UN/CEFACT

DRAFT

United Nations Centre for Trade Facilitation and Electronic Business

UN/CEFACT – ebXML Core Components Technical Specification

11 December 2002
Version 1.90

1 Status of This Document

This *UN/CEFACT – ebXML Technical Specification* is being developed in accordance with the UN/CEFACT/TRADE/22 Open Development Process for Technical Specifications. It has been approved by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) Techniques and Methodology Group (TMG) for implementation verification as defined in Step 6 of the Open Development Process.

This document contains information to guide in the interpretation or implementation of ebXML concepts.

Distribution of this document is unlimited.

The document formatting is based on the Internet Society's Standard RFC format.

This version: *UN/CEFACT – ebXML Core Components Technical Specification*, Version 1.9 of 11 December 2002

Previous version: *UN/CEFACT – ebXML Core Components Technical Specification*, Version 1.85 of 30 September 2002

2 UN/CEFACT – ebXML Core Components Technical Specification Project Team Participants

We would like to recognise the following for their significant participation to the development of this Technical Specification.

Project Team Leader:	Hartmut Hermes	Siemens
Lead Editor:	Mark Crawford	Logistics Management Institute
Editing Team	Mike Adcock	APACS
	Mary Kay Blantz	AIAG
	Arofan Gregory	AEON Consulting
	Alan Stitzer	Marsh, Inc.
	Frank Vandamme	SWIFT
	James Whittle	e Centre

Contributors:

Bernd Boesler	DIN
Todd Boyle	NetAccount
Kerstien Celis	Seagha c.v.
Jean-Luc Champion	Enterprise Integration Partners
Marianne Cockle	APACS
Scott Colthurst	State Farm
Alain Dechamps	CEN/ISSS
Eduardo Gutentag	Sun Microsystems
Paula Heilig	Worldspan
Stig Korsgaard	Danish Bankers Association
Melanie McCarthy	General Motors
Sue Probert	Commerce One
Andreas Schultz	DKV
Lisa Seaburg	AEON Consulting
Gunther Stuhec	SAP AG
Hisanao Sugamata	ECOM-Japan
Herbert Thomas	AustriaPro
Fred Van Blommestein	Berenschot
Nigel Wooden	ACORD

3 Table of Contents

4 Introduction

This *UN/CEFACT – ebXML Core Components Technical Specification* describes and specifies a new approach to the well-understood problem of the lack of information interoperability between applications in the e-business arena. Traditionally, standards for the exchange of business data have been focused on static message definitions that have not enabled a sufficient degree of interoperability or flexibility. A more flexible and interoperable way of standardising *Business Semantics* is required. The UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) – ebXML *Core Component* solution described in this specification presents a methodology for developing a common set of semantic building blocks that represent the general types of business data in use today and provides for the creation of new business vocabularies and restructuring of existing business vocabularies.

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in Internet Engineering Task Force (IETF) Request For Comments (RFC) 2119.1

4.1 Scope and Focus

This *UN/CEFACT – ebXML Core Components Technical Specification* can be employed wherever business information is being shared or exchanged amongst and between enterprises, governmental agencies, and/or other organisations in an open and worldwide environment. The *Core Components User Community* consists of business people, business document modellers and business data modellers, *Business Process* modellers, and application developers of different organisations that require interoperability of business information. This interoperability covers both interactive and batch exchanges of business data between applications through the use of Internet and Web based information exchanges as well as traditional Electronic Data Interchange (EDI) systems.

This specification will form the basis for standards development work of business analysts, business users and information technology specialists supplying the content of and implementing applications that will employ the UN/CEFACT *Core Component Library* (CCL). The *Core Component Library* will be stored in a UN/CEFACT repository and identified in an ebXML compliant registry.

Due to the evolving nature of the UN/CEFACT *Core Component Library*, the specification includes material that focuses on the business community doing further discovery and analysis work. Some of the contents of this specification are not typical of this type of technical document. However, they are critical for successful adoption and standardization in this area to move forward.

---

[1] *Key words for use in RFCs to Indicate Requirement Levels* - Internet Engineering Task Force, Request For Comments 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt?number=2119

4.2 Structure of this Specification

Due to the diversity of the intended audience, this document has been divided into five main Sections.

- Section 5: Working Process and Methodology for Business Users—Discovery, Harmonization, Assessment and How to Use [informative]
- Section 6: Technical Details—*Core Components* and *Context* [normative]
- Section 7: Technical Details—Storage and Metadata [normative]
- Section 8: Technical Details— Permissible *Representation Terms* and Approved *Core Component Type*, *Content*, and *Supplementary Components* [normative]
- Section 9: Definition of Terms [normative]

Sections 5, 6, 7 and 8 are complementary, but may also be used independently of each other. Section 5 is informative. A business audience may choose to read through the working process and methodology section (Section 5) and only reference the Technical Details (Sections 6, 7 and 8) as needed. Sections 6, 7 and 8 are normative. A technical audience may choose to focus on the technical details (Sections 6, 7, and 8), referring to the methodology (Section 5) and example (published as a supplemental document) sections as appropriate, using the current permissible *Representation Terms* and approved *Core Component Type*, *Content*, and *Supplementary Components* (Section 8) and the glossary (Section 9).

In addition, the UN/CEFACT Forum will prepare supplemental documents that may be used in conjunction with this *Core Components Technical Specification.* These supplemental documents will include:

- *Message Assembly* – expands on the *Assembly* principles and *Constraints Language* contained in the *Core Components Technical Specification* and provides specific methodology for assembling higher level *Business Information Entities* for electronic messages.
- *Core Components Primer* – details how the contents of Sections 5, 6, and 7 would be used in practice to create a library of *Core Components* and *Business Information Entities.*
- *Catalogue of Core Components* – represents the work of various organizations working in a joint endeavour to develop and publish semantically correct and meaningful information exchange parcels.

4.2.1 Notation

[Definition] – A formal definition of a term. Definitions are normative.

[Example] – A representation of a definition or a rule. Examples are informative.

[Note] – Explanatory information. Notes are informative.

[Rn] – Identification of a rule that requires conformance to ensure discovered *Core Components* are properly discovered, named and stored. The value R is a prefix to categorise the type of rule where R=A for Conformance rule, R=B for *Business Information Entity* rule, R=C for *Core Component* rule, R=D for *Data Type* rule, or R=S for *Storage* rule; and n (1..n) indicates the sequential number of the rule]. Rules are normative. In order to ensure continuity across versions of the specification, rule numbers that are deleted will not be re-issued, and any new rules will be assigned the next higher number - regardless of location in the text.

Italics – All words appearing in italics, when not titles or used for emphasis, are special terms defined in Section 9.

`Courier` – All words appearing in bolded **`courier font`** are values or objects.

4.3 Conformance

Applications will be considered to be in full conformance with this technical specification if they comply with the content of normative sections, rules and definitions.

[A1] Conformance shall be determined through adherence to the content of normative sections, rules and definitions.

4.4 Related Documents

The following documents provided significant levels of influence in the development of this document:

- ebXML Technical Architecture Specification v1.04
- ebXML Business Process Specification Schema v1.01
- OASIS/ebXML Registry Information Model v2.0
- OASIS/ebXML Registry Services Specification v2.0
- ebXML Requirements Specification v1.06
- OASIS/ebXML Collaboration-Protocol Profile and Agreement Specification v2.0
- OASIS/ebXML Message Service Specification v2.0
- ebXML Technical Report, Business Process and Business Information Analysis Overview v1.0
- ebXML Business Process Analysis Worksheets & Guidelines v1.0
- ebXML Technical Report, E-Commerce Patterns v1.0
- ebXML Technical Report, Catalog of Common Business Processes v1.0
- ebXML Technical Report, Core Component Overview v1.05
- ebXML Technical Report, Core Component Discovery and Analysis v1.04
- ebXML Technical Report, Context and Re-Usability of Core Components v1.04

- ebXML Technical Report, Guide to the Core Components Dictionary v1.04
- ebXML Technical Report, Naming Convention for Core Components v1.04
- ebXML Technical Report, Document Assembly and Context Rules v1.04
- ebXML Technical Report, Catalogue of Context Categories v1.04
- ebXML Technical Report, Core Component Dictionary v1.04
- ebXML Technical Report, Core Component Structure v1.04
- Information Technology - Metadata registries: Framework for the Specification and Standardization of Data Elements, International Standardization Organization, ISO 11179-1
- Information Technology - Metadata registries: Classification of Concepts for the Identification of Domains, International Standardization Organization, ISO 11179-2
- Information Technology - Metadata registries: Registry Metamodel, International Standardization Organization, ISO 11179-3
- Information Technology - Metadata registries: Rules and Guidelines for the Formulation of Data Definitions, International Standardization Organization, ISO 11179-4
- Information Technology - Metadata registries: Naming and Identification Principles for Data Elements, International Standardization Organization, ISO 11179-5
- Information Technology - Metadata registries: Framework for the Specification and Standardization of Data Elements, International Standardization Organization, ISO 11179-6

4.5 Overview

This *Core Components Technical Specification* provides a way to identify, capture and maximise the re-use of business information to support and enhance information interoperability across multiple business situations. The specification focuses both on human-readable and machine-processable representations of this information.

The *Core Components* approach described in this document is more flexible than current standards in this area because the semantic standardisation is done in a syntax-neutral fashion. Using *Core Components* as part of the ebXML framework will help to ensure that two trading partners using different syntaxes [e.g. Extensible Markup Language (XML) and United Nations/EDI for Administration, Commerce, and Transport (UN/EDIFACT)] are using *Business Semantics* in the same way on condition that both syntaxes have been based on the same *Core Components*. This enables clean mapping between disparate message definitions across syntaxes, industry and regional boundaries.

UN/CEFACT *Business Process* and *Core Component* solutions capture a wealth of information about the business reasons for variation in message semantics and structure. In the past, such variations have introduced incompatibilities. The *Core Components* mechanism uses this rich information to allow identification of exact similarities and differences between semantic models. Incompatibility becomes incremental rather than wholesale, i.e. the detailed points of difference are noted, rather than a whole model being dismissed as incompatible.

4.6 Key Concepts

The *Core Components Technical Specification* key concepts cover two focus areas—*Core Components* and *Business Information Entities*. Each of these focus areas is discussed in the following subsections. In each subsection concepts are introduced, followed by a normative definition and, where appropriate, an example for each.

4.6.1 Key Core Component Concepts

The central concept of this specification is the *Core Component*. The *Core Component* is a semantic building block, which is used as a basis to construct all electronic business messages.

[Definition] *Core Component* (CC)

A building block for the creation of a semantically correct and meaningful information exchange package. It contains only the information pieces necessary to describe a specific concept.

There are four different categories of *Core Components*: *Basic Core Component*, *Association Core Component*, *Core Component Type* and *Aggregate Core Component*. The following definitions explain each of these:

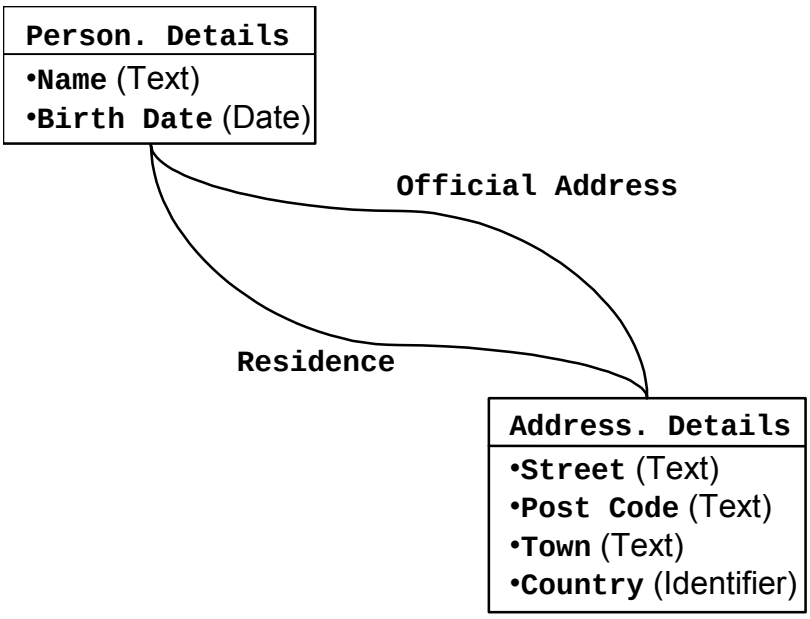
[Definition] *Basic Core Component* (BCC)

A *Core Component* which constitutes a singular business characteristic of a specific *Aggregate Core Component* that represents an *Object Class*. It has a unique *Business Semantic* definition. A *Basic Core Component* represents a *Basic Core Component Property* and is therefore of a *Data Type*, which defines its set of values. *Basic Core Components* function as the *Properties* of *Aggregate Core Components*.

[Definition] *Association Core Component* (ASCC)

A *Core Component* which constitutes a complex business characteristic of a specific *Aggregate Core Component* that represents an *Object Class*. It has a unique *Business Semantic* definition. An *Association Core Component* represents an *Association Core Component Property* and is associated to an *Aggregate Core Component*, which describes its structure.

[Example] *Association Core Component*



The example shows two *Aggregate Core Components*, **Person. Details** and **Address. Details**. Each *Aggregate Core Component* has a number of *Properties* (i.e. business characteristics). The *Aggregate Core Component* **Person. Details** has four Properties, namely **Name**, **Birth Date**, **Residence** and **Official Address**. The *Aggregate Core Component* **Address. Details** also has four *Properties*, namely **Street**, **Post Code**, **Town** and **Country**.

Most of these *Properties* are *Basic Core Components*. These *Properties* represent a singular business characteristic and their set of allowed values is defined by a *Data Type*. The *Data Types* **Name**, **Street**, **Post Code** and **Town** are of the *Data Type* **Text**, **Birth Date** is of the *Data Type* **Date** and **Country** is of the *Data Type* **Identifier**.

The other *Properties* are *Association Core Components*. They represent complex business characteristics and their structure is therefore defined by another *Aggregate Core Component*. **Residence** and **Official Address** are both *Association Core Components* and their structure is described by **Address. Details**.

This example will therefore result in following set of *Core Components*:

- **Person. Details** (*Aggregate Core Component*)
- **Person. Name. Text** (*Basic Core Component*)
- **Person. Birth. Date** (*Basic Core Component*)
- **Person. Residence. Address** (*Association Core Component*)
- **Person. Official. Address** (*Association Core Component*)

[Example] *Association Core Component* (Continued)

- **Address. Details** (*Aggregate Core Component*)
- **Address. Street. Text** (*Basic Core Component*)
- **Address. Post Code. Text** (*Basic Core Component*)
- **Address. Town. Text** (*Basic Core Component*)
- **Address. Country. Identifier** (*Basic Core Component*)

[Definition] *Core Component Type* (CCT)

A *Core Component*, which consists of one and only one *Content Component*, that carries the actual content plus one or more *Supplementary Components* giving an essential extra definition to the *Content Component*. *Core Component Types* do not have *Business Semantics*.

[Example] *Core Component Types*

For a *Core Component Type* of **Amount. Type**, the *Content Component* carries the value of **12**. This value has no semantic meaning on its own. But **12 Euro**, where **Euro** is the *Supplementary Component* that gives essential extra definition to the *Content Component*, does have meaning.

[Definition] *Aggregate Core Component*

A collection of related pieces of business information that together convey a distinct business meaning, independent of any specific *Business Context*. Expressed in modelling terms, it is the representation of an *Object Class*, independent of any specific *Business Context*.

[Example] – *Aggregate Core Component*

Aggregate: **Financial Account. Details**[2]

Definition: A service through a bank or other organization through which funds are held on behalf of a client.

Basic Core Components:

- **Financial Account. Identifier**
- **Financial Account. Name**

[2] See section 6.1.4 for detailed rules for developing *Core Component* names.

[Example] – *Aggregate Core Component* (Continued)

- **Financial Account. Country. Identifier**
- **Financial Account. Product Type. Identifier**
- **Financial Account. Nickname. Name**

Core Components (and *Business Information Entities*) have *Properties* that are defined by *Data Types*.

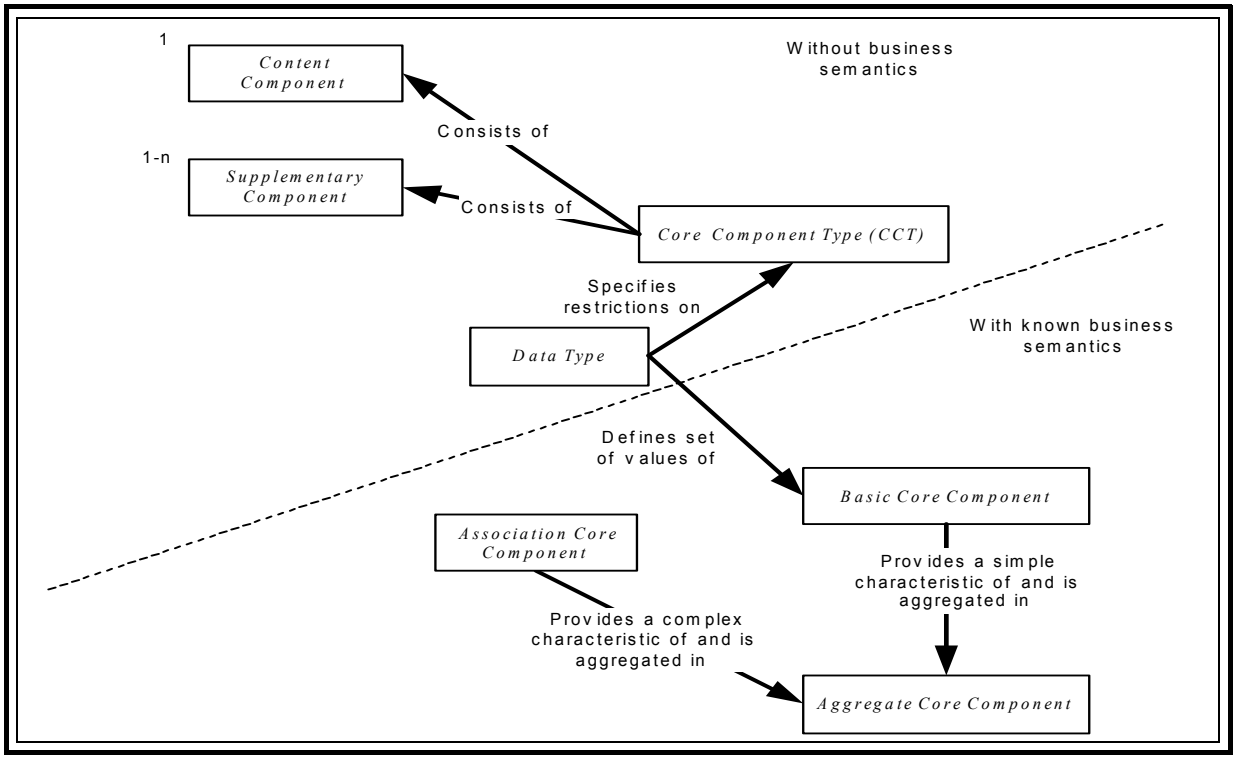
A *Data Type* represents the full range of values that shall be used for the representation of a particular *Core Component Property*. A *Data Type* must be based on one of the *Core Component Types*, but may include restrictions of the set of values of that *Core Component Type's Content Component* and/or *Supplementary Component(s)*.

