteristic group** is set to **multiple** the **characteristic group** becomes a **characteristic group**.

11 DEFINING CHARACTERISTICS

CHARACTERISTIC GROUP

1. A **characteristic group** is a group of **characteristics** that are treated collectively while each **characteristic** in the **characteristic group** remains a discrete **characteristic**.
2. Some **characteristic groups** have a predefined structure such as **common reference quantity**.

REPRESENTATION OF CHARACTERISTICS

3. The representation uses an approved CBML graphical notation.
4. Note that in the examples of graphical representation shown in Figure 11-1 to Figure 11-3:
 - a. The choice of mandatory/optional, single/multiple, fixed/variable for the usage rules of the **characteristic** and the **fragment** are entirely arbitrary.
 - b. For **simple fragments** the REF **data type** has not been used; this is only to simplify the diagram and in all cases the REF **data type** would be valid.
5. The components of a **characteristic group** may be of any type of **characteristic**.
6. A non-predefined **characteristic group** has a graphical representation similar to that shown in Figure 11-1 but the actual structure will be defined by the requirements.

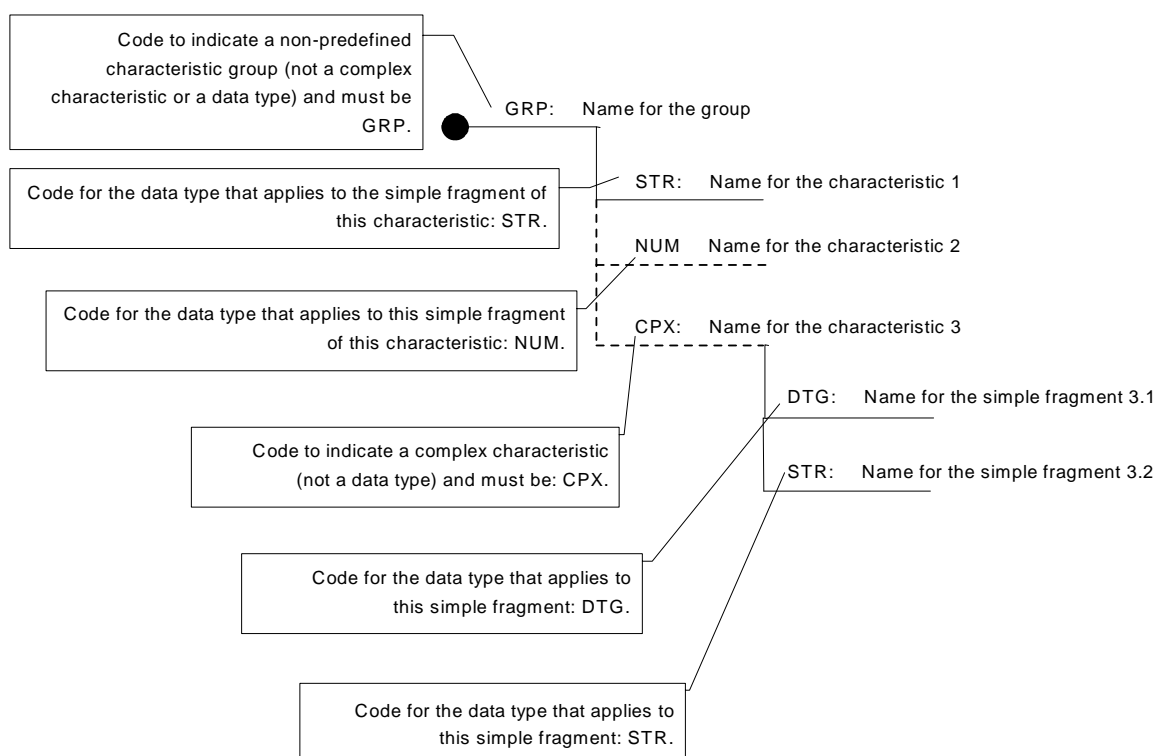


Figure 11-1 Representation of a non-predefined characteristic group

7. The **complex characteristic** CPX can be to any level of complexity. The one shown in Figure 11-1 is one with a second level of complexity.
8. A predefined **characteristic group** of type **common reference quantity** has the graphical representation shown in Figure 11-2.