[Definition] – *Data Type*

Defines the set of valid values that can be used for a particular *Basic Core Component Property* or *Basic Business Information Entity Property*. It is defined by specifying restrictions on the *Core Component Type* that forms the basis of the *Data Type*.

The diagram in Figure 4-1 shows the relationships between the various *Core Component* elements.

Figure 4-1. Core Component Overview



4.6.2 Key Business Information Entity Concepts

The key differentiator between *Core Components* and *Business Information Entities* is the concept of *Business Context*. *Business Context* is a mechanism for qualifying and refining *Core Components* according to their use under particular business circumstances. Once *Business Contexts* are identified, *Core Components* can be differentiated to take into account any necessary qualification and refinement needed to support the use of the *Core Component* in the given *Business Context.* The *Business Process* definition provides a high level description of the use of a message and its contents.[3]

[Definition] *Business Context*

The formal description of a specific business circumstance as identified by the values of a set of *Context Categories*, allowing different business circumstances to be uniquely distinguished.

When a *Core Component* is used in a real business circumstance it serves as the basis of a *Business Information Entity*. The *Business Information Entity* is the result of using a *Core Component* within a specific *Business Context*.

[Definition] *Business Information Entity* (BIE)

A piece of business data or a group of pieces of business data with a unique *Business Semantic* definition. A *Business Information Entity* can be a *Basic Business Information Entity* (BBIE), an *Association Business Information Entity* (ASBIE), or an *Aggregate Business Information Entity* (ABIE).

A specific relationship exists between *Core Components* and *Business Information Entities*. *Core Components* and *Business Information Entities* are complementary in many respects. *Core Components* are intended to be the linchpin for creating interoperable *Business Process* models and business documents using a *Controlled Vocabulary*.

There are three different categories of Business Information Entities: Basic Business Information Entity, Association Business Information Entity, and Aggregate Business Information Entity. The most primitive of these is the Basic Business Information Entity. A Basic Business Information Entity is a Basic Core Component used in a specific Business Context.

[Definition] *Basic Business Information Entity* (BBIE)

A *Business Information Entity* that represents a singular business characteristic of a specific *Object Class* in a specific *Business Context*. It has a unique *Business Semantic* definition. A *Basic Business Information Entity* represents a *Basic Business Information Entity Property* and is therefore linked to a *Data Type*, which describes it values. A *Basic Business Information Entity* is derived from a *Basic Core Component*.

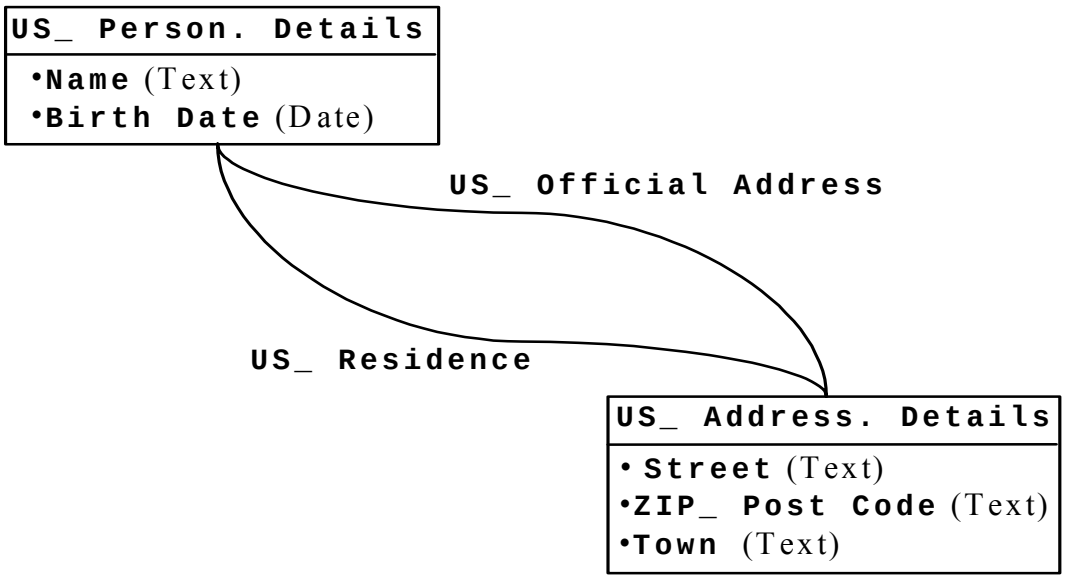
[3] The *Core Component*s' *Context* mechanism provides the more detailed linkage between specific business data and the exact circumstances of its business use.

Whenever a *Property* of an *Aggregate Business Information Entity* is of a complex nature, and has the structure of another *Aggregate Business Information Entity*, an *Association Business Information Entity* is used to represent that *Property*. An *Association Business Information Entity* is based on an *Association Core Component*, but exists in a *Business Context*.

[Definition] *Association Business Information Entity* (ASBIE)

A *Business Information Entity* that represents a complex business characteristic of a specific *Object Class* in a specific *Business Context*. It has a unique *Business Semantic* definition. An *Association Business Information Entity* represents an *Association Business Information Entity Property* and is associated to an *Aggregate Business Information Entity*, which describes its structure. An *Association Business Information Entity* is derived from an *Association Core Component*.

[Example] *Association Business Information Entity*



The example shows two *Aggregate Business Information Entities*, **US_ Person. Details** and **US_ Address. Details**. Each *Aggregate Business Information Entity* has a number of *Properties* (i.e. business characteristics). **US_ Person. Details** has

four *Properties*, namely **Name**, **Birth Date**, **US_ Residence** and **US_ Official Address**. The *Aggregate Business Information Entity* **US_ Address. Details** has three *Properties*, namely **Street**, **ZIP_ Post Code** and **Town**.

Most of these *Properties* are *Basic Business Information Entities*. They represent a singular business characteristic and their set of allowed values is defined by a *Data Type*. The *Data Types* **Name**, **Street**, **ZIP_ Post Code** and **Town** are of the *Data Type* **Text** and the *Data Type* **Birth Date** is of the *Data Type* **Date**.

The other *Properties* are *Association Business Information Entities*. They represent complex business characteristics and their structure is therefore defined by another

[Example] *Association Business Information Entity* (Continued)

Aggregate Business Information Entity. **US_ Residence** and **US_ Official Address** are both *Association Business Information Entities* and their structure is described by **US_ Address. Details**.

This example will therefore result in following set of *Business Information Entities*:

- **US_ Person. Details** (*Aggregate Business Information Entity*)
- **US_ Person. Name. Text** (*Basic Business Information Entity*)
- **US_ Person. Birth. Date** (*Basic Business Information Entity*)
- **US_ Person. US_ Residence. US_ Address** (*Association Business Information Entity*)
- **US_ Person. US_ Official. US_ Address** (*Association Business Information Entity*)
- **US_ Address. Details** (*Aggregate Business Information Entity*)
- **US_ Address. Street. Text** (*Basic Business Information Entity*)
- **US_ Address. ZIP_ Post Code. Text** (*Basic Business Information Entity*)
- **US_ Address. Town. Text** (*Basic Business Information Entity*)

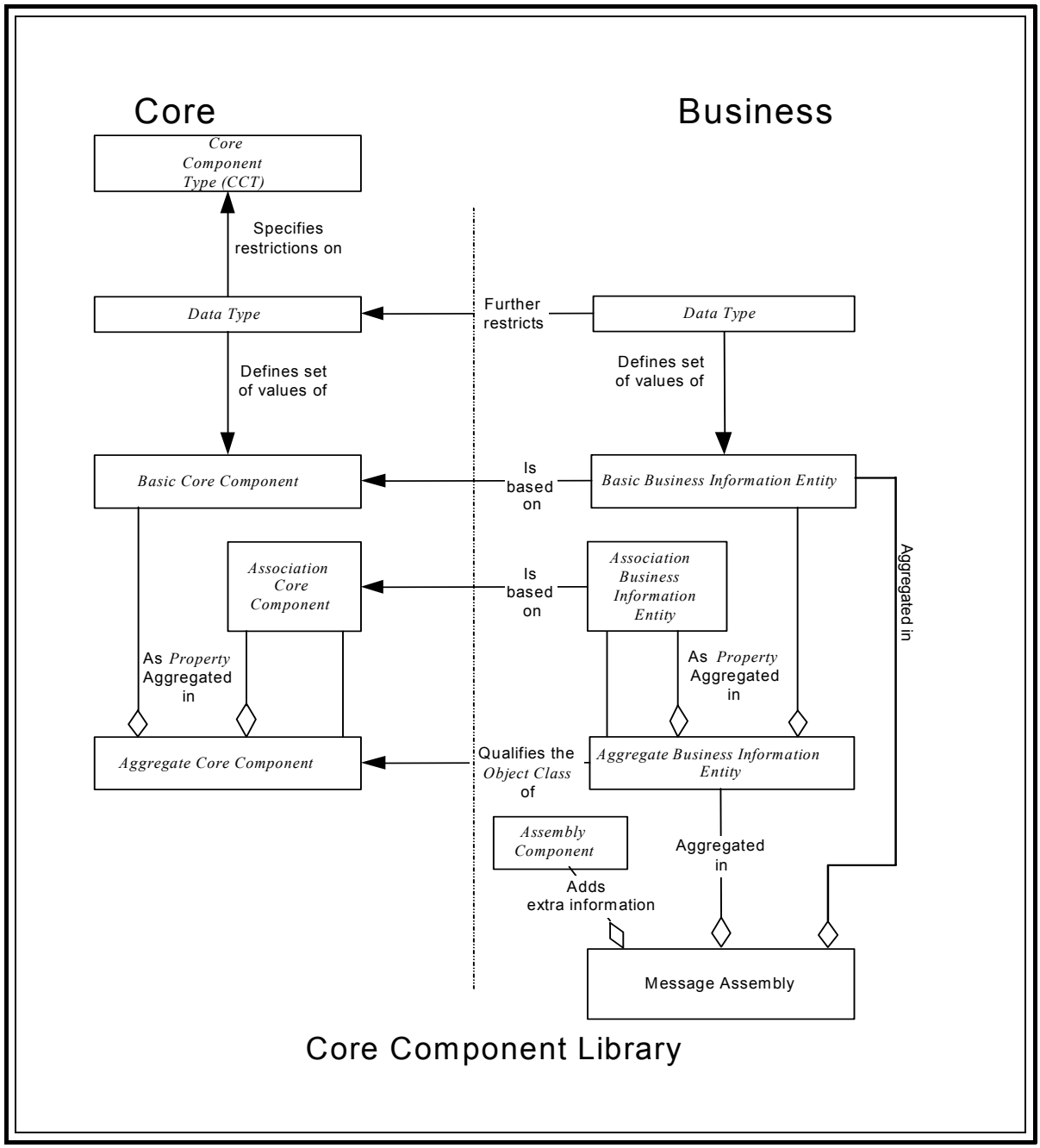
An *Aggregate Business Information Entity* is a piece of business data or a group of pieces of business data with a unique *Business Semantic* definition in a specific *Business Context*.

[Definition] *Aggregate Business Information Entity*

A collection of related pieces of business information that together convey a distinct business meaning in a specific *Business Context*. Expressed in modelling terms, it is the representation of an *Object Class*, in a specific *Business Context*.

The features of the relationship between *Core Components* and *Business Information Entities* are described in Figure 4-2.

Figure 4-2. Relationships between Core Components and Business Information Entities



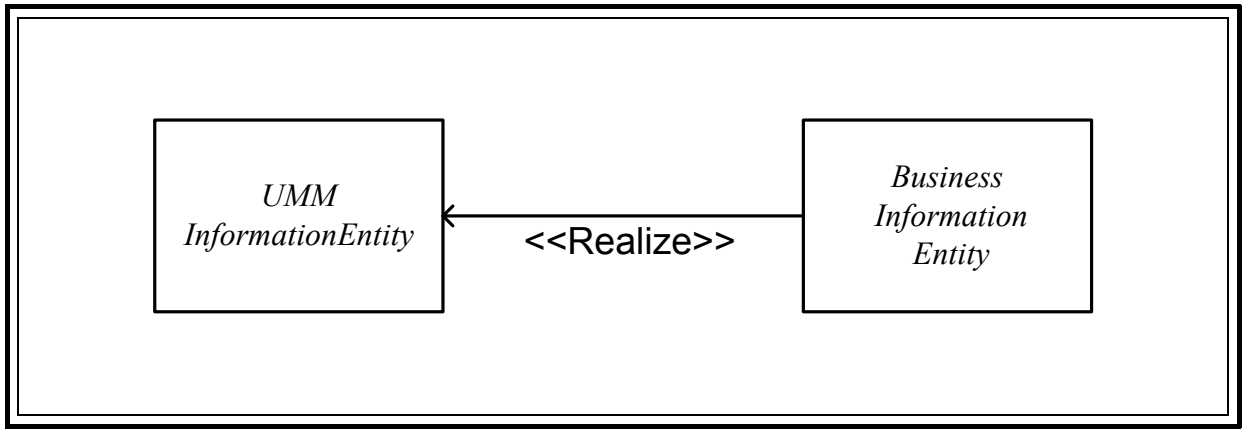
[Note]

The term *Core Component* is used as a generic term that encompasses *Basic Core Components, Association Core Components, Aggregate Core Components*, and their associated *Core Component Types.* Equally, the term *Business Information Entity* is used as a generic term encompassing *Basic Business Information Entities, Association Business Information Entities,* and *Aggregate Business Information Entities.*

4.7 Relationship between UN/CEFACT Modelling Methodology and Core Components

UN/CEFACT has developed the *UN/CEFACT Modelling Methodology* (UMM). UMM describes a Unified Modeling Language (UML) based modelling approach to develop *UMM InformationEntities*.[4] Within UN/CEFACT standards efforts, the *Core Component* framework of *Core Components* and *Business Information Entities* prescribes the mechanism for discovery, normalization, *Context* specialization and structure of *UMM InformationEntities*. The *Aggregate Business Information Entity-Basic Business Information Entity* framework provides the structure for components of the body of the business document. The *Core Component-Business Information Entity-Context* mapping framework provides the basis for mapping *UMM InformationEntity* realisations to business entities. The *Business Information Entity* to *Core Component* relationship provides the dictionary reference as specified in the information model abstract syntax. The UN/CEFACT *Core Component Library* is an implementation of the *UN/CEFACT Modelling Methodology* dictionary concept. The *Basic Core Component* is the realization of a non-aggregate *UMM InformationEntity* and provides the mapping to *Data Types*. The relationship between the *Core Component Framework* and the *UMM InformationEntity* is illustrated in Figure 4-3.

Figure 4-3. Relationship between Core Component Framework and UMM InformationEntity



[4] The UN/CEFACT Modelling Methodology (UMM) is a methodology for *Business Process* and information modelling that is based on the Object Management Group's Unified Modeling Language.

5 Working Process and Methodology

This section identifies aspects of *Core Component* working processes and methodologies. It includes an overview of the discovery and usage characteristics of *Core Components*. In addition, it includes detailed recommendations for conducting discovery, storage, approval, and application of *Context*.

5.1 Overview

The analysis of *Business Processes* builds a picture of requirements, identifying the business collaboration, i.e. timing and purpose of each process step. Detailed examination of the *Business Processes* at this level reveals the individual pieces of business information that are used and at what stage they are exchanged.

5.1.1 Discovery

A *Business Process* should be modelled using a standard approach. UN/CEFACT requires the *UN/CEFACT Modelling Methodology* (UMM) as the approach.[5] One of the results is a model, including a class diagram, which shows the business information and its inter-relationships. *Business Information Entities* can be identified from the *ebXML Business Process Analysis Worksheets and Guidelines*[6] that provide a simplified modelling approach.

For example, if a domain team has modelled the publication of catalogue data to trading partners, the result is a *Business Information Entity* representing the distributed catalogue data made up of a set of smaller *Business Information Entities* that are its component parts. Thus, the description of an item is identified as a *Business Information Entity* for this *Business Process*.

In order to improve interoperability across *Business Context*s, *Business Information Entities* must be based on a basic library of clearly defined semantic constructs to help ensure that they will inter-operate. This library must include a set of globally agreed semantic definitions such as those that will be contained in the *UN/CEFACT Core Components Library*.

A *Business Information Entity* is a *Core Component* used in a specific *Business Context* and given its own unique name. As *Basic Core Components* are single pieces of business information, when they are used directly in specific *Business Context*s the structure (components) does not change, but values may be restricted.

Just as each *Basic Business Information Entity* must ultimately be based on a *Basic Core Component*, each *Aggregate Business Information Entity* must ultimately be based on an existing *Aggregate Core Component*. The underlying *Aggregate Core Component* identifies the generic, standard definition of business information that is being used in the *Aggregate Business Information Entity*. The definition of the *Aggregate Business Information Entity* is based upon the generic description, being then modified and enhanced to be specific to the *Business Context* in which the *Aggregate Business Information Entity* is used. An *Aggregate*

[5] The UN/CEFACT Modelling Methodology (UMM) is a methodology for *Business Process* and information modelling that is based on the Unified Modeling Language.
[6] The ebXML Business Process Analysis Worksheets & Guidelines can be found at http://www.ebxml.org/

Business Information Entity is thus directly tied to a specific *Business Process* or to a *Business Context*. (See Section 5.6 for a fuller understanding of *Context*.)

[Example]

An invoicing *Business Process* uses a piece of information such as **Invoice. VAT_ Tax. Amount.*** **Invoice. VAT_ Tax. Amount** is a *Basic Business Information Entity* that is based on the *Basic Core Component* of **Invoice. Tax. Amount**. The invoicing *Business Process* is using **Invoice. Tax. Amount** in a specific *Business Context* where the *Business Process Context* = **Purchasing**, and the *Geopolitical Context* = **EU**. Therefore the application of *Context* adds a specialised definition, but in all other respects the *Basic Business Information Entity* is the same as the associated *Core Component* of **Invoice. Tax. Amount**, i.e. it has the same structure and *Data type*.

*In accordance with rule [B17], VAT would be defined as Value Added Tax in the definition for the *Basic Business Information Entity* of **Invoice. VAT_ Tax. Amount**.

When an *Aggregate Business Information Entity* has a complex *Property*, that *Property* is represented by an *Association Business Information Entity*. *Association Business Information Entities* are specific to their *Business Context*, and relate to *Association Core Components.* This relationship is the same as the relationship between *Aggregate Business Information Entities* and *Aggregate Core Components* and between *Basic Business Information Entities* and *Basic Core Components*. (See Figure 6-2 for a better understanding of this concept.)

An important aspect of information interoperability is that each *Business Information Entity* is based upon a *Core Component* structure and associated semantic definitions derived from the *Core Component Library*. The structure and definition of the *Business Information Entity* may be a refined and/or restricted *Version* of the structure and definition of the *Core Component* upon which it is based.

The following section describes the procedures by which the content of the UN/CEFACT ebXML compliant *Core Component Library* may be developed and maintained.

5.1.2 How to use UN/CEFACT Core Components

This section provides a procedure for the more technical user who wants to understand how to use *Core Components*. It assumes the user is dealing with an established set of *Core Components*, *Context Categories* and metadata/storage. The established set of *Core Components* being used should be based on those discovered, harmonized, and published by recognized standards groups. It is further assumed that the recognized standards group(s) and other business association group(s) have also made available sets of *Business Information Entities* for use in a published set of *Business Processes*.

5.1.2.1 Core Components and Semantic Interoperability

Today, the e-business community generally agrees on the definition of a standard message structure expressed as an UN/EDIFACT Message Implementation Guide (MIG), an XML

schema, or similar syntax specific representation. UN/CEFACT will produce standards based representations of these artefacts for implementation.[7]

Under the *Core Components* concept, defining and storing *Core Components* and associated *Context* mechanisms occur prior to the creation of a MIG or an XML schema. In this manner, the focus of the user changes from examining the MIG or XML schema, and moves to an examination of the semantic models. Accordingly, interoperability between syntaxes no longer depends on analysing specific instances, but naturally occurs during the *Business Process* model definition phase.

5.1.2.2 Overall Discovery and Document Design

Overall discovery and document design can be thought of as a series of steps that starts with determining the availability of existing *Business Process* definitions and ultimately results in standard business documents. Figure 5-1 illustrates this process. Specific steps to be followed are further described below.

Step 1: Search the registry/repository[8] – A search should be made in the registry to find the *Business Process* that meets the business requirement.

Step 1a: If no existing *Business Process* is found to be appropriate, then the new *Business Process* should be modelled using *UN/CEFACT Modelling Methodology* and submitted to the registry.

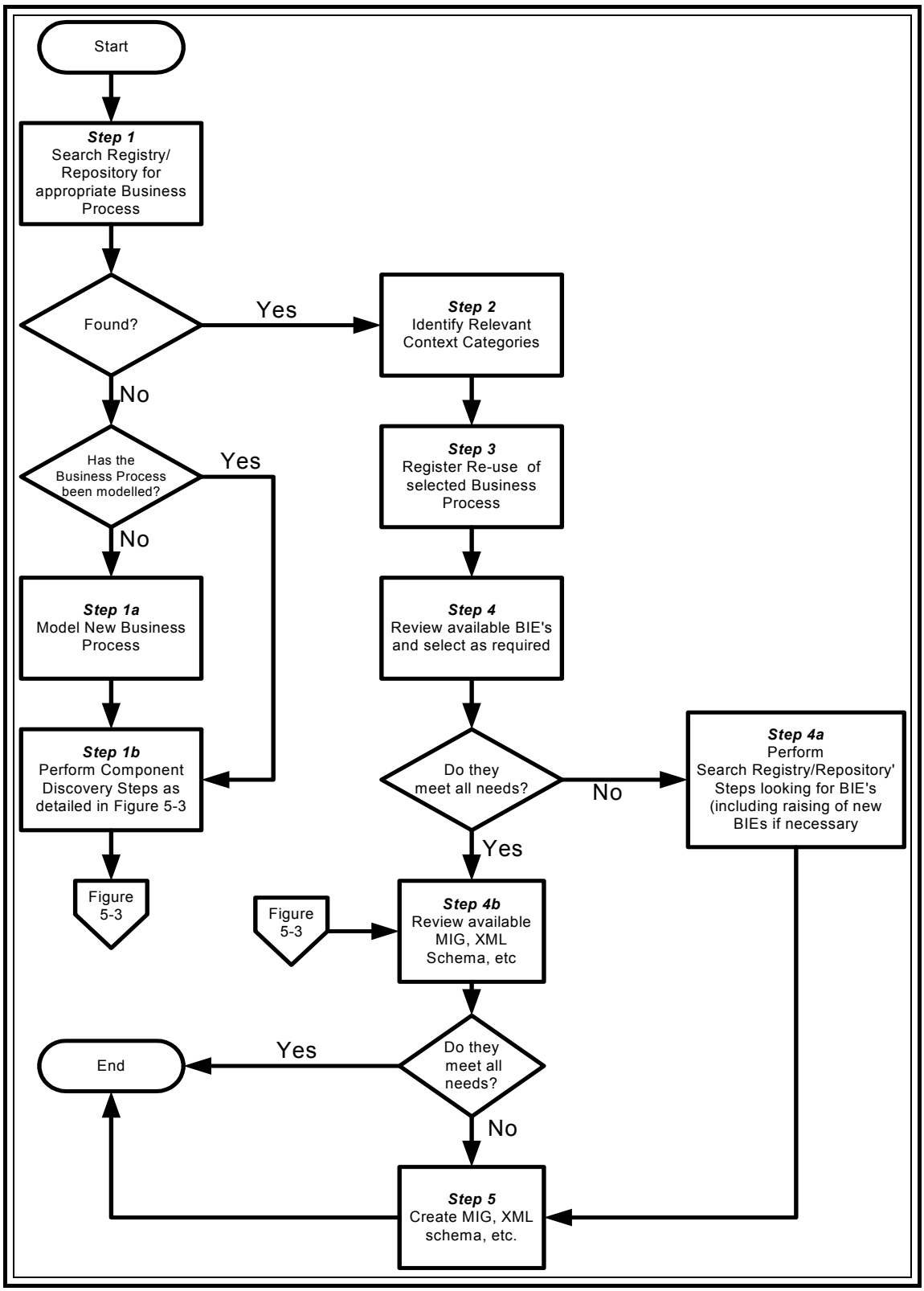
Step 1b: Conduct a thorough analysis of the business information requirements by following the *Core Component* discovery steps (Section 5.2)

Step 2: Identify relevant *Context Categories* – Access the registry interface and identify the relevant *Context Categories* of the selected *Business Process* by determining the following *Context Categories* (See Section 6.2.2):

- *Business Process Context* – Identify the interaction between trading partners to achieve a given business objective.
- *Product Classification Context* – Determine the goods or services concerned in the collaboration.
- *Industry Classification Context* – Determine the relevant trading partner industries.
- *Geopolitical Context* – Determine where the *Business Process* is to be conducted. Determine if the *Business Process* crosses regional, national, or international boundaries.
- *Official Constraints Context* – Determine any legal restrictions or requirements on this *Business Process.*

[7] The term XML schema includes XML Schema as defined in World Wide Web Consortium XML Schema Part 1: Structures XML Schema Definition Language, XML Document Type Definitions, Schematron, SOX, Relax NG, ASN.1, XDR, or any other notation that specifies the form and information content of an XML document.
[8] See the list of referred documents for explanation of 'registry/repository' within the ebXML architecture.

Figure 5-1. Steps from Business Process Discovery to Core Component Discovery



- *Business Process Role Context* – Identify the roles played by the trading partners. These can be derived from the *Business Process*.
- *Supporting Role Context* – Determine what other significant parties will be using the data in the messages. Determine their role in the overall process.
- *System Capabilities Context* – Determine any major restrictions derived from system, a class of systems or standard in the business situation. Identify the type of system.

The registry will provide a list of pre-defined *Business Information Entities* that are available to the selected *Business Process*, and which meet the *Context* criteria specified. These will come with identified relationships to the *Core Components* upon which they are based, and the *Context* rules/values that fully qualify them. The registry should also return partial matches with an indication of how closely they match the specified *Context*.

Step 3: Register re-use of the selected *Business Process* in the set of *Context*s in which it is being used. Registration of each re-use ensures the gradual development of a library of re-uses that will be available to the widening user base.

Step 4: Review the available *Business Information Entities* and select the appropriate subset that meets the needs of the *Business Process* requirement that is being developed.

Step 4a: If the *Business Information Entities* available for the specific *Business Process* do not address all of the data requirements, the registry of all *Business Information Entities* should be searched to see if the appropriate *Business Information Entities* already exist. The procedure for this is described under Search Registry/Repository (Section 5.2), which includes the steps to raise any new *Business Information Entities*, required because no appropriate *Business Information Entities* can be found.

Step 4b: If all required *Business Information Entities* are already available, review the available MIG, XML schema, and/or other syntax-specific message description and select the appropriate one(s) for use that meet the technical implementation/solution requirements identified. If no appropriate technical implementation/solution is already available, continue with Step 5 to create new ones.

Step 5: Create MIG, XML schema, etc. – The resulting semantic model (the set of *Business Information Entities*) is manually or programmatically rendered into a syntax-specific message description. The resulting MIG, XML schema or other syntax specific message description is submitted to the registry where it is associated with the *Business Information Entities* it represents.

[Note]

When selecting a *Business Process* and defining the required messages, searches may be made against potential trading partners' data requirements and processes. The *Context Rules* and *Business Information Entities* represent useful metadata in determining the best possible match between the user and their partners. The fact that the rules can be made available in processable formats means that the comparison itself could be automated and made available as a feature of the repository implementation.

5.2 Core Components Discovery

The steps in *Core Component* discovery are preparation and search for candidate common information building blocks. In order to properly define the UN/CEFACT *Core Component Library*, domain or project groups must follow the prescribed preparation and search steps as outlined in the following subsections. See the *Core Components Primer* supplemental document for a detailed end-to-end example of discovering *Core Components*.

5.2.1 Core Component Discovery – Preparation Steps

These steps identify pieces of business information such as *Aggregate Business Information Entities* and their properties. An analysis of *Business Information Entities* from a variety of similar *Business Processes* leads to the underlying core structures and semantics of the *Core Components*. Figure 5-2 graphically portrays the prescribed preparation steps that are described below.

Step 1. Select a *Business Process* that provides a wide range of business information content within the domain being addressed. The broader the range of the chosen *Business Process*, the greater the opportunity to discover candidate *Core Components*. (e.g. **`Make a Payment, Place an Order, Issue an Invoice`**)

Step 2. Focus on each known data exchange within the *Business Process* that contains key business information (e.g. **`Payment Order, Purchase Order, Invoice`**).

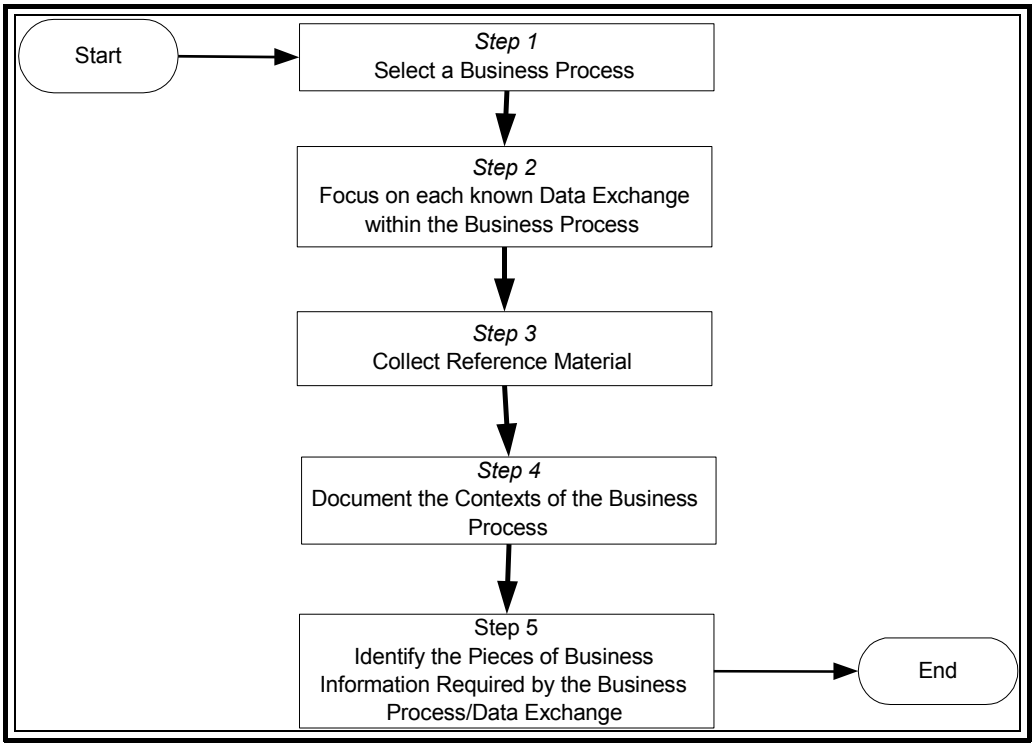
Step 3. Collect all the business information and associated details relevant to the chosen business exchange for the previously identified *Business Process*. Use a cross section of Message Implementation Guides, RosettaNet Partner Interface Process (PIP), Business Process Information Models (BPIMs) or similar domain-specific artefacts as sources of information about the business exchange.

Step 4. Document the *Context*(s) of the *Business Process* being analysed. Identify what is applicable for each category of *Context*, i.e. whether it is none, in all *Context*s, or one or multiple specific *Context* value(s). (See Section 5.6 for a more detailed explanation of how to determine *Context*).

Step 5. Compile a list of the pieces of information required for the *Business Process*.

- If starting from a model (UN/CEFACT recommends UMM models of *Business Processes*), identify the objects (*Aggregate Business Information Entities*) that are needed.
- If not starting from a model, collect the pieces of information into object-like groups (*Aggregate Business Information Entities*). It is important to recognise and avoid pieces of information that are purely used for legacy system or syntax purposes.
- For each *Aggregate Business Information Entity*, capture its unique semantic definition, any *Business Terms* by which it is commonly known and any other information identified in the previous steps.
- At this point of preparation, and before searching the registry/repository, these are candidate *Aggregate Business Information Entities*.

Figure 5-2 Preparation Steps



5.2.2 Core Component Discovery – Search Registry/Repository

Having identified the need for a number of candidate *Aggregate Business Information Entities* in the preparation Step 5 identified in Section 5.2.1 above, repeat the following steps for each *Aggregate Business Information Entity,* as shown in Figure 5-3.

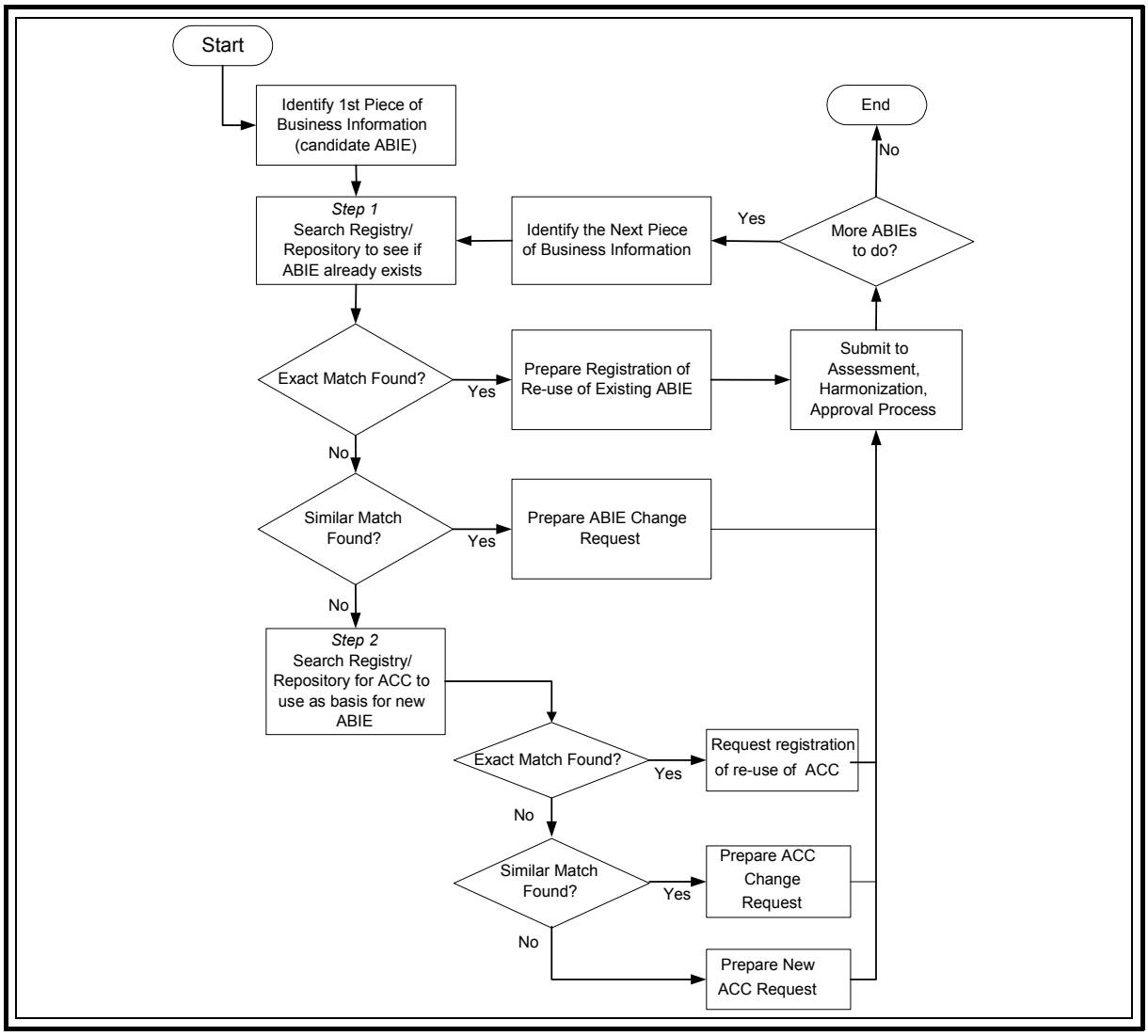
Step 1 It is recommended to start with *Aggregate Business Information Entities* at the highest level of aggregation. Search the *Catalogue of Business Information Entities* for an existing *Aggregate Business Information Entity* with the same definition.