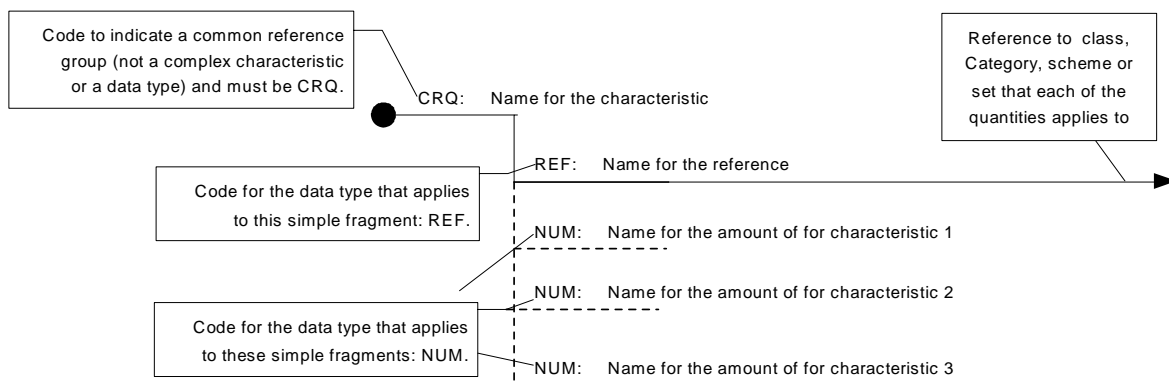


Figure 11-2 A predefined characteristic group with common reference quantity structure

9. In Figure 11-2:
 - a. The **NUM data type** for “characteristic 1” combines with the **REF data type** to form a defined **complex characteristic** of type QTY.
 - b. The **NUM data type** for “characteristic 2” combines with the **REF data type** to form a defined **complex characteristic** of type QTY.
 - c. The **NUM data type** for “characteristic 3” combines with the **REF data type** to form a defined **complex characteristic** of type QTY.
 - d. There must be at least two NUMs but there is no maximum limit and hence no maximum limit on the number of QTYs but all the quantities when instantiated must be about the same type of thing.
10. An example of a group is shown in Figure 11-3.

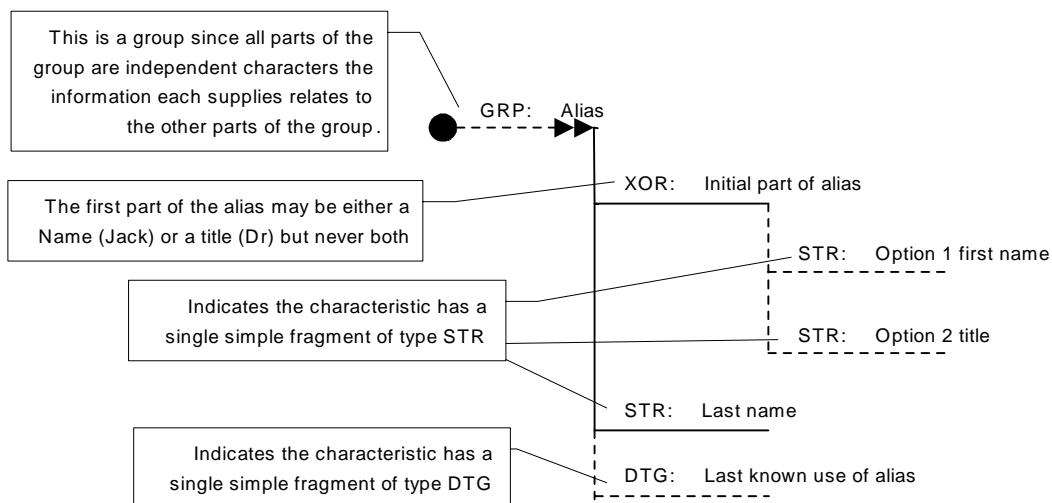


Figure 11-3 Example of a group

11. Figure 11-3 shows a **group** called “Alias” that consists of 3 parts:
 - a. The first part is an optional rule of **XOR**; that is it is either a first name or a title.
 - b. The second part is a last name.
 - c. The third part is an optional date on which the alias was last used.

RULES FOR DEFINING GROUPS

12. The rules for defining **groups** are shown in Table 11-1.

Table 11-1 Rules For Defining Groups

CBML group type	Code	Meaning of code	Constraint on use	Position in CBML
Generic	GRP	A group consisting of more than one characteristic	Cannot be used to define a fragment	Class, category scheme or set OR Characteristic group
Common reference quantity	CRQ	A set of quantities that share a common "of what"	Cannot be used to define a fragment	Class, category scheme or set OR Characteristic group

Intentionally Left Blank

12 COMMON FRAGMENT GROUPS

PURPOSE AND USE

1. **Common fragment groups** identify **complex characteristics** in the same **native element** that share **characteristic compound fragments** and/or **fragments**.

GENERAL DESCRIPTION

2. A **common fragment group** is not a CBML element but defines a common property of the **complex characteristics** within the **common fragment group**.

3. The behaviour of a **complex characteristic** that shares a **fragment** with another **complex characteristic** is not affected by the sharing.

4. CBML has defined the following **common fragment group** and more are expected to be defined with time:

a. **Common reference quantity**.

COMMON REFERENCE QUANTITY

5. These need to be defined when different **quantity complex characteristics** in the same **native element** refer to the same **class** or **category** that is the type of thing the quantity refers to.

6. **Common reference quantities** define a number of QTY characteristics that share a common 'of what'. These are characteristics not fragments and consequently may have plurality attached.

Intentionally Left Blank

13 DEFINING OPTIONALITY RULES

OR AND XOR

1. The optionality rules for **OR** and **XOR** are:
 - a. **OR** defines a set of CBML **characteristics** and/or **characteristic groups** and/or **fragments** that are alternatives that are not mutually exclusive.
 - b. **XOR** defines a set of CBML **characteristics** and/or **characteristic groups** and/or **fragments** that are mutually exclusive.
2. Optionality rules OR or XOR are identical in structure and have the graphical representation shown in Figure 13-1.

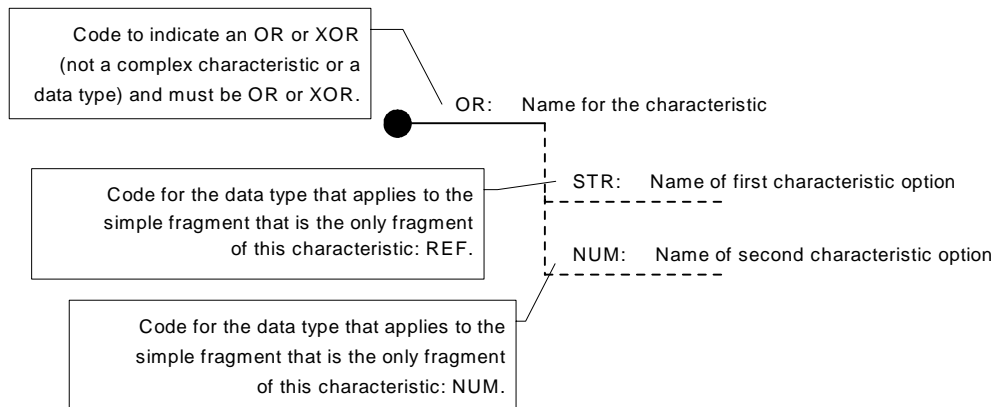


Figure 13-1 Representation of a predefined OR or XOR characteristic group

3. There is no limit on the number of options within an **OR** or **XOR** characteristic group.
4. The options may be:
 - a. **Characteristics** including **complex characteristics**.
 - b. An **OR** or **XOR**.
 - c. A **characteristic group**.
5. A more complicated optionality rule OR or XOR could have the following graphical representation:

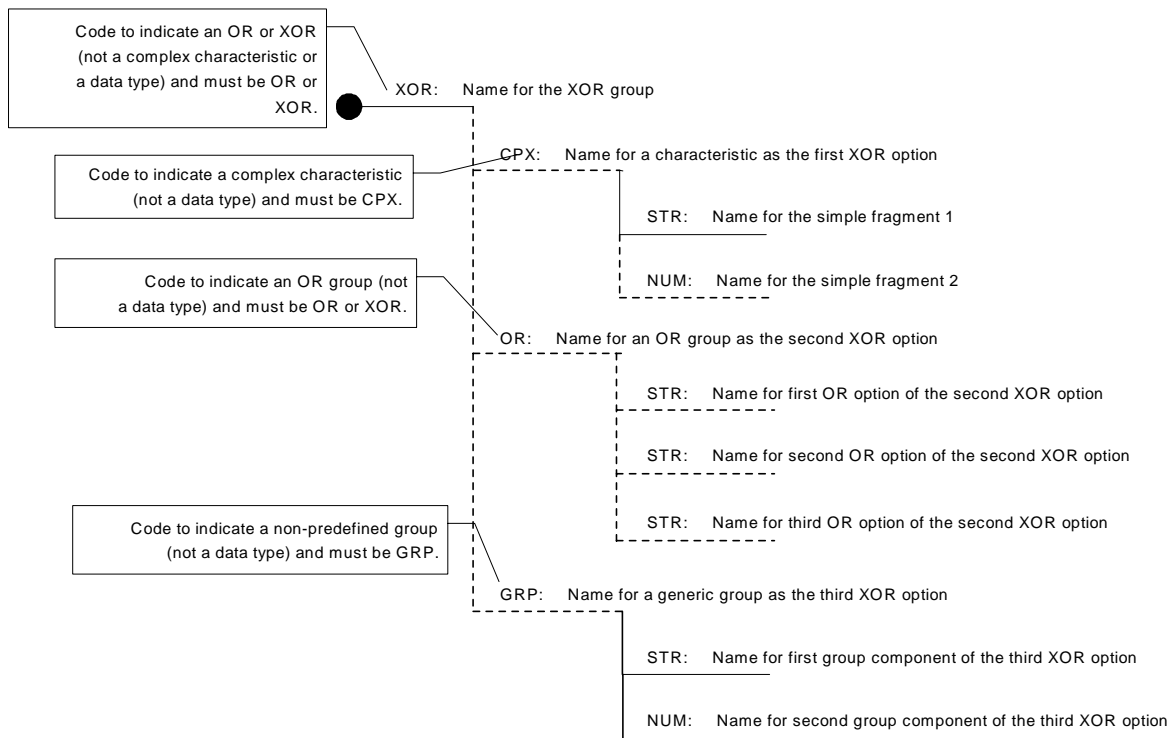


Figure 13-2 Example of a more complicated predefined characteristic group of OR or XOR

ANNEX A GLOSSARY

Actual User Perspective (AUP)	An information definition that represents precisely the perception of a user of information independent of wider aspects of information usage.
apex	The class , not a category , that is the scope of the categorising set that is at the top of a categorisation structure .
array	Multiple repetitions of the same characteristic or fragment definition treated as a single characteristic or fragment .
boolean	BOL – A data type that can have a value of true or false only.
categorisation	Defines how a class may be categorised. When a categorisation is in place the class or category takes on the characteristics of the relevant category .
categorisation structure	A hierarchy of categorising sets , dividing sets and categories . The top categorising set has a scope that is a class . This class is the apex .
categorising classification	A classification of an entity class , substance class , entity category or substance category that is maintained during the life of the entity class or substance class being categorised in accordance with rules defined in a CBML information definition .
categorising set	An inclusive term for entity categorising set and substance categorising set .
category	An inclusive term for entity category and substance category .
category descriptor	The definition of a type of information that is used to describe a category within a specific categorising set .
category dividing set	An inclusive term for entity category dividing set and substance category dividing set .
CBML information definitions	A definition of information defined in CBML.
character	STR – A data type whose value is a sequence of characters.
characteristic	The definition of an intrinsic type of information.
characteristic group	A group of characteristics that are treated collectively while each characteristic in the characteristic group remains a discrete characteristic .
characteristic type	A characteristic is either a class characteristic , that is it applies to the class , or an instance characteristic , that is it applies to an instance of a class .
class	An inclusive term for entity class and substance class .