- Exact Match: If there is an *Aggregate Business Information Entity* with a definition and composition that meets the business need, register the re-use including *Business Context* and any *Business Terms*. (Go to next *Aggregate Business Information Entity*)

- Similar Match: If there is an *Aggregate Business Information Entity* with a definition that potentially could be modified to meet the business need, prepare an *Aggregate Business Information Entity* change request for submission to the harmonization and approval process. Proposed changes have to be assessed to ensure that any adaptation is sensible, reasonable and applied in the most appropriate way. This, together with registration of re-uses, will ensure the availability of a real and usable pool of material to a widening user base. Include re-use, *Business Context* and any *Business Terms*. (Go to next *Aggregate Business Information Entity*)

- If there is not an *Aggregate Business Information Entity* with a suitable definition, go to Step 2.

Figure 5-3 Search Steps



[Note]

Exact is 'a precise match in all details'.

Similar is 'of the same kind without being identical'.

Employment of common sense and good judgement is essential in making these determinations.

Step 2 Search the *Catalogue of Core Components* for an existing *Aggregate Core Component* that has the appropriate generic definition and structure from which the new required *Aggregate Business Information Entity* can be formed.

- If there is an existing *Aggregate Core Component* with a definition and structure that meets the business needs, register the re-use of the *Aggregate Core Component* as an *Aggregate Business Information Entity* including the definition and name created according to the *Naming Convention*. (Go to next *Aggregate Business Information Entity*)

- If there is an *Aggregate Core Component* with a definition and structure that potentially could be modified to meet the business need, prepare an *Aggregate Core Component* change request for submission to the harmonization and approval process. Include the re-use of the *Aggregate Core Component* as an *Aggregate Business Information Entity*, including the definition and name created according to the *Naming Convention* (See Section 6.1.4), and the *Business Context* in which it is used. (Go to next *Aggregate Business Information Entity*)

- If there is not an *Aggregate Core Component* with a suitable definition and structure, prepare a new *Aggregate Core Component* request for submission to the harmonization and approval process. Include the re-use of the *Aggregate Core Component* as an *Aggregate Business Information Entity*, including the definition and name created according to the *Naming Convention*, and the *Business Context* in which it is used. (Go to next *Aggregate Business Information Entity*)

5.2.3 Core Component Discovery – Basic and Association Business Information Entities

This procedure is exactly the same as that described in Section 5.2.2, except that the reader should read *Basic* or *Association Business Information Entity* for *Aggregate Business Information Entity* and *Basic* or *Association Core Component* for *Aggregate Core Component*.

5.2.4 Data Types, Property, and Identifying Similarities

When looking for similarities between existing *Business Information Entities* and *Core Components*, and those *Business Information Entities* that are required but not present, the user should consider *Property* and *Data Types*. If a *Core Component* is found that has a very similar *Property* to an existing *Core Component*, but a different *Object Class*, then that *Property* should be used for the new *Core Component* that is to be created where the basic structure and semantics are similar. The key to the similarities of *Property* is that they share a *Data Type*. If a new *Core Component* is requested, these identified similarities at the level of *Property* should also be identified.

[Example]

There is an existing *Basic Business Information Entity* for **`Invoice. Total_ Tax. Amount`**, based on a corresponding *Basic Core Component*. The user needs a *Basic Business Information Entity* for **`Order. Total_ Tax. Amount`**, but after searching the registry/repository determines this does not exist. Because both the existing *Basic Business Information Entity* of **`Invoice. Total_ Tax. Amount`** and the desired *Basic Business Information Entity* of **`Order. Total_ Tax. Amount`** share strong similarities—they are the same *Property* and share a specific *Data Type*, but are applied to different *Object Classes*—the user would identify this similarity, and use it to take the appropriate action in the discovery process.

5.3 Preparation for Submission

Following the search of the *Core Component Library*, there may be a need to prepare submissions for the harmonization and approval process. (See Section 5.4)

- Preparation of submissions will be carried out by the business domain or project group making the discovery.

- Harmonization and approval will be conducted by appropriate Assessment, Harmonization and Approval teams to be set up as part of the UN/CEFACT electronic business standards forum.

The different types of submissions that may be required are detailed below.

The following submissions are simple documented requests, following procedures to be established by the Assessment, Harmonization and Approval teams.

- To request registration for a re-use of an existing *Aggregate Business Information Entity*

- To make a Change Request for an existing *Aggregate Business Information Entity*

- To make a Change Request for an existing *Aggregate Core Component*

The following submissions require more significant preparation, as part of the *Core Component* working methodology, to be carried out by the business domain or project group conducting the discovery and analysis.

- Preparation for Requesting a new *Basic Core Component*

- Preparation for Requesting a new *Association Core Component*

- Preparation for Requesting a new *Aggregate Core Component*

- Preparation for Requesting a new *Basic Business Information Entity*

- Preparation for Requesting a new *Association Business Information Entity*

- Preparation for Requesting a new *Aggregate Business Information Entity*

Each of these needs to initially follow the same steps in applying the *Naming Convention* (Section 6.1.4) to arrive at the name of the new item.

5.3.1 Applying the Naming Convention to a New Item

For all new items, the *Naming Convention* and associated rules defined in Section 6.1.4 must be applied. Figure 5-4 shows the steps that must be taken, each of which is described in the accompanying text.

Step 1. Develop a thorough semantic definition and include any useful business remarks as comments. Semantic definitions should:

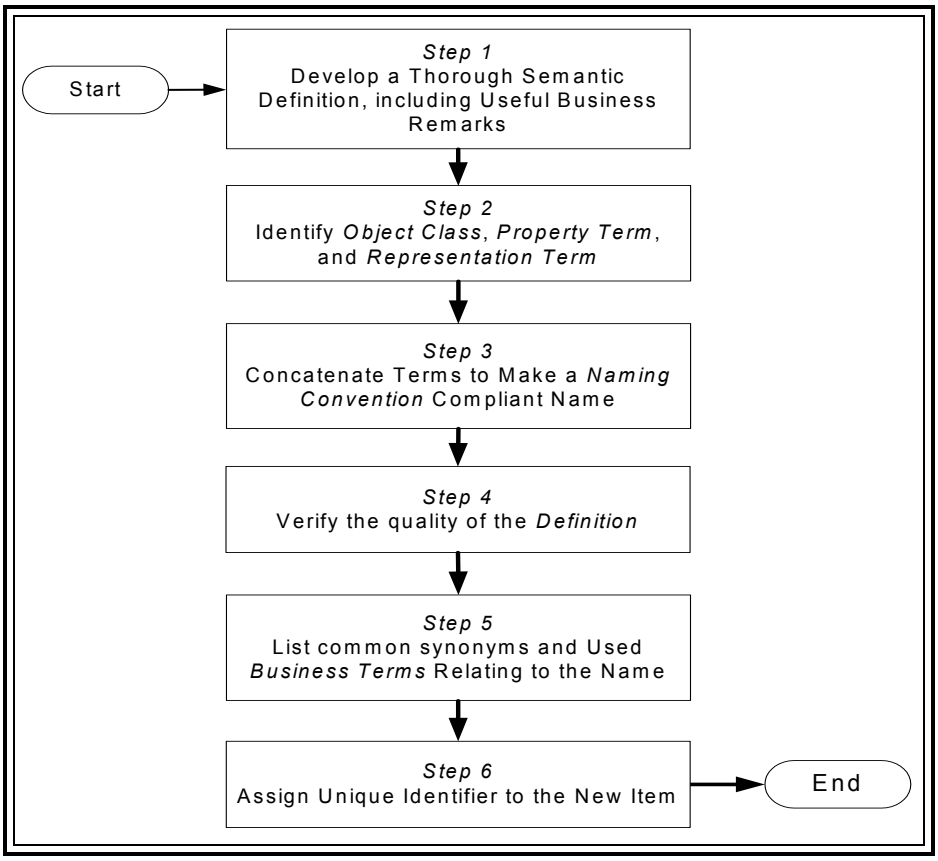
- use words different to those being defined provided that no ambiguity is thereby introduced,

- be globally applicable,

- be generic (i.e. able to cover the same business concept for different products/services),

- be applicable across multiple industries or domains, and
- be simple and clear to enable unambiguous translation to other languages

Figure 5-4 Applying the Naming Convention



Step 2. Follow the *Naming Convention for Core Components or Business Information Entities* (Section 6.1.4) to identify as appropriate:

- *Object Class Term*
- *Property Term*
- *Representation Term*
- *Qualifier Term(s)*

[Note]

When creating names for *Business Information Entities* that have properties identical to those of other existing *Business Information Entities*, the name of the *Property* should be used to validate the correct naming of the new *Business Information Entity*. Consistent naming of similar *Business Information Entities* and *Core Components* contributes to their usability.

Step 3. Concatenate the terms to create a *Naming Convention* compliant *Dictionary Entry Name*.

[Note]

The resultant name may seem artificial in that it might not be the same as any of the *Business Terms* used for that concept. However, rigor of the *Naming Convention* enables future translation of the name into other languages.

Step 4. Verify the quality of the definition by adding the words *"[Dictionary Entry Name]* is" to the front of the definition, where *[Dictionary Entry Name]* is the agreed name.

Step 5. List common synonyms or *Business Term(s)* that are used within the domain to identify the piece of business information (e.g. **`Account Number, Account Identifier`**).

[Note]

Some *Business Terms* are used for several different pieces of business information. It is perfectly acceptable to have the same *Business Term* listed as a synonym for two or more pieces of business information. For example, as shown in Figure 5-5, **`Account Number`** is a synonym for **`Financial Account Identifier`** and for **`Sales Account Identifier`**.

Figure 5-5 Core Component Catalogue Extract

Temp Identifier	Definition	Remarks	Business Terms	CCT	Dictionary Entry Name	Object Class	Property Term	Representation Term
C00010	A Financial is a service through bank or other organisation through which funds are on behalf of a or goods or are supplied on	Not a general ledger.	Account	n/a	Financial Account. Details	Financial Account	Details	
F00012	A Sales Account is relationship a vendor and a customer.	Usually includes a contract specifying the terms of	Account	n/a	Sales Account. Details	Sales Account	Details	

Same Business Term

Step 6. Assign a Temporary Identifier to the new item in the form of a 6 digit alphanumeric string, chosen at the discretion of the user.

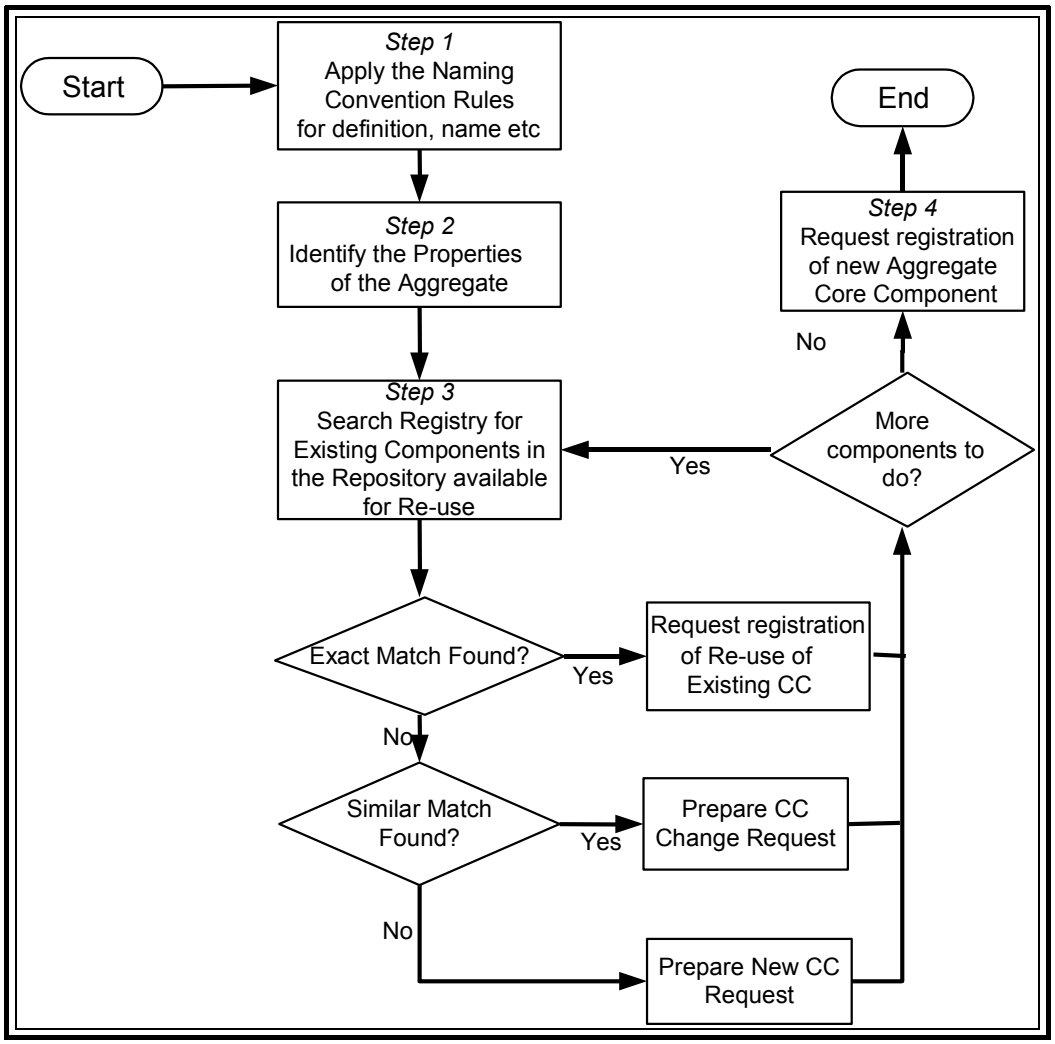
5.3.2 Preparation for Submitting New Items

This section contains illustrative procedures for submitting new items. The following subsections address submitting new *Aggregate Core Components*, new *Basic Core Components*, and new *Aggregate Business Information Entities* that re-use an existing *Aggregate Core Component*. Similar submission procedures will need to be used for submitting *Association Core Components*, *Basic Business Information Entities*, and *Association Business Information Entities*.

5.3.2.1 New Aggregate Core Components

The development of a new *Aggregate Core Component* requires adherence to the *Naming Convention* rules for naming and definition. Once named, the new aggregate's constituent parts need to be individually examined. The following diagram and text describe the procedure to be followed.

Figure 5-6 Preparation for requesting a new Aggregate Core Component



Step 1. Apply the *Naming Convention* rules to arrive at the name of the new *Aggregate Core Component*

Step 2. Identify all of the *Properties* within the new *Aggregate Core Component*.

Repeat the following step for each constituent *Property* identified in Step 2:

Step 3. Search the Registry for an existing *Core Component* or *Data Type* that has the appropriate generic definition and structure.

- If there is an existing *Core Component* or *Data Type* with a definition and structure that meets the requirement, request registration of this re-use of the *Core Component* or *Data Type* including the *Context* in which it is used.
- If there is an existing *Core Component* or *Data Type* with a definition and structure that potentially could be modified to meet the requirement, prepare an appropriate change request for submission to the harmonization and approval process, including the re-use of the *Core Component* or *Data Type* and the *Context* in which it is used.
- If there is not an existing *Core Component* or *Data Type* with a suitable definition and structure, prepare an appropriate new item request, that includes identification of the *Context*, for submission to the harmonization and approval process.

When all the constituent properties identified in Step 2 have been checked as described in Step 3, then:

Step 4. Request registration of the new *Aggregate Core Component*.

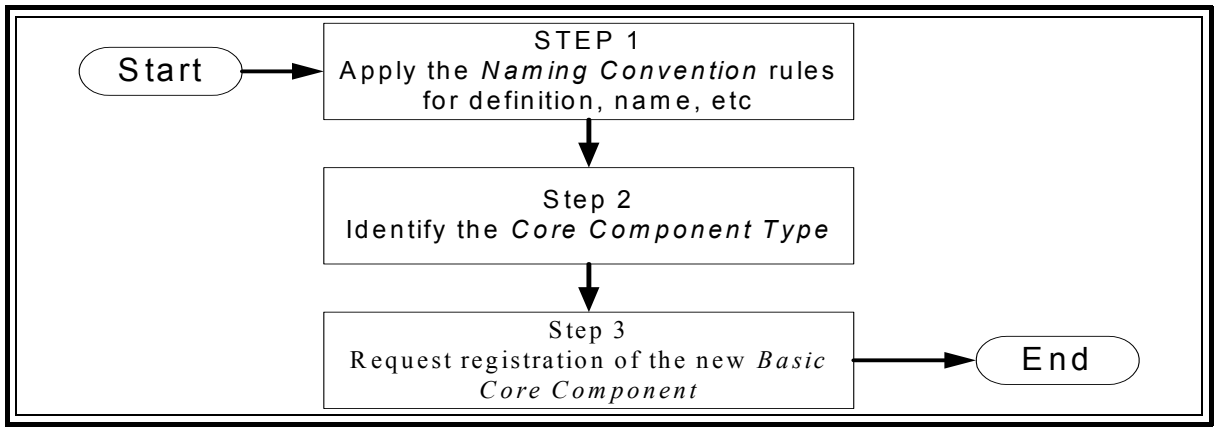
Prepare the new *Aggregate Core Component* request and submit to the harmonization and approval process.

5.3.2.2 New Basic Core Components

As shown in Figure 5-7, there are three steps necessary to prepare for requesting a new *Basic Core Component*. These three steps are:

Step 1. Apply the *Naming Convention Rules* to arrive at the name of the new *Basic Core Component*

Figure 5-7 Preparation Steps for Requesting a New Core Component.



Step 2. Select the appropriate *Core Component Type*. (See Section 6.1.1 for an explanation and listing of *Core Component Types*).