class characteristic	A characteristic that is applicable to a class not its instances .
class specialising scheme	An inclusive term for entity class specialising scheme and substance class specialising scheme .
classification	The generic term for all ways in which a type definition may be applied to a <i>subject</i> .
classification structure	A hierarchy of class schemes and classes . The top of the hierarchy is a class that has a usage rule of root .
common fragment group	A fragment is used to define a number of quantities all refer to the same class or category .
common reference quantity	A group of quantity complex characteristics that are all quantities of the same thing.
Common Standard (CS)	An information definition that is not a perspective but a way of holding the information that is capable of representing all the information from all the different information perspectives within an organisation.
complex characteristic	A characteristic that consists of more than one fragment .
composition	The internal structure of a simple fragment consisting of three parts: data type ; composition qualifying detail and composition constraint on value .
composition constraint on value	A constraint on the values that are valid for a characteristic that could be for example: minimum, maximum or a range of values, or a catalogue of permitted values.
composition qualifying detail.	Further details of the form of a valid value for the characteristic that could be for example: the character alphabet or the language used for a data type of character , an applicable unit of measure for a data type of numeric , or an applicable unit time, the applicable calendar, a time datum, a time period of applicability for a data type of time .
compound fragment	A sub-part of a characteristic that consists of a combination of other compound fragments and/or simple fragments .
context	The purpose of the and defined by the domain that it applies to and the scope of the information definition .
context at a level	A combination of context and level .
contingent	A characteristic defined for a category whose applicability to a class is contingent upon the class being categorised to the category the characteristic is defined for.
data type	The basic form of a value to be assigned to a fragment , there are five data types : character , numeric , boolean , time and referential .
dividing set	An inclusive term for entity category dividing set and substance category dividing set .

element	An inclusive term for a class, category, scheme, set, categorisation or characteristic .
element description	A textual description or explanatory information for a CBML element .
element name	A name for a CBML element .
element types	An inclusive term for all the different types of CBML elements , these are: entity class, substance class, entity category, substance category, entity class specialising scheme, substance class specialising scheme, entity categorising set, substance categorising set or characteristic, entity category dividing set, substance category dividing set, categorisation, simple characteristic, complex characteristic or characteristic group .
entity categorising set	A group of entity categories that share a common scope , delineating criteria, and are mutually exclusive.
entity category	A holder for class and instance contingent characteristics and possibly categorisation used to further describe an entity when the entity is classified through categorisation as belonging to the entity category based on rules defined in a CBML information definition .
entity category dividing set	A group of entity categories that are sub- categories of the same entity category as owner of the entity category dividing set .
entity class	Something that exists in the real-world as an individual object, either tangible or intangible, such as “person”, “vehicle”, “invoice”.
entity class specialisation structure	A hierarchical structure of entity classes and entity class specialising schemes , the top of the structure is always an entity class defined as the root .
entity class specialising scheme	A definition of the criteria to delineate the entity class , or possibly the entity category , that is the owner of the entity class specialising scheme into sub-classes of the entity class that are themselves entity classes and members of the entity class specialising scheme .
extant data	Data that exists in the implemented information systems.
fixed	Applies to characteristics and categorisations to indicate that once a value has been assigned to a characteristic or a categorisation has been defined it cannot be changed.
fragment	An inclusive term for simple fragment and compound fragment ; part of a characteristic that requires other fragments to form a characteristic .
generic complex	A complex characteristic whose fragments have not been explicitly defined.
group	GRP – A group consisting of more than one characteristic .

holder	This is a place where characteristics may be defined that is:
	<ul style="list-style-type: none"> Entity class Entity class specialising scheme Substance class specialising scheme Category Categorisation structure Category dividing set
information element	A single piece of information defined by a characteristic that has a value, or set of values, ascribed to it when used to describe an actual thing in the real world.
information fragment	A single value defined by a fragment that is a component of an information element .
instance	An individual thing in the real-world defined by an entity.
instance characteristic	A characteristic that is applicable to an instance of an entity class ; in real-world terms it is applicable to a real-world object.
intrinsic	Applied to characteristic and categorisation when the characteristic or categorisation is not dependent on contingency to be applicable to a holder .
intrinsic characteristic	An applicable characteristic independent of any contingency.
level	There are 3 levels, or type of information definition , within the CBML environment
mandatory	Applies to characteristics and categorisations to indicate that a value for a characteristic or a categorisation is required.
measurable property	A feature of an entity or substance that may be measured such as: dimension, area, volume, speed, mass, etc.
members	The classes or categories that are mutually exclusive sub-classes or sub-categories of the owner of that are defined by the scheme or set .
information definition	The combination of CBML elements defined for a context at a level .
information perspective	The user perspective from which an information definition is defined.

multiple	Applies to characteristics and categorisations to indicate that there may be more than one concurrent value for a characteristic or a categorisation . This is different to the presence of competing values through plurality .
native	Within an information definition the holder that is being described.
non-root	Not the top entity class or substance class in a specialisation hierarchy and hence the entity class or substance class is a member of a scheme . Whether an entity class or a substance class is or is not a root is a usage rule .
Normalised Physical Model (NPM)	A model of the physical representation of an implemented data store.
Normalised User Perspective (NUP)	An information definition that defines the information contained in a single Actual User Perspective in a form that places intrinsic information elements in appropriate holders
numeric	NUM – A data type whose value is a real number in the range of minus infinity to plus infinity including zero but excluding plus and minus infinity.
optional	Applies to characteristics and categorisations to indicate that a value for a characteristic or a categorisation may not be required.
OR	A set of CBML characteristics and/or characteristic groups and/or fragments that are alternatives. These alternatives are not mutually exclusive.
overall status	A value ascribed to an element to define the current state of the element definition within the approvals process.
owner	The class or category that a scheme or set is describing.
plurality	An inclusive term for source plurality and time plurality .
Physical Model (PM)	A model of the physical representation of an implemented data store.
quantity complex characteristic	A complex characteristic to represent a quantity requiring not only a numeric value but also “to what” the value applies.
real-world	The world external to an information definition .
real-world entity	An entity that is an identifiable thing, physical or otherwise, that may be encountered in the real world.
real-world substance	Something that is tangible and exists in the real world for which there cannot be instances .
referential	REF – A data type that is a pointer to a class , scheme , category or set .

referred	The class , scheme , category or set that is referred to either by a referential , the class or category referred to as owner of a scheme or set , or the scheme or set referred to by the members of a scheme or set .
characteristic group	A characteristic group that has a multiple usage rule .
root	The top entity class or substance class in a specialisation hierarchy and hence the entity class or substance class is not a member of a scheme . Whether an entity class or a substance class is or is not a root is a usage rule .
scheme	An inclusive term for entity class specialising scheme and substance class specialising scheme .
scope	The class or category that is being categorised.
set	An inclusive term for entity categorising set , substance categorising set , entity category dividing set and substance category dividing set .
simple characteristic	The simplest part of a characteristic that represents a single item of data with a defined composition .
simple fragment	A fragment that represents a single piece of data.
single	Applies to characteristics and categorisations to indicate that there is only one concurrent value for a characteristic or a categorisation . This does not prevent the presence of competing values through plurality .
source plurality	Applies to characteristics and categorisations . For a characteristic it asserts that the value for a simple characteristic or set of values for a complex characteristic , may vary dependent upon the source of the information that defined value or set of values. For a categorisation it asserts that the member or members of the categorising set to which a class or instance is categorised may vary dependent upon the source of the information that defined the member or members .
specialisation structure	An inclusive term for entity class specialisation structure or a substance class specialisation structure .
specialising classification	A generic term for the specialisation structure .
specialising scheme	An inclusive term for entity class specialising scheme or a substance class specialising scheme .
status by authority	An indicator of whether the definition of a CBML element is approved by a particular authority. There may be a number of authorities that need to approve a definition during the approval process.
sub-category	A category that is belongs exclusively to a higher order category but has one or more additional characteristics at least one of which will have a constant value.