Step 3. Request registration the new *Basic Core Component*

5.3.2.3 New Aggregate Business Information Entities which re-use Existing Aggregate Core Components

As shown in Figure 5-8, there are four steps necessary to prepare for requesting a new *Aggregate Business Information Entity* that re-uses an existing *Aggregate Core Component.* These four steps are:

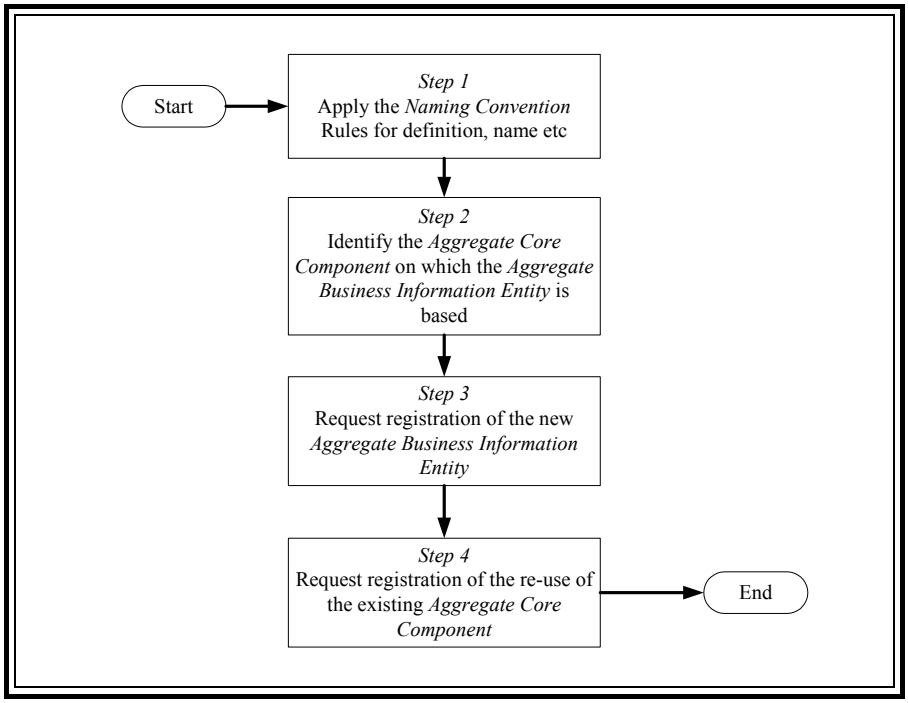
Step 1. Apply the *Naming Convention Rules* to arrive at the name of the new *Aggregate Business Information Entity*.

Step 2. Identify the *Aggregate Core Component* on which the new *Aggregate Business Information Entity* is based

Step 3. Request registration of the new *Aggregate Business Information Entity.*

Step 4. Request registration the re-use of the existing *Aggregate Core Component* by this new *Aggregate Business Information Entity.*

Figure 5-8 Preparation Steps for Requesting a New Aggregate Business Information Entity using Existing Aggregate Core Component



5.4 Harmonization

The purpose of harmonization is to take candidate *Core Components* or *Business Information Entities* submitted by different domains, identify differences and similarities between the submissions and existing library entries, and produce a single, complete cross-domain set, i.e. the *Core Component Library*. Harmonization is a critical process in the overall *Core*

Component procedures. The following describes the recommended areas that harmonization procedures should cover.

- Evaluate each submitted *Core Component* for consistent application of the discovery methodology. Resolve any questions or issues by discussion with the submitting groups.

- Compare the definition and structure of each submitted *Core Component* with what already exists in the *Core Component Library*.

  - If the submitted *Core Component* is the same or similar, compare the properties of each to identify any differences. If the submitted *Core Component* has properties missing in the existing one, enforce a harmonized form that contains the properties of each. If the submitted *Core Component* is a subset of the existing *Core Component* definition, then recommend the use of the existing one. Similarities between *Core Components* should be judged on whether or not the *Property* of each shares a *Data Type*. A *Data Type* should be reused as much as possible across *Properties* of *Core Components*.

  - If the definition of the *Core Component* does not match any existing ones, then proceed with the next step.

- Publish the results of harmonization to the submitting groups for review and finalization.

[Note]

In order to ensure that each submission is evaluated on its own merits, and that no submission is given precedence over others, all submissions should be processed separately and serially against the full cross-domain library.

Once the submitted material has passed the harmonization procedure, it may now be submitted for technical assessment and approval.

5.5 Technical Assessment and Approval

Technical assessment must be done in close coordination with the discovery teams and the harmonization process in order to minimise domain re-working after technical assessment and harmonization review. This section defines a recommended process for conducting technical assessment and approval of all newly submitted and changed *Core Components*. A technical assessment and approval process for *Business Information Entities* should also be developed and applied.

Technical assessment procedures define the processing that shall be followed by the joint development groups, the harmonization group, submission entry points, the technical assessment group, and the secretariat as related to the review of *Core Components*. The result of this process is the final publication of approved *Core Components*.

These procedures are needed in order to facilitate the process of reviewing and approving submissions to the *Core Component Library*. In order to minimise the requirements for technical assessment and harmonization, and to expedite the review and approval process, *Core Component* development groups should work with the technical assessment group, and the harmonization group during the early development stages of component discovery.

In outline, these procedures should cover:

- Submission of *Core Component* work ready to be reviewed to a designated secretariat.
- Recording of all *Core Component* submissions and distribution to the harmonization group members.
- Review procedures and criteria followed by the harmonization group.
- Return of harmonized *Core Component* submissions for technical assessment.
- Review procedures and criteria followed by the technical assessment group.
- Registration of the approved *Core Component*(s) in the appropriate *Core Component* registry.

5.6 Context in the Discovery Process

Information that is needed by a *Business Process* is used in a *Context* defined by how and where the *Business Process* can take place. The initial analysis will be performed on a set of *Business Information Entities*, i.e. *Basic, Association,* and *Aggregate Business Information Entities*, and not on a set of *Core Components* (See Figure 5-1). The analysis that produces *Core Components* is, among other things, a process of identifying the various *Context Categories* and values, to determine the underlying *Context*-independent *Properties*.

The guidelines presented here facilitate the analysis of *Business Information Entities* to determine core *Business Semantics* or provide a mechanism to describe *Business Information Entities* when they are entered into a registry and published in a repository.

If there are any instances of the *Business Information Entity* in which a *Property* is not present, this raises the issue of identity. Specifically – is the *Business Information Entity* which lacks that *Property* really the same *Business Information Entity*, just used in a different *Context*?

If the answer to this question is *yes*, then that *Property* is part of the *Core Component*. If the answer is *no*, then it is possible that a second, different *Core Component* has been discovered.

5.6.1 Context Categories

Context Categories are introduced here and are followed by a brief description. After which the various guidelines used to determine *Context* are introduced:

- *Business Process Context* – This is the classification of the *Business Process*, business collaboration or business transaction as described in the *UN/CEFACT*

Catalogue of Common Business Processes. It is the primary *Context Category*, and provides many useful distinctions in the analysis of *Core Components*.

- *Product Classification Context* – There are many types of information that are specific to products or services being traded or referred to in a *Business Process*.

- *Industry Classification Context* – Traditionally, business vocabularies are divided into industry verticals. This *Context Category* specifies a particular industry vertical.

- *Geopolitical Context* – Specifies the semantic and structural variation. This is often the result of regional or cultural factors.

- *Official Constraints Context* – Specifies the legal or contractual influences upon *Business Semantics*.

- *Business Process Role Context* – Every partner in a *Business Process* data exchange has a particular role – buyer, seller, etc. These roles are described in the *UN/CEFACT Catalogue of Common Business Processes* and in other *Business Libraries* (libraries of *Business Process* models). Depending on the *Business Process*, the nature of these roles may require that certain semantics and data be employed in the messages exchanged. In any *Business Process Role Context*, one must either be a sender or receiver of data in that particular exchange – otherwise, role is described by the *Supporting Role Context*.

- *Supporting Role Context* – Parties in a *Business Process* who are neither senders nor receivers of data in a particular exchange, may place requirements on the data exchanged by partners who are sending or receiving of data in that exchange. These non-sending, non-receiving parties in this exchange play a supporting role, and are described by the *Supporting Role Context*.

- *System Capabilities Context* – When a particular semantic or structure is primarily the result of system constraints or compliance with a standard, it is attributable to the *System Capabilities Context*.

5.6.2 Guidelines for Analysing Business Information Entities in Context

Using the criteria given in section 5.6.1 for determining that a particular *Property* of a *Business Information Entity* is the product of its use in *Context*, the analyst must ascertain and document the applicable *Context Categories*. To accomplish this, the analyst should list all the *Context Categories*, and assign a value or values to each category for that component. If a *Context* category has no particular value or values, the analyst should assign a value of *In All Contexts* (for all *Contexts* except *Official Constraints*) or *None* (for *Official Constraints*). As this analysis is conducted, different *Context Categories* might appear to be in competition for application. The analyst must ascertain which *Context Category* is responsible. This section provides some guidelines for answering this question in a systematic and consistent fashion, by examining the typical ambiguities that arise.

It is possible that a particular *Property* of a *Business Information Entity* may be the result of several *Context* factors. These *Context* factors are identified by analysis of differences and similarities across particular *Context*s. For example, comparing the same *Business Information*

Entity as used in different regions of the world, variation will probably be the result of a *Geopolitical Context* or *Official Constraints Context* (see below). If a single *Business Information Entity* differs between *Business Processes*, then the *Business Process Context* is probably the cause.

The following guidelines apply:

1) *Geopolitical Context* versus *Official Constraints Context*

If a *Property* can be traced to a specific body of law or international treaty then it is the result of an *Official Constraint*. For example, if a warning about hazardous goods is required as part of a goods description, and it is required on all uses of that goods description within the United States, then both *Geopolitical* and *Official Constraints* are involved. The value of an *Official Constraint Context* should always be the body of law or treaty that is being cited. The value of a *Geopolitical Context* always expresses the region or regions that are relevant.

2) *Product Classification Context* versus *Industry Classification Context*

When a particular variation on a given product or service is specific to a particular industry, then the *Industry Classification Context* is adequate to specify the *Context*. If all examples of the particular product or service are described by the same unique set of *Properties* across industries, then only a *Product Classification Context* is required. In other cases, a value or values should be supplied for both *Context Categories*.

3) *Business Process Context versus Business Process Role Context*

Business Process Role Context is employed when one actor in the *Business Process* has an information requirement and the other does not. If both actors have the same information requirement, then it is a *Business Process Context*.

4) *System Capability Context Categories*

This *Context* is the result of system or classes of systems that *primarily* influence data variation. For example, if a specific Enterprise Resource Planning (ERP) provider's proprietary data formats use a particular field, and no other applications use that field, then the presence of the data can be attributed to the processing capabilities of that specific system.

The following detailed example illustrates the process of assigning values for all *Context Categories* as part of the *Business Information Entity* analysis process:

[Example]

Case: A **`buyer address`** *Business Information Entity* is taken from a standard that is used across all industry boundaries and in all processes within the United States. The *Business Information Entity* also contains a *Property* that holds the **`State`** information.

The following set of values could be ascribed to this *Property* for this *Business Information Entity*:

Business Process = **`In All Contexts`**

Product Classification = **`In All Context`**

Industry Classification = **`In All Contexts`**

[Example] (Continued)

Geopolitical = **United States**

Official Constraint = **None**

Business Process Role = **In All Contexts**

Supporting Role = **In All Contexts**

System Capabilities = **In All Contexts**

These values were selected based on the following analysis:

The *Business Information Entity* construct is the same in every *Business Process* covered by the standard in question – the address always contains a **State** field. Therefore, for the range of *Business Processes* covered by the *Business Information Entity* being analysed, – the *Business Process Context* category is marked **In All Contexts**.

The products that might be described in the same business message do not affect the address. Since the standard from which the *Business Information Entity* has been extracted is horizontal across industry boundaries, it is equally valid in all *Industry Classification Contexts*.

As a *Property* of **Buyer Address**, it is clear that the **State** *Property* is intended to hold a value specific to United States geopolitical demarcations. Therefore the *Geopolitical Context Category* is properly assigned the value **United States**.

No specific law can be cited that requires the presence of the **State** *Property* in the address. Therefore, a value of *None* is given to the *Official Constraint Context Category*.

On inspection of *Business Process Role*, it appears that all addresses in the standard in question are required to provide the **State** information, regardless of what role they play in the transaction. The fact that a **Buyer** *Business Process Role* is being analysed has no effect on this *Property*: all types of addresses have the same semantics. Therefore, all roles provide the data equally when giving an address. A value of **In All Contexts** is applicable here. The same reasoning holds for the *Supporting Role Context*.

Finally, considering the *System Capabilities Context*, there are no specific systems that act as the primary reason for the presence or absence of the semantic. Instead, the primary existence of the *Property* can be ascribed to the fact that in common usage, US addresses include the **State** *Property*. Therefore, we can provide the value **In All Contexts** here. Note that as wide of a range of values as possible should be provided to ensure completeness.

If, in the above example, the address was taken from a French standard, it might be that some *Properties* are common across a number of countries in the same region, and perhaps even in multiple regions. Providing the value **France** as a *Geopolitical Context* here would be incomplete – every known valid value should be given.

6 Technical Details

This section provides a detailed technical explanation of the *Core Component*, *Business Process* integration, storage and metamodel elements of the UN/CEFACT *Core Components* concept.

The *Core Component* framework prescribes the mechanism for discovery, normalisation, *Context* specialisation, and structure of *UMM InformationEntities*. The *Aggregate Business Information Entity-Basic Business Information Entity* framework provides the structure for components of the body of the business document. The *Core Component-Business Information Entity-Context* mapping framework provides the basis for mapping *Information Entity* realizations to business entities. The *Business Information Entity* to *Core Component* relationship provides the dictionary reference as specified in the information model abstract syntax. The *Core Component Library* is an implementation of the UMM dictionary concept. The *Basic Core Component* is the realization of a *UMM InformationEntity* and provides the mapping to *Data Types*.

6.1 Core Components, Data Types and Business Information Entities

This section defines the following:

- *Core Component* rules,
- *Data Type* rules,
- *Business Information Entity* rules,
- *Naming Conventions*,
- *Core Component Types,*
- *Content* and *Supplementary Components*, and
- *Representation Terms*.

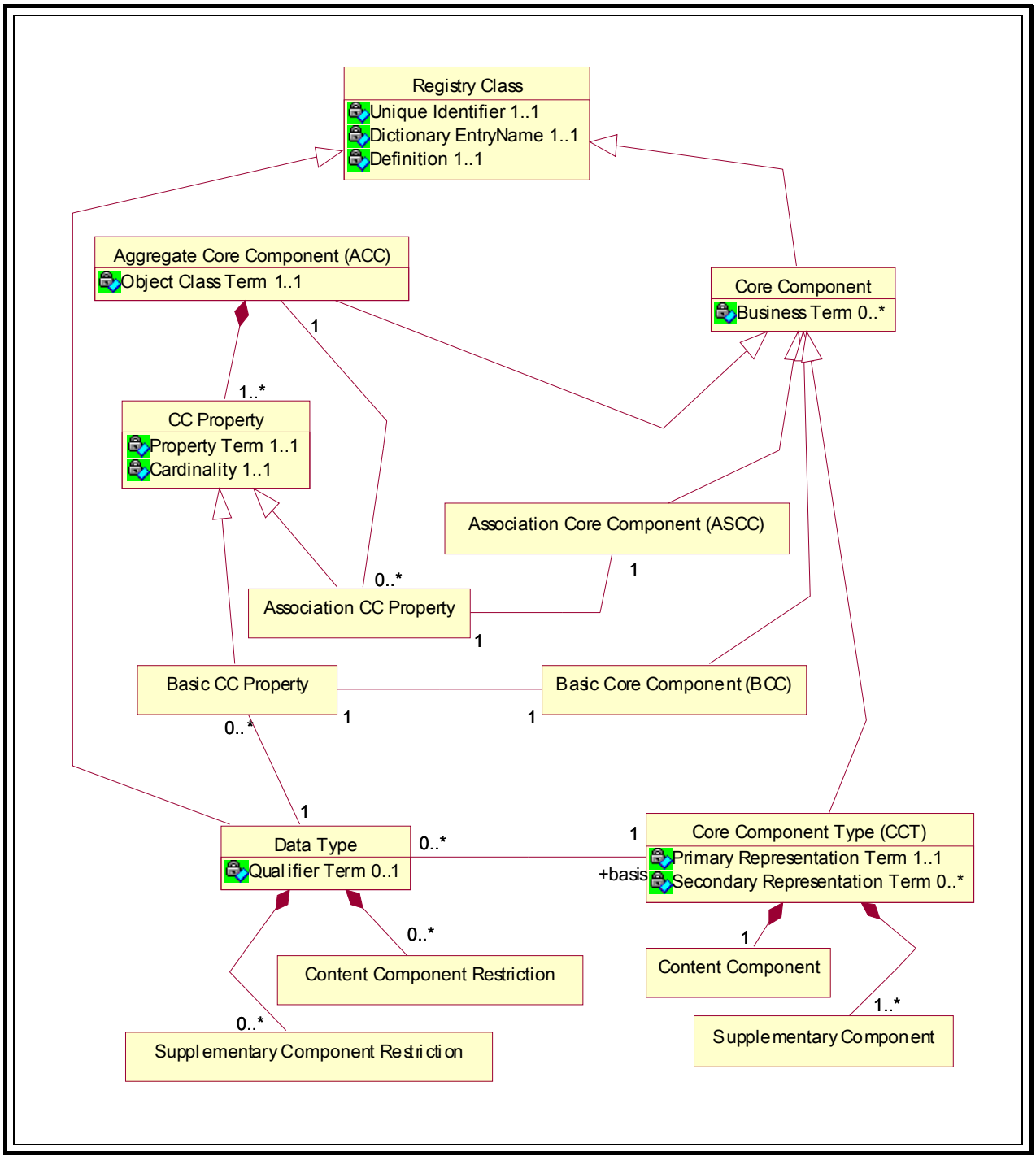
This section also specifies relationships between *Core Components*, *Data Types* and *Business Information Entities* and includes details required for constructing the *Core Components Catalogue* and a larger *Core Component Library*.