sub-class	A class that is belongs exclusively to a higher order class but has one or more additional characteristics and/or categorisations at least one of which will have a constant value.
substance categorising set	A group of substance categories that share a common scope and are mutually exclusive.
substance category	A category that defines class characteristics that may be applicable to a substance class based on rules defined in a CBML information definition .
substance category dividing set	A group of substance categories that are sub- categories of the same substance category as owner of the substance category dividing set .
substance class	Something that is tangible and exists in the real-world but never as an individual object for example “petrol”, “water”, “carbon dioxide” etc.
substance class categorisation structure	A hierarchical structure of substance categories and substance categorising sets that always has an entity class at the top.
substance class specialisation structure	A hierarchical structure of substance classes and substance class specialising schemes , the top of which is always a substance class defined as the root .
substance class specialising scheme	A definition of the criteria to delineate the substance class that is the owner of the substance class specialising scheme into sub-classes of the substance class that are themselves substance classes and members of the substance class specialising scheme .
sub-type	A class or category that defines explicit divisions of another class or category .
time	DTG – A data type whose value is a position or period in the continuum of time.
time plurality	Applies to characteristics and categorisations . For a characteristic it asserts that as the value for a simple characteristic or set of values for a complex characteristic , change with time these different values or sets of values may compete for relevance. For a categorisation it asserts that the member or members of the categorising set to which a class or instance is categorised change with time these different member or members of the categorising set may compete for relevance.
unique	Having the quality of uniqueness .
uniqueness	The value for a simple characteristic or set of values for a complex characteristic is unique within a defined uniqueness scope .

uniqueness scope	The scope within which uniqueness applies, for example within all instances of a class , within a single value of another characteristic .
unit of measure	An amount or quantity adopted as a standard of measurement for amounts or quantities of the same kind.
unit of measure group	A set of units of measure for the defined measurable property such as metric or imperial.
usage rule	Rules that apply to some CBML elements within a context at a level the same element potentially having different usage rules in a different context at a level . The applicable usage rules for characteristics and categorisations are: optional/mandatory , single/multiple, fixed/variable , time plurality , source plurality . The applicable usage rule for classes is root or not root . The applicable usage rule for characteristics in a characteristic group is optional/mandatory .
variable	Applies to characteristics and categorisations to indicate that once a value has been assigned to a characteristic or a categorisation it may be changed with time.
XOR	A set of CBML characteristics and/or characteristic groups and/or fragments that are alternatives. These alternatives are mutually exclusive.

ANNEX B VALID REFERENTIALS

THE VALIDITY OF REFERENTIAL COMBINATIONS ARE:

Native		Referred		Validity and referential target
type	Characteristic class or instance	type	Referred class or instance	
class	class	class	class	Not valid
class	class	class	instance	Not valid
class	class	category	class	Not valid
class	class	category	instance	Not valid
class	class	scheme	class	Not valid
class	class	scheme	instance	Not valid
class	class	set	class	Not valid
class	class	set	instance	Not valid
entity class	instance	class	class	Valid. The referred class
entity class	instance	class	instance	Valid. An instance of the referred class
entity class	instance	category	class	Valid. A class that is categorised to the referred category
entity class	instance	category	instance	Valid. An instance of a class that is categorised to the referred category
entity class	instance	scheme	class	Valid. A member of the referred scheme
entity class	instance	scheme	instance	Valid. An instance of a member of the referred scheme
entity class	instance	set	class	Valid. A class that is categorised to the referred set
entity class	instance	set	instance	Valid. An instance of a class that is categorised to the referred set
substance class	instance	class	class	Not valid