6.1.1 Core Components

A *Core Component* is a building block for the development of a semantically correct and meaningful business information exchange 'parcel' containing the information pieces needed to describe a specific concept. There are four categories of *Core Components*: *Basic Core Component, Association Core Component, Core Component Type* and *Aggregate Core Component*. Figure 6-1 illustrates these four categories and their

relationships. The complete *Core Component* Metamodel is illustrated in Figure 7-1. Models are normative to the level of detail at which they exist.

Figure 6-1. Core Components and Data Types Metamodel



The following general rules must be followed in discovering and documenting the four types of *Core Components*:

[C1] Each *Core Component Type*, *Basic Core Component*, *Association Core Component* or *Aggregate Core Component* must have its own unique semantic definition within the library of which it is a part. The definition shall be developed

first and the *Dictionary Entry Name* shall be extracted from it. Comments can be used to further clarify the definition, to provide examples and/or to reference a recognized standard.

[Note]

The *Core Components Dictionary* is one of several ways that *Core Components* are to be made available. The purpose of the *Core Components Dictionary* is to provide a ready reference of the *Core Component* through its *Dictionary Entry Name*, component parts, and definition. The *Core Components Dictionary* will be considered a supplement to the *Catalogue of Core Components* which in turn is a documented listing of the contents of the *Core Components* Registry/Repository.

[C2] Within an *Aggregate Core Component*, all embedded *Core Component Properties* shall be related to the concept of the aggregate.

[C3] There shall be no semantic overlap between the *Core Component Properties* embedded within the same *Aggregate Core Component*.

[C4] The representation of the information in a *Core Component* whose *Core Component Type* is **`Code. Type`** should use a standard issued by a recognized standards body, whenever a standard exists. If international standards are not used a business driven justification shall be provided.

[C5] An *Aggregate Core Component* shall contain at least one *Core Component Property*. A *Core Component Property* shall be either a *Basic Core Component Property* or an *Association Core Component Property*.

[Note]

At the deepest level of nesting an *Aggregate Core Component* shall only contain *Basic Core Component Properties*.

[Note]

For the purpose of exchanging information a practical compromise on the level of detail of a *Basic Core Component* is required. This compromise shall be based on the business need. It is not necessary to have absolute detail, which decomposes a piece of information down to its lowest level.

[C6] An *Aggregate Core Component* shall never contain – indirectly or at any nested level – a mandatory *Association Core Component Property* that references itself.

[Note]

The objective of the above rule is to avoid endless loops in the definition of an *Aggregate Core Component*. The rule allows an *Aggregate Core Component* to contain an *Association Core Component Property* that references itself. The fact that the *Association Core Component Property* is not mandatory makes it possible to stop the loop after a finite number of iterations.

[C7] The *Core Component Type* shall be one of the approved *Core Component Types*

Table 8-1 provides a complete list of the approved *Core Component Types* as of the date of this specification.

[Note]

Table 8-1 may subsequently be published separately to facilitate maintenance outside the body of this specification.

Table 8-2 provides a complete list of the approved *Content Components* and *Supplementary Components* as of the date of this specification.

[C8] The *Content Component* shall be the approved *Content Component* for the related *Core Component Type*

[C9] The *Supplementary Component* shall be one of the approved *Supplementary Components* for the related *Core Component Type*

[Note]

Table 8-2 may subsequently be published separately to facilitate maintenance outside the body of this specification.

6.1.2 Data Types

A *Data Type* defines the set of valid values that can be used for a particular *Basic Core Component Property* or *Basic Business Information Entity Property*. It is defined by specifying restrictions on the *Core Component Type* from which the *Data Type* is derived. Figure 6-1 describes the *Data Type* and shows relationships to the *Core Component Type*.

[D1] A *Data Type* shall be based on one of the approved *Core Component Types*.

[D2] Where necessary, a *Data Type* shall restrict the set of valid values allowed by the *Core Component Type* on which it is based, by imposing restrictions on the *Content Component* and/or the *Supplementary Component*.

6.1.3 Business Information Entities

A *Business Information Entity* is a piece of business data or a group of pieces of business data with a unique *Business Semantic* definition in a specific *Business Context*. A *Business Information Entity* can be a *Basic Business Information Entity* (BBIE), an *Association Business Information Entity* (ASBIE) or an *Aggregate Business Information Entity* (ABIE).

- A *Basic Business Information Entity* is based on a *Basic Core Component* (BCC).
- An *Association Business Information Entity* is based on an *Association Core Component* (ASCC).

- An *Aggregate Business Information Entity* is a re-use of an *Aggregate Core Component* (ACC) in a specified *Business Context*.

Figure 6-2. Business Information Entities Basic Definition Model

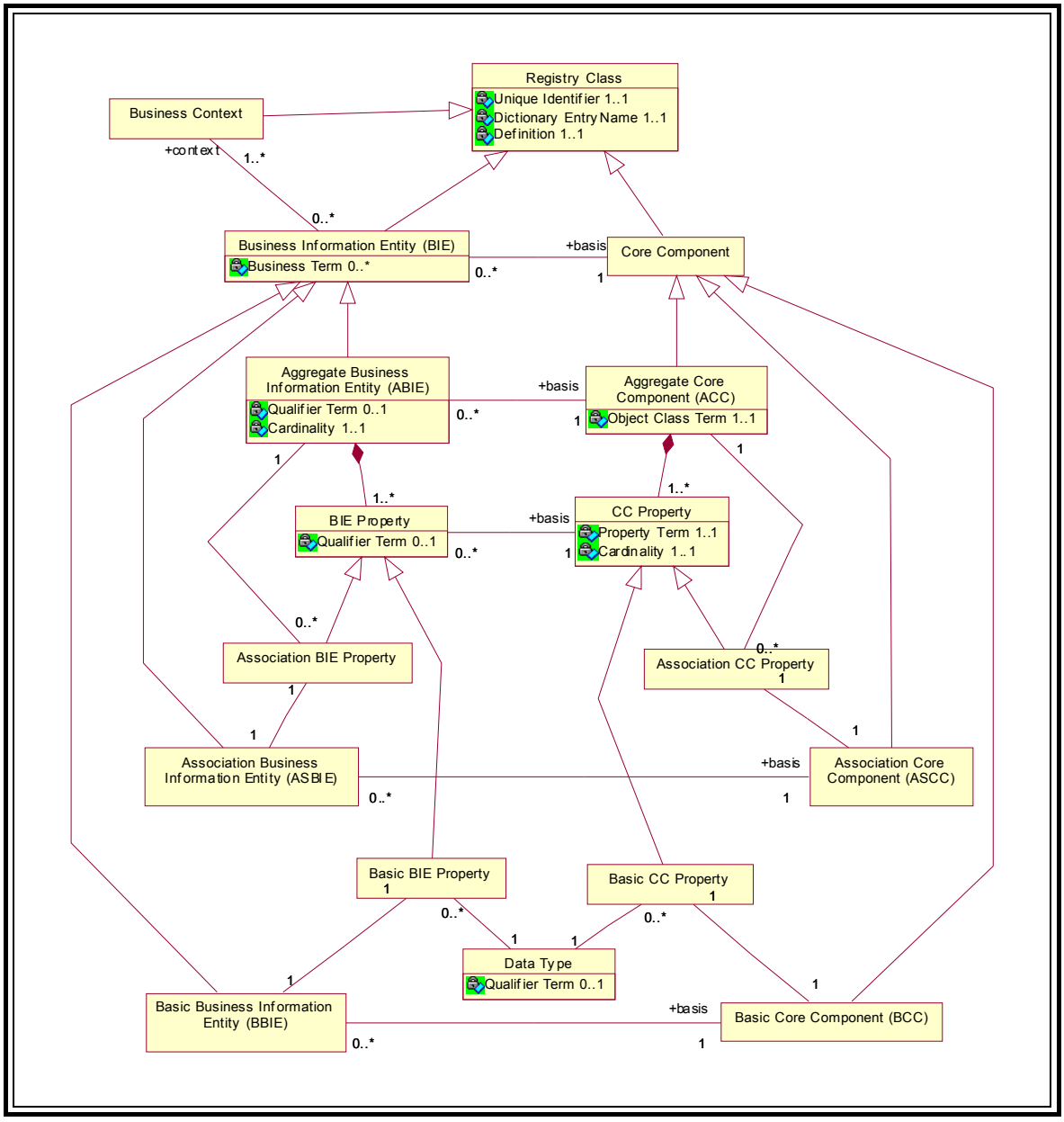


Figure 6-2 describes the *Business Information Entity* types and shows relationships to the *Core Component* counterparts.

[B1] A *Business Information Entity* shall be a *Basic Business Information Entity*, an *Association Business Information Entity* or an *Aggregate Business Information Entity*

[B2] A *Business Information Entity* shall be defined by one or more *Business Contexts*

[B3] A *Basic Business Information Entity* shall be based on a *Basic Core Component*

[B4] An *Association Business Information Entity* shall be based on an *Association Core Component*

[B5] An *Aggregate Business Information Entity* shall be based on an *Aggregate Core Component*

[B6] An *Aggregate Business Information Entity* shall contain at least one *Business Information Entity Property*. A *Business Information Entity Property* shall either be a *Basic Business Information Entity Property* or an *Association Business Information Entity Property*.

[Note]

At the deepest nesting level an *Aggregate Business Information Entity* will only contain *Basic Business Information Entity Properties*.

[B7] A *Business Information Entity Property* of an *Aggregate Business Information Entity* shall be based on a *Core Component Property* of the corresponding *Aggregate Core Component*.

[B8] The *Data Type*, on which a *Basic Business Information Entity Property* is based, shall itself be similar to the *Data Type* on which the corresponding *Basic Core Component Property* is based (i.e. it shall either be the same *Data Type* or a more restricted one).

[B9] The *Aggregate Business Information Entity*, on which an *Association Business Information Entity Property* is based, shall itself be based on the *Aggregate Core Component* on which the corresponding *Association Core Component Property* is based.

[B10] An *Aggregate Business Information Entity* shall never contain – directly or at any nested level – a mandatory *Association Business Information Entity Property* that references itself.

[Note]

The objective of the above rule is to avoid endless loops in the definition of an *Aggregate Business Information Entity*. The rule allows an *Aggregate Business Information Entity* to contain an *Association Business Information Entity Property* that references itself. The fact that the *Association Business Information Entity Property* is not mandatory makes it possible to stop the loop after a finite number of iterations.

6.1.4 Naming Convention

A *Naming Convention* is necessary to gain consistency in the naming and defining of all *Core Components*, *Data Types* and *Business Information Entities*. The resulting consistency facilitates comparison during the discovery and analysis process, and precludes ambiguity, such as the development of multiple *Core Components* with different names that have the same semantic meaning.

The *Naming Convention* is derived from the guidelines and principles described in document ISO 11179 Part 5 -- *Naming and Identification Principles For Data Elements*. In certain instances, these guidelines have been adapted to the *Core Component* environment. In particular, the guidelines have been extended to cover the naming and defining of *Core Component Types*, *Data Types* and *Business Information Entities*.

The official language for UN/CEFACT *Core Components* is English. All official dictionary entries will be in English. *Core Component* discovery work may very well occur in other languages, however official submissions for inclusion in the UN/CEFACT library must be in English. In order to ensure absolute clarity and understanding of the names and definitions it is essential to use words from the *Oxford English Dictionary*. A supplementary *Controlled Vocabulary* will be developed to identify the definition to be used for any words that are potentially ambiguous. This *Controlled Vocabulary* shall also be used to identify the preferred word in cases where more than one word might be used to cover the same definition. The *Controlled Vocabulary* will also contain terms not found in the *Oxford English Dictionary*. This will ensure that each word within any of the names and definitions is used in a consistent and unambiguous way. The resultant semantic integrity will also mean that translation into other languages retains the precise original meaning.

6.1.4.1 Core Component Naming Rules

The following subsections contain all *Core Component* naming rules.

6.1.4.1.1 Core Component Dictionary Information

Each *Core Component* contains the following dictionary information that is impacted by the naming rules in subsequent sub-sections:

- ***Dictionary Entry Name*** (Mandatory). This is the unique official name of the *Core Component* in the dictionary.
- ***Definition*** (Mandatory). This is the unique *Business Semantic* of that *Core Component*.
- ***Business Term*** (Optional). This is a synonym term under which the *Core Component* is commonly known and used in the business. A *Core Component* may have several *Business Terms* or synonyms.

[Example]

Dictionary Entry Name – **`Person. Tax. Identifier`**

Definition – The registered national tax identification of a person

Business Term – **`Income tax number`**, **`national register number`**, **`personal tax register number`**, **`social security number`**, **`national insurance number`**

The naming rules are also based on the following concepts as defined in ISO 11179:

- ***Object Class***. This represents the logical data grouping or aggregation (in a logical data model) to which a *Property* belongs. The *Object Class* is expressed by an *Object Class Term*. The *Object Class* thus is the part of a *Dictionary Entry Name* of the *Core Component* that represents an activity or object in a specific *Context*. *Object Classes* have explicit boundaries and meaning and their *Properties* and behaviour follow the same rules.

- ***Property Term***. This represents the distinguishing characteristic or *Property* of the *Object Class* and shall occur naturally in the definition.

- ***Representation Term***. An element of the *Core Component* name which describes the form in which the *Core Component* is represented.

6.1.4.1.2 Core Component General Rules

[C10] The dictionary content, with the exception of *Business Terms*, shall be in the *English Language* following the primary *Oxford English Dictionary* English spellings to assure unambiguous spelling.

> [Note]
>
> There may be restrictions in specific languages, which need to be applied when translating the *Core Component Dictionary* into other languages. These restrictions shall be formulated as additional rules and added as separate language specific annexes to this document.

6.1.4.1.3 Core Component Rules for Definitions

[C11] The definition shall be consistent with the requirements of ISO 11179-4 Section 4.4 and will provide an understandable meaning, which should also be translatable to other languages.

[C12] The definition shall take into account the fact that the users of the *Core Component Dictionary* are not necessarily native English speakers. It shall therefore contain short sentences, using normal words. Wherever synonym terms are possible, the definition shall use the preferred term as identified in the *Controlled Vocabulary*.

[C13] The definition of a *Basic Core Component* shall use a structure that is based on the existence of the *Object Class Term*, the *Property Term*, and the *Data Type* of the corresponding *Basic Core Component Property*.

[C14] The definition of an *Association Core Component* shall use a structure that is based on the existence of the *Object Class Term*, the *Property Term* and the *Object Class Term* of the *Aggregate Core Component* on which the corresponding *Association Core Component Property* is based.

[C15] Whenever both the definite (i.e. **the***)* and indefinite article (i.e. **a**) are possible in a definition, preference shall be given to an indefinite article (i.e. **a**).

[Note]

To verify the quality of the definition, place the *Dictionary Entry Name* followed by *is* before the definition to ensure that it is not simply a repetition of the *Dictionary Entry Name.*

6.1.4.1.4 Core Component Rules for Dictionary Entry Names

[C16] The *Dictionary Entry Name* shall be unique.

[C17] The *Dictionary Entry Name* shall be extracted from the *Core Component* definition.

[C18] The *Dictionary Entry Name* shall be concise and shall not contain consecutive redundant words.

[C19] The *Dictionary Entry Name* and all its components shall be in singular form unless the concept itself is plural.

[Example]

The singular **`Good`** does not exist, whereas **`Goods`** is a plural noun whose concept involves one or multiple (plural) items

[C20] The *Dictionary Entry Name* shall not use non-letter characters unless required by language rules.

[C21] The *Dictionary Entry Name* shall only contain verbs, nouns and adjectives (i.e. no words like *and, of, the*, etc.). This rule shall be applied to the English language, and may be applied to other languages as appropriate.

[C22] Abbreviations and acronyms that are part of the *Dictionary Entry Name* shall be expanded or explained in the definition.

[C23] The *Dictionary Entry Name* of a *Basic Core Component* shall consist of the following parts in the order specified:

- the *Object Class Term* of the *Aggregate Core Component* owning the corresponding *Basic Core Component Property*,
- the *Property Term* of the corresponding *Basic Core Component Property*, and
- the *Representation Term* of the *Data Type* on which the corresponding *Basic Core Component Property* is based.

[Example]

`Tax. Description. Text`

[C24] The *Dictionary Entry Name* of an *Association Core Component* shall consist of the following components in the specified order:

- the *Object Class Term* of the *Aggregate Core Component* owning the corresponding *Association Core Component Property*,

- the *Property Term* of the corresponding *Association Core Component Property*, and
- the *Object Class Term* of the *Aggregate Core Component* on which the corresponding *Association Core Component Property* is based.

[Example]

Person. Residence. Address

[C25] The components of a *Dictionary Entry Name* shall be separated by dots. The space character shall separate words in multi-word *Object Class Terms* and/or multi-word *Property Terms*. Every word shall start with a capital letter. To allow spell checking of the *Directory Entry Names*' words, the dots after *Object Class Terms* and *Property Terms* shall be followed by a space character.

[Note]

The use of CamelCase for *Dictionary Entry Names* has been considered, but has been rejected for following reasons:

- Use of CamelCase will not allow the use of spell checkers
- Strict use of CamelCase makes it impossible to use separators (".") and therefore doesn't allow an unambiguous identification of the composing parts of the *Dictionary Entry Name.*

[C26] The name of an *Object Class* shall always have the same semantic meaning throughout the dictionary and may consist of more than one word.

[C27] The name of a *Property Term* shall occur naturally in the definition and may consist of more than one word. A name of a *Property Term* shall be unique within the *Context* of an *Object Class* but may be reused across different *Object Classes*.

[Example]

Car. Colour. Code and **Shirt. Colour. Code** may both exist.

[C28] For *Basic* and *Association Core Components*, if the *Property Term* uses the same (or equivalent) word or words as the third component of the *Dictionary Entry Name*, the redundant word(s) in the *Property Term* shall be removed from the *Dictionary Entry Name.*

[Note]

This may lead to the case where the complete *Property Term* is removed from the *Dictionary Entry Name*.

[Example]

If the *Object Class* is **Goods**, the *Property Term* is **Delivery Date**, and *Representation Term* is **Date**, the *Dictionary Entry Name* is **Goods. Delivery. Date**; the *Dictionary Entry Name* for an identifier of a party (**Party. Identification. Identifier**) will be truncated to **Party. Identifier**.

[C30] The *Dictionary Entry Name* of a *Core Component Type* shall consist of a *Representation Term* followed by a dot, a space character, and the term *Type*.

[Example]

Amount. Type; **Date Time. Type**

[C31] In the *Dictionary Entry Name* of a *Core Component Type,* the name of the *Representation Term* shall be one of the primary terms specified in the list of permissible *Representation Terms* as included in this specification (See Section 8.3).

[C32] The *Dictionary Entry Name* of an *Aggregate Core Component* shall consist of a meaningful *Object Class Term* followed by a dot, a space character, and the term *Details*. The *Object Class Term* may consist of more than one word.

[Example]

Postal Address. Details; **Party. Details**

6.1.4.1.5 Rules for Core Component Business Terms

Core Component Business Terms are those terms commonly used for day-to-day information exchanges within a given domain. As such, no specific naming rules apply to *Business Terms*. Interoperability of *Business Terms* will be given by linking them to *Core Component* dictionary entries.

6.1.4.2 Rules for Business Information Entities

The following subsections contain the naming rules for *Business Information Entities*.

6.1.4.2.1 Business Information Entity Dictionary Information

Each *Business Information Entity* contains the following dictionary information that is impacted by the naming rules:

- ***Dictionary Entry Name*** (Mandatory). This is the unique official name of the *Business Information Entity* in the dictionary.
- ***Definition*** (Mandatory). This is the unique semantic business meaning of that *Business Information Entity.*

- ***Business Term*** *(*Optional*)*. This is a synonym term under which the *Business Information Entity* is commonly known and used in the business for a specific *Context*. A *Business Information Entity* may have several *Business Terms* or synonyms.

The *Business Information Entity* naming rules are also based on the following concepts as defined in ISO 11179:

- ***Object Class***. This represents the logical data grouping or aggregation (in a logical data model) to which a data element belongs. The *Object Class* is expressed as an *Object Class Term*. The *Object Class* thus is the part of a *Business Information Entity's Dictionary Entry Name* that represents an activity or object in a specific *Context*. *Object Classes* have explicit boundaries and meaning and their *Properties* and behaviour follow the same rules.

- ***Property Term***. This represents the distinguishing characteristic or *Property* of the *Object Class* and shall occur naturally in the definition.

- ***Representation Term.*** An element of the *Business Information Entity* name which describes the form in which the *Business Information Entity* is represented.

- ***Qualifier Term***. A word or words which help define and differentiate a *Business Information Entity* from its associated *Core Component* and other *Business Information Entities*.

6.1.4.2.2 Business Information Entity General Rules

[B11] The dictionary content, with the exception of *Business Terms*, shall be in the English Language following the primary *Oxford English Dictionary* English spellings to ensure unambiguous spelling.

6.1.4.2.3 Business Information Entity Rules for Definitions

[B12] The definition shall be consistent with the requirements of ISO 11179-4 Section 4.4 and will provide an understandable meaning, which should also be translatable to other languages.

[B13] The definition shall take into account the fact that the users of the *Business Information Entity* dictionary are not necessarily native English speakers. It shall therefore contain short sentences, using normal words. Wherever synonym terms are possible, the definition shall use the preferred term as identified in the *Controlled Vocabulary*.

[B14] The definition of a *Basic Business Information Entity* shall use a structure that is based on the existence of the *Object Class Term*, the *Property Term*, and the *Representation Term,* and enhanced by business related *Qualifier Terms*.

[B15] The definition of an *Association Business Information Entity* shall use a structure that is based on the existence of the *Object Class Term*, the *Property Term* and the *Object Class Term* of the *Aggregate Business Information Entity* on which the

corresponding *Association Business Information Entity Property* is based, and enhanced by business related *Qualifier Terms*.

[B16] Whenever both the definite (i.e. **the**) and indefinite article (i.e. **a**) are possible in a definition, preference shall be given to an indefinite article (i.e. **a**).

6.1.4.2.4 Rules for Business Information Entity Dictionary Entry Names

[B17] The *Dictionary Entry Name* shall be unique.

[B18] The *Dictionary Entry Name* shall be extracted from the *Business Information Entity* definition.

[B19] The *Dictionary Entry Name* shall be concise and shall not contain consecutive redundant words.

[B20] The *Dictionary Entry Name* and all its components shall be in singular form unless the concept itself is plural.

[B21] The *Dictionary Entry Name* shall not use non-letter characters unless required by language rules.

[B22] The *Dictionary Entry Name* shall only contain verbs, nouns and adjectives (i.e. no words like **and, of, the,** etc.). This rule shall be applied to the English language, and may be applied to other languages as appropriate.

[B23] Abbreviations and acronyms that are part of the *Dictionary Entry Name* shall be expanded or explained in the definition.

[B24] The *Dictionary Entry Name* of a *Basic Business Information Entity* shall consist of the following components in the specified order:

- the *Object Class Term* of the corresponding *Basic Core Component*, and possibly additional *Qualifier Term(s),*
- the *Property Term* of the corresponding *Basic Core Component*, and possibly additional *Qualifier Term(s)*,
- the *Representation Term* of the *Data Type* on which the corresponding *Basic Business Information Entity Property* is based.

[B25] The *Dictionary Entry Name* of an *Association Business Information Entity* shall consist of the following components in the specified order:

- the *Object Class Term* of the corresponding *Association Core Component*, and possibly additional *Qualifier Term(s),*
- the *Property Term* of the corresponding *Association Core Component*, and possibly additional *Qualifier Term(s)*,
- the *Object Class Term* of the *Association Business Information Entity* on which the corresponding *Association Business Information Entity Property* is based.

[B26] The *Object Class Term, Property Term*, and *Representation Term* components of a *Dictionary Entry Name* shall be separated by dots. The space character shall separate words in multi-word *Object Class Terms* and/or multiword *Property Terms,* including their *Qualifier Terms*. Every word shall start with a capital letter. *Qualifier Terms* shall be separated from their associated *Object Class* or

Property Term by an underscore (_) followed by a space to separate each qualifier. To allow spell checking of the words in the *Dictionary Entry Nam*e, a space character shall follow the dots after *Object Class Term(s)* and *Property Term(s)*.

[B27] *Qualifier Terms* shall precede the associated *Object Class Term* or *Property Term.* The order of qualifiers shall not be used to differentiate *Dictionary Entry Names*.

[Example]

In the *Business Information Entity* entitled **`Cost. Budget Period_ Total. Amount`**, the component **`Budget Period`** is a *Qualifier Term* for the *Property Term* of **`Total`**. This is derived from the *Core Component* of **`Cost. Total. Amount`**.

[B28] The name of a qualified *Object Class* refers to an activity or object within a *Business Context*. It shall be unique throughout the dictionary and may consist of more than one word.

[B29] For *Basic* and *Association Business Information Entities*, if the *Property Term* uses the same (or equivalent) word or words as the third component of the *Dictionary Entry Name*, and the *Property Term* is not qualified, the redundant word(s) in the *Property Term* shall be removed from the *Dictionary Entry Name*.

[B30] The *Dictionary Entry Name* of an *Aggregate Business Information Entity* shall consist of the name of the *Object Class* of its associated *Aggregate Core Component* and possibly additional *Qualifier Term(s)* to represent its specific *Business Context*, followed by a dot, a space character, and the term *Details*.

6.1.4.2.5 Rules for Business Information Entity Business Terms

Business Information Entity Business Terms are those terms that are commonly used for day-to-day information exchanges within a given domain. As such, no specific naming rules apply to *Business Terms*. Interoperability of *Business Terms* will be given by linking them to the formalised names of the corresponding *Business Information Entity* dictionary entries.

6.1.4.3 Rules for Data Types

6.1.4.3.1 Data Type Dictionary Information

Each *Data Type* contains the following dictionary information that is impacted by the naming rules:

- ***Dictionary Entry Name*** (Mandatory). This is the unique official name of the *Data Type* in the dictionary.
- ***Definition*** (Mandatory). This is the unique *Business Semantic* of that *Data Type.*

The *Data Type* naming rules are also based on the following concepts as defined in ISO 11179:

- ***Representation Term***. This defines the type of valid values for an *Information Entity*.

- ***Qualifier Term***. A word or words which help define and differentiate a *Data Type* from its associated *Core Component Type* and other *Data Types*.

6.1.4.3.2 Data Type General Rules

[D3] The dictionary content shall be in the English Language following the primary *Oxford English Dictionary* English spellings to assure unambiguous spelling.

6.1.4.3.3 Data Type Rules for Definitions

[D4] The definition shall be consistent with the requirements of ISO 11179-4 Section 4.4 and shall provide an understandable meaning, which should also be translatable to other languages.

[D5] The definition shall take into account the fact that the users of the *Data Type Dictionary* are not necessarily native English speakers. It shall therefore contain short sentences, using normal words. Wherever synonym terms are possible, the definition shall use the preferred term as identified in the *Controlled Vocabulary*.

[D6] The definition of a *Data Type* shall use a structure that is based on the existence of primary and secondary *Representation Terms* of the associated *Core Component Type,* and is enhanced by *Qualifier Terms*.

[D7] Whenever both the definite (i.e. **`the`**) and indefinite article (i.e. **`a`**) are possible in a definition, preference shall be given to an indefinite article (i.e. **`a`**).

6.1.4.3.4 Rules for Data Type Dictionary Entry Names

[D8] The *Dictionary Entry Name* shall be unique.

[D9] The *Dictionary Entry Name* shall be extracted from the *Data Type* definition.

[D10] The *Dictionary Entry Name* shall be concise and shall not contain consecutive redundant words.

[D11] The *Dictionary Entry Name* shall not use non-letter characters unless required by language rules.

[D12] The *Dictionary Entry Name* shall only contain verbs, nouns and adjectives (i.e. no words like **`and`**, **`of`**, **`the`**, etc.). This rule shall be applied to the English Language, and may be applied to other languages as appropriate.

[D13] Abbreviations and acronyms that are part of the *Dictionary Entry Name* shall be expanded or explained in the definition.

[D14] The *Dictionary Entry Name* of a *Data Type* shall consist of a *Representation Term*—preceded by *Qualifier Term(s)* as necessary —followed by a dot, a space character, and the term *Type*. The space character shall separate words in multi-word *Qualifier Terms* and *Representation Terms*. Each *Qualifier Term* shall be followed by an underscore. To allow spell checking of the words in the *Dictionary Entry Name*, a space character shall follow the underscores after *Qualifier Terms*.

[Example]

Country_ Identifier. Type

[D15] In the *Dictionary Entry Name* of a *Data Type*, the name of the *Representation Term* shall be one of the primary or secondary terms specified in the *List of Permissible Representation Terms* as included in this specification (See Section 8.3).

[Note]

Whereas the name of the *Core Component Type* shall only be based on a primary *Representation Term*, the *Representation Term* that is used in the *Dictionary Entry Name* of a *Data Type* can also be a secondary *Representation Term*. This will be the case when the *Data Type* restricts the *Core Component Type* in such a way that it only covers a part of the full semantic meaning of the primary *Representation Term*.

6.1.4.3.5 List of Permissible Representation Terms

The *Representation Term* is the part of a *Core Component* name that describes the form of valid values in which the business information is expressed in a data item. For instance all *Basic Core Components* representing a monetary amount shall be named *[Name]. [Qualifier]_ Amount* where *[Name]* represents a specialisation of the generic amount, *[Qualifier]* specifies a restriction of the possible values and *Amount* is the *Representation Term*. Table 8-3 lists the permissible *Representation Terms.*

[Note]

Table 8-3 may subsequently be published separately to facilitate maintenance outside the body of this specification.

[C33] When a *Representation Term* contains more than one word, and the specific use of the *Representation Term* requires only one word, the other word(s) in the *Representation Term* may be dropped.

[Example]

For the *Core Component* entitled **Product Service Start. DateTime**, the *Representation Term* is **DateTime** and the *Core Component* is defined as a date and/or time on which a product/service starts. The *Representation Term* remains **DateTime**. For the *Core Component* **Payment Card. Expiration. Date**, the *Representation Term* is still **DateTime**, however since the specific use of the *Representation Term* requires only date, the word time is dropped.

6.1.5 Catalogue of Core Components

As originally articulated in the ebXML architecture concept and perpetuated in the developing UN/CEFACT architecture concept, all *Core Components* will be recorded in an ebXML compliant registry and stored in a related repository. However, small and medium enterprise (SME) organisations may not be able to readily access such architecture. As such, it is important that the full range of UN/CEFACT *Core Components* be published in a freely available catalogue. This catalogue must convey the full details of each *Core Component* consistent with how those components are stored as UML objects in the registry/repository. Table 6-1 identifies a proper format for the catalogue and contains representative entries from the existing UN/CEFACT *Core Components Catalogue*.

Table 6-1. Core Component Catalogue Format Example

Temporary Identifier	***Dictionary Entry Name***	**Type of *Core Component - Basic, Association, Aggregate***	**Definition**	**Comments**	***Object Class Term***	***Property Term***	***Type (Data Type or Object Class Term)***	***Business Terms***	***Core Component Properties***
000024	Address. Type. Code	Basic	The type of the address.	For example a business address or a home address. Not the Role of the address.	Address	Type	Code		
000147	Base Charge Price. Quantity	Basic	The base quantity of the charge/price unit amount.	For example, for a charge of $5/day for 10 days, the charge base quantity is 1 day.	Base Charge Price	Quantity	Quantity		
000139	Base. Currency. Identifier	Basic	The currency that is on the 'one unit' side of the rate of exchange.	The base currency amount divided by the currency exchange rate gives the second currency amount.	Base	Currency	Identifier		
000012	Birth. Date	Basic	The date on which a person was born.	Applies only to parties being natural persons.	Birth	Date	DateTime		

[Note]

In Table 6-1, the * in the *Property Term* column indicates cases where the *Property Term* is the same as either the *Representation Term* or *Object Class Term*, and is consequently dropped from the *Dictionary Entry Name*.

The catalogue is intended to be part of a larger *Core Component Library*. The *Core Component Library* will consist of the following parts:

- *Core Component Types* and *Data Types*
- *Core Component Catalogue*, including *Basic Core Components, Association Core Components*, and *Aggregate Core Components*
- *Catalogue of Business Information Entities*

6.1.6 Catalogue of Business Information Entities

For the same reasons that a *Core Components Catalogue* is necessary, a *Catalogue of Business Information Entities* is also required. Predefined *Business Information Entities* are not provided in this specification. Rather, the working registries and the groups defining business messages will be responsible for developing a *Catalogue of Business Information Entities* that will include *Basic, Association,* and *Aggregate Business Information Entities.*

6.2 Context

This section fully describes applicable rules and applications for the use of *Context* in *Core Component* discovery, analysis, and use to include *Context Categories* and their values, and the *Constraint Language*.

6.2.1 Overview of Context Specification

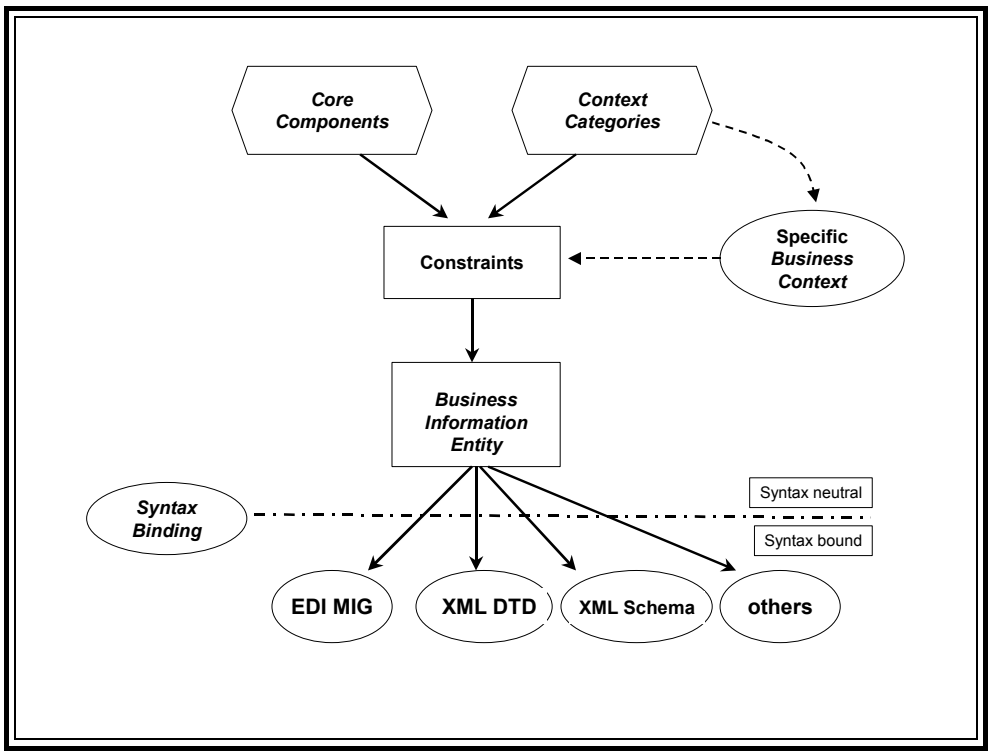
Whenever business collaboration takes place between specific trading partners, data is exchanged in the form of business messages. When used as such, that data exists in a particular *Business Context*. In its simplest form, this is the idea of *Context* as used in ebXML. The *Context* in which the business collaboration takes place can be specified by a set of categories and their associated values.

The *Core Components* have no *Context* independent of their use. The *Context* mechanism provides a full semantic qualification for the *Core Component* used in a *Business Process*. Figure 6-3 shows how the *Constraint Language* applies *Business Context Categories* and specific *Business Context*(s) to *Core Components* to develop *Business Information Entities*. Qualification is to be interpreted as specialization as defined in UML. Qualification narrows the semantic concept to a more specific one. The structure of qualified *Business Information Entities* may be a subset (but never a superset) of the structure of the (unqualified) *Business Information Entities* or *Core Components* upon which they are based. That means that value ranges may be restricted, components may be removed or their repetition factor may be lowered and *Cardinality* may change from optional to mandatory. The *Business Information Entity* resulting from this process can be manifested as a model, which in turn can be used as the basis of a syntax-bound

business message description (an EDI message implementation guide, an XML schema[9], etc.)

The following sections address the *Context Categories*, and the *Constraint Language* more closely.

Figure 6-3. Operation of The Context Mechanism



6.2.1.1 Context Categories

Context Categories exist to allow users to uniquely identify and distinguish between different *Business Contexts*. Eight *Context Categories* have been identified (Table 6-2). Each of the identified categories, unless otherwise stated, uses a standard classification to provide values for the category. Constraint rules, and therefore *Business Information Entities,* are tied to a particular set of standard classifications for identifying and distinguishing *Contexts*.