Native		Referred		Validity and referential target
type	Characteristic class or instance	type	Referred class or instance	
substance class	instance	class	instance	Not valid
substance class	instance	category	class	Not valid
substance class	instance	category	instance	Not valid
substance class	instance	scheme	class	Not valid
substance class	instance	scheme	instance	Not valid
substance class	instance	set	class	Not valid
substance class	instance	set	instance	Not valid
category	class	class	class	Valid. The referred class
category	class	class	instance	Valid. An instance of the referred class
category	class	category	class	Valid. A class that is categorised to the referred category
category	class	category	instance	Valid. An instance of a class that is categorised to the referred category
category	class	scheme	class	Valid. A member of the referred scheme
category	class	scheme	instance	Valid. An instance of a member of the referred scheme
category	class	set	class	Valid. A class that is categorised to the referred set
category	class	set	instance	Valid. An instance of a class that is categorised to the referred set
category	instance	class	class	Valid. The referred class

Native		Referred		Validity and referential target
type	Characteristic class or instance	type	Referred class or instance	
category	instance	class	instance	Valid. An instance of the referred class
category	instance	category	class	Valid. A class that is categorised to the referred category
category	instance	category	instance	Valid. An instance of a class that is categorised to the referred category
category	instance	scheme	class	Valid. A member of the referred scheme
category	instance	scheme	instance	Valid. An instance of a member of the referred scheme
category	instance	set	class	Valid. A class that is categorised to the referred set
category	instance	set	instance	Valid. An instance of a class that is categorised to the referred set
scheme	class	class	class	Valid. The referred class
scheme	class	class	instance	Valid. An instance of the referred class
scheme	class	category	class	Valid. A class that is categorised to the referred category
scheme	class	category	instance	Valid. An instance of a class that is categorised to the referred category
scheme	class	scheme	class	Valid. A member of the referred scheme
scheme	class	scheme	instance	Valid. An instance of a member of the referred scheme
scheme	class	set	class	Valid. A class that is categorised to the referred set

Native		Referred		Validity and referential target
type	Characteristic class or instance	type	Referred class or instance	
scheme	class	set	instance	Valid. An instance of a class that is categorised to the referred set
scheme	instance	class	class	Valid. The referred class
scheme	instance	class	instance	Valid. An instance of the referred class
scheme	instance	category	class	Valid. A class that is categorised to the referred category
scheme	instance	category	instance	Valid. An instance of a class that is categorised to the referred category
scheme	instance	scheme	class	Valid. A member of the referred scheme
scheme	instance	scheme	instance	Valid. An instance of a member of the referred scheme
scheme	instance	set	class	Valid. A class that is categorised to the referred set
scheme	instance	set	instance	Valid. An instance of a class that is categorised to the referred set
set	class	class	class	Valid. The referred class
set	class	class	instance	Valid. An instance of the referred class
set	class	category	class	Valid. A class that is categorised to the referred category
set	class	category	instance	Valid. An instance of a class that is categorised to the referred category
set	class	scheme	class	Valid. A member of the referred scheme
set	class	scheme	instance	Valid. An instance of a member of the referred scheme

Native		Referred		Validity and referential target
type	Characteristic class or instance	type	Referred class or instance	
set	class	set	class	Valid. A class that is categorised to the referred set
set	class	set	instance	Valid. An instance of a class that is categorised to the referred set
set	instance	class	class	Valid. The referred class
set	instance	class	instance	Valid. An instance of the referred class
set	instance	category	class	Valid. A class that is categorised to the referred category
set	instance	category	instance	Valid. An instance of a class that is categorised to the referred category
set	instance	scheme	class	Valid. A member of the referred scheme
set	instance	scheme	instance	Valid. An instance of a member of the referred scheme
set	instance	set	class	Valid. A class that is categorised to the referred set
set	instance	set	instance	Valid. An instance of a class that is categorised to the referred set

Intentionally Left Blank