6.2.1.2 Constraint Language

A *Constraint Language* is used to express the relationship between specific *Business Contexts* and how semantics are applied to the *Core Components* to produce *Business Information Entities*. The scope of this language covers two functional parts:

- *Assembly* of a large aggregate (the *Document*). The *Constraint Language* addresses how *Assembly* is done. It does not address the design or design

[9] The term XML schema includes XML Schema as defined in World Wide Web Consortium XML Schema Part 1: Structures XML Schema Definition Language, XML Document Type Definitions, Schematron, SOX, Relax NG, ASN.1, XDR or any other notation that specifies the form and information content of an XML document.

principles of business document assembly. That subject will be covered by the *Message Assembly* supplemental document.

- Refinement of the assembly as appropriate. Refinement is both the addition of semantics specific to the *Business Process*, and the restriction of the semantic model.

This separation is a convenience for implementation (it simplifies the development of processing tools) and development of standard assemblies that can then be refined by specific users (akin to how EDI standards and message implementation guides function today).

The *Constraint Language* allows, for example, simple commands indicating how *Core Components* will be used, how they will be named for these specific uses, and how to refine the *Cardinality* (if necessary). Further, conditional relationships can be expressed. Specific *Context* values or sets of values can be tied to the actions performed on *Core Components* to produce *Business Information Entities*.

[Example]

If the *Geopolitical Context* has a value of **`Anywhere in the European Union`**, and the specific *Business Context* value indicates that the *Business Process* occurs in France, then the *Context*-appropriate *Business Information Entity* can be assembled by modifying the correct *Core Component*.

The *Constraint Language* would say—If the *Geopolitical Context* equals the **`European Union`**, then take the core **`NameAddress`** component and rules to provide the correct names, *Cardinality*, and arrangement to the fields. To do business in France, the specific *Context* value for that process will trigger this rule, giving a set of appropriate *Business Semantics* (*Business Information Entities*).

6.2.1.3 Syntax Binding

The *Business Information Entity* in its standard form is a model that has no specific relationship to any given syntax. A given *Business Information Entity* can subsequently be expressed in any of a number of syntaxes through a binding process. This process is called *Syntax Binding*, and is independent of (has no relationship to) a specific syntax. The *Syntax Binding* process does not alter the semantics of the *Business Information Entity*, but simply instantiates the *Business Information Entity* for use in syntax specific documents. It may be possible to express *Syntax Binding* in an algorithm.

[B31] *Syntax Binding* shall not change the semantics of a *Business Information Entity*.

6.2.2 Approved Context Categories

Table 6-3 contains the eight approved *Context Categories*.

[C34] When describing a specific *Business Context*, a value or set of values shall be assigned to each of the approved *Context Categories* in order to describe the business situation in an unambiguous and formal way.

Table 6-3. Approved Context Categories

Context Category	Description
Business Process	The *Business Process* name(s) as described using the *UN/CEFACT Catalogue of Common Business Processes* as extended by the user.
Product Classification	Factors influencing semantics that are the result of the goods or services being exchanged, handled, or paid for, etc. (e.g. the buying of consulting services as opposed to materials)
Industry Classification	Semantic influences related to the industry or industries of the trading partners (e.g., product identification schemes used in different industries).
Geopolitical	Geographical factors that influence *Business Semantics* (e.g., the structure of an address).
Official Constraints	Legal and governmental influences on semantics (e.g. hazardous materials information required by law when shipping goods).
Business Process Role	The actors conducting a particular *Business Process*, as identified in the *UN/CEFACT Catalogue of Common Business Processes*.
Supporting Role	Semantic influences related to non-partner roles (e.g., data required by a third-party shipper in an order response going from seller to buyer.)
System Capabilities	This *Context Category* exists to capture the limitations of systems (e.g. an existing back office can only support an address in a certain form).

6.2.2.1 Business Process Context

In describing a business situation, generally the most important aspect of that situation is the business activity being conducted. *Business Process Context* provides a way to unambiguously identify the business activity. To ensure consistency with *Business Process* activities, it is important to use a common point of reference. The definitive point of reference for international standards is the UN/CEFACT *Catalogue of Common Business Processes.*

[C35] Assigned *Business Process Contexts* shall be from the standard hierarchical classification: provided as part of the *UN/CEFACT Catalogue of Common Business Processes.*

[C36] *Business Process Context* values may be expressed as a single *Business Process*, or as a hierarchical set of *Business Processes*.

[C37] *Business Process Context* values may be taken from extensions to the *Business Processes* described in the *UN/CEFACT Catalogue of Common Business Processes* as provided for in that document.

[C38] When *Business Process* extensions are used, they shall include full information for each value sufficient to unambiguously identify which extension is providing the value used.

6.2.2.2 Product Classification Context

The *Product Classification Context* describes those aspects of a business situation related to the goods or services being exchanged by, or otherwise manipulated, or concerned, in the *Business Process*. Recognized code lists exist that provide authoritative sources of *Product Classification Contexts*.

[C39] A single value or set of values may be used in a *Product Classification Context.*

[C40] If a hierarchical system of values is used for *Product Classification Context*, then these values may be at any level of the hierarchy.

[C41] If more than one classification system is being employed, an additional value specifying which *Classification Scheme* has supplied the values used shall be conveyed.

[C42] *Product Classification Context* code values shall be taken from recognized code lists to include:

- *Universal Standard Product and Service Specification* (UNSPSC)
  - Custodian: Electronic Commerce Code Management Association (ECCMA)
- *Standard International Trade Classification* (SITC Rev .3)
  - Custodian: United Nations Statistics Division (UNSD)
- *Harmonized Commodity Description and Coding System* (HS)
  - Custodian: World Customs Organization (WCO)
- *Classification Of the purposes of non Profit Institutions serving households* (COPI)
  - Custodian: UNSD (This provides a mapping between the first three.)

6.2.2.3 Industry Classification Context

The *Industry Classification Context* provides a description of the industry or sub-industry in which the *Business Process* takes place.

[C43] An *Industry Classification Context* may contain a single value or set of values at any appropriate level of the value hierarchy.

[C44] The *Industry Classification Context* value hierarchy must be identified.

[C45] *Industry Classification Context* code values shall be taken from recognized code lists to include:

- *International Standard Industrial Classification* (ISIC)
  - Custodian: UNSD
- *Universal Standard Product and Service Specification* (UNSPSC) Top-level Segment [digits 1 and 2] used to define industry.
  - Custodian: ECCMA

[Note]

There are many other industry *Classification Schemes* that may be used for *Industry Classification Context*.

6.2.2.4 Geopolitical Context

Geopolitical Contexts allow description of those aspects of the *Business Context* that are related to region, nationality, or geographically based cultural factors.

[C46] *Geopolitical Context* shall consist of appropriate continent, economic region, country, and region identifiers.

[C47] *Geopolitica Context* may associate one or more values with any business message or component.

[C48] *Geopolitical Context* shall employ the following hierarchical structure:

Global

[Continent]

[**Economic Region**]

[**Country**] - ISO 3166.1

[**Region**] - ISO 3166.2

[C49] At any level of the *Context* hierarchy, a value may be a single value, a named aggregate, or cross-border value.

[C50] *Geopolitical Context* hierarchy values shall be structured as follows:

- **Single Value:** A single value indicating a single continent, economic region, country, or region, depending on position within the hierarchy.
- **Named Aggregate:** A related group of values (which may themselves be single values, named aggregates, or cross-border pairs of values), which have been related and assigned a name. A named aggregate contains at least two values.

- **`Cross-Border:`** One or more pairs of values, designated **`To`**, **`From`**, or **`Bi-directional`**, indicating the direction of cross-border *Context*. Values may be named aggregates or single values.

[Example]

The following example shows an extract of the basic, single-value hierarchy of recommended values, based on the common ISO 3166.1 *Country Codes*. (The value at the top of any hierarchy is always understood to be *Global*.)

```
Europe

    Eastern Europe

        AL – ALBANIA

        AM – ARMENIA
```

[C51] Points in the *Geopolitical Context* hierarchy shall be specified by the use of the node value, or by the full or partial path.

[C52] The full path of the *Geopolitical Context* hierarchy must be used to understand the hierarchy when complex constructs are employed.

[C53] A specific level in the *Geopolitical Context* hierarchy is understood to inherit all of the properties within its specific hierarchical path except where otherwise specified.

[C54] *Geopolitical Context* values shall be taken from ISO 3166.1 and 3166.2

6.2.2.5 Official Constraints Context

The *Official Constraints Context Category* describes those aspects of the business situation that result from legal or regulatory requirements and similar official categories. This category contains two distinct parts:

- Regulatory and Legislative. These are normally unilateral in nature and include such things as Customs Authority regulations.

- Conventions and Treaties. These are normally bi- or multilateral agreements and as such are different from regulatory and legislative constraints.

[C55] The *Official Constraints Context* shall consist of at least two values:

- Identification of the legal or other classification used to identify the *Context* values.

- Identification of the official constraint itself. These values may represent a hierarchical structure depending on the official constraints system being referenced.

Because there is no known global classification of all *Official Constraints Contexts* as used here, any implementation must provide a set of recognized official constraints classifications for use within the appropriate *Core Components* registry implementation.

[C56] Individual *Core Component* implementations shall register used official constraint *Classification Schemes* with the appropriate supporting *Core Components* registry implementation.

6.2.2.6 Business Process Role Context

The *Business Process Role Context* describes those aspects of a business situation that are specific to an actor or actors within the *Business Process*. Its values are taken from the set of *Role* values provided by the UN/CEFACT *Catalogue of Common Business Processes*. A *Business Process Role Context* is specified by using a value or set of values from this source.

[C57] *Business Process Role Context* values shall be taken from an approved list provided by the *Business Process* model library being employed.

[C58] The *UN/CEFACT Catalogue of Common Business Processes* shall be the definitive source of *Business Process Role Context* values for all UN/CEFACT *Business Information Entities*.

6.2.2.7 Supporting Role Context

The *Supporting Role Context* identifies those parties that are not active participants in the *Business Process* being conducted but who are interested in it. A *Supporting Role Context* is specified with a value or set of values from a standard classification.

[C59] *Supporting Role Context* values shall be taken from the UN/EDIFACT *Code List for DE 3035 Party Roles.*

> [Note]
>
> Users are cautioned that duplication exists in the current version of the required code list. UN/CEFACT will review this code list to clarify duplicates and identify non-*Supporting Role Context* values.

6.2.2.8 System Capabilities Context

This category identifies a system, a class of systems or standard in the business situation. The *System Capabilities Context* requires a least one pair of values: an identification of the *Classification Scheme* being used and a value from that scheme. A valid *System Capabilities Context* may include more than one such pair of values.

[C60] *Systems Capabilities Context* values shall consist of pairs of values. Each pair shall be comprised of an identification of the referenced *Classification Scheme* and the value(s) being employed.

[Note]

There is no known classification of all types of information systems and standards. It is recommended that a mechanism for the registration of system and standard names be provided by the ebXML registry, as valid values for the *System Capabilities Context.*

6.2.3 Context Values

A specific *Business Context* is formally described using a set of *Context* values. Every *Context Category* must have a valid value, even if this value is **`In All Contexts`** or **`None`**. The value **`None`** is appropriate for *Official Constraints Context* because there will be instances where there are no official constraints.

[C61] The **`In All Contexts`** value shall be a valid value for every *Context Category* except for *Official Constraints Context*.

[C62] The value **`None`** shall be a valid value for *Official Constraints Context*.

6.2.4 Core Components Context Constraints Language

The *Core Component*s *Context Constraints Language* consists of a set of constructs (See Table 6-3) that allow users to express the relationships between specific business situations and the specific structure and meaning of business data used in that situation. The *Constraints Language* refers to specific *Context*s as described in the *Context Categories* specification and uses *Unique Identifiers* to refer to *Core Components* semantic models. The constraints applied to *Core Components* in specific *Business Context*s to generate *Business Information Entities* are expressed using the *Constraints Language*.

[Note]

The ebXML *Unique Identifier* is fully described in the *ebXML Technical Architecture Specification Version 1.04*. Its construct is fully specified in the *ebXML Registry Specification 2.0.*

[C63] The *Core Component*s *Context Constraints Language* shall be used to describe the constraints being applied to *Core Components* to develop *Business Information Entities*.

An **`Assembly`** is the overall expression of a single set of *Assembly Rules*, which groups a set of unrefined *Business Information Entities* in to a larger structure. When working with pre-assembled standard document sets, it should not be necessary for users to create **`Assembly`** constraints.

[C64] **`Assembly`** shall be the top-level construct in any set of *Assembly Rules.*

The *ContextRules Construct* is the overall expression of a single set of rules that are used to apply *Context* to *Core Components*. The *ContextRules* adds the full semantic and structural refinement to the *Core Components* to produce *Business Information Entities*.

This mechanism supports specifying *Cardinality*, addition and subtraction of child *Core Component*s, renaming of those children, assigning *Business Information Entity* names to the *Context*-specific instances of the *Core Components*, and adding structure to develop *Aggregate Business Information Entities*.

[C65] A single set of *Context* rules shall be described using the *ContextRules* expression.

Table 6-3 Core Components Context Constraints Language

Construct	Component Constructs	Description
Assembly		An **Assembly** contains at least one **Assemble**, optionally either an **@ID** or an **@idref**, and optionally one **@version** **Note:** An **Assembly** is the top level construct in a set of *Assembly Rules*
	Assemble	List of assembled *Core Components* to be grouped together to form **BIEs**
	@ID	ID of an **Assembly**
	@idref	Reference to an **Assembly** ID
	@version	Version of the *Assembly Rules* document.
Assemble		An **Assemble** contains at least either a **CreateBIE** or a **CreateGroup**, optionally either an **@ID** or an **@idref**, and one **@name**
	CreateBIE	List of *Core Components*
	CreateGroup	Create a group of **BIEs**
	@name	Name of the highest-level **BIE** being assembled
	@ID	**ID** of an **Assemble** rule
	@idref	Reference to an **Assemble ID**
CreateGroup		A **CreateGroup** contains at least one of **CreateGroup** or **CreateBIE** or **UseBIE** or Annotation, optionally an **@ID** or an **@idref**, and one **@type**
	@type	Type of group to be created (the only permitted values are ‘sequence’ and ‘choice’)
	@ID	ID of a **CreateGroup** rule
	@idref	Reference to **CreateGroup** ID
	CreateGroup	Create a group of **BIEs**
	CreateBIE	Create a **BIE**
	UseBIE	Use the named **BIE** from among the children of the **BIE** being created.
	Annotation	Insert **Annotation**

Construct	Component Constructs	Description
CreateBIE		A **CreateBIE** rule contains an optional Name followed by an optional Type followed by a **MinOccurs** followed by a **MaxOccurs** followed by zero or more **CreateGroup** or **Rename**, or **UseBIE**, or **Condition** or **Annotation**, optionally an **@ID** or an **@idref**, and an optional **@location**
	Type	Type of **BIE** to be created – a reference to a *Core Component*
	MinOccurs	Minimum occurrences for the **BIE** created
	MaxOccurs	Maximum occurrences for the **BIE** created. One possible value (other than integer) is **unbounded**.
	@ID	**ID** of the created **BIE**
	@idref	Reference to the **ID** of another created **BIE**
	Name	Name of the **BIE** to be assembled
	@location	Location of the **BIE** to be assembled (i.e. query to the registry)
	Rename	Renames children of the created **BIE**
	Condition	Condition under which this rule should apply
	Annotation	Insert **Annotation**
Name		A **Name** contains only a string of characters
Type		A **Type** contains only a string of characters. It represents a type in the output – representation class or *Core Component*, depending on where used.
Rename		A **Rename** rule contains optionally an **@ID** or an **@idref**, and one **@from** and one **@to**
	@ID	**ID** of the **Rename** rule
	@idref	Reference to the **ID** of another **Rename** rule
	@from	Original name of the child **BIE** being renamed
	@to	New name of the child being renamed
ContextRules		**ContextRules** contains one or more Rules **Note:** A **ContextRules** is the top level construct in a set of *Context* rules
	Rule	List of refinement and qualification rules to be applied

Construct	Component Constructs	Description
ContextRules (Continued)	**@ID**	**ID** of the **ContextRules** rule
	@idref	Reference to the **ID** of another **ContextRules** rule
	@version	Version of the **ContextRules** document.
Rule		A Rule contains one or more Taxonomy, followed by one or more Condition, one **@apply**, and an optional **@order**.
	@apply	(See note below)
	Condition	When rule should be run
	@order	Defines order for running rules. Rules with lower value for order are run first
	Taxonomy	List of taxonomies used in a rule that employs hierarchical conditions.
Taxonomy		A **Taxonomy** contains a **@context** and a @ref, and optionally an **@ID** or an **@idref**
	@ref	Pointer to a **taxonomy**.
	@context	Name of the *Context* category to which this **Taxonomy** applies
	@ID	**ID** of the **Taxonomy** rule
	@idref	Reference to the **ID** of another **Taxonomy** rule
Condition		A **Condition** contains at least one of **Action** or **Condition** or **Occurs**, one **@test**, and optionally an **@ID** or an **@idref**
	Action	What happens when rule is run
	Condition	A nested condition
	Occurs	Specify number of occurrences
	@ID	**ID** of the Condition rule
	@idref	Reference to the **ID** of another **Condition** rule
	@test	Boolean expression testing whether the rule should be run.
Action		An **Action** contains at least one of **Add** or **Occurs** or **Subtract** or **Condition** or **Comment** or **Rename**, one **@applyTo** and optionally an **@ID** or an **@idref**
	Add	Add a component to the content model
	Subtract	Subtract a component from the content model
	Occurs	Constrain or expand the number of occurrences of the component
	Condition	When rule should be run

Construct	Component Constructs	Description
Action (Continued)	**Comment**	Add a comment
	Rename	Rename a component
	@ID	**ID** of the **Condition** rule
	@idref	Reference to the **ID** of another **Condition** rule
	@apply To	Name of the component to apply this rule to
Add		Add contains a **MinOccurs** followed by a **MaxOccurs** followed by at least one of an optional **BIE** or an optional **Attribute**, or a **CreateGroup** or an **Annotation**, optionally an **@ID** or an **@idref**, an optional **@before** or an optional **@after**
	MinOccurs	Minimum number of times that the new instance must occur
	MaxOccurs	Maximum number of times that the new instance can occur
	@before	Specifies before which component the addition should occur.
	@after	Specifies after which component the addition should occur.
	CreateGroup	Create a group of **BIEs**
	BIE	Adds a new **BIE** to the content model.
	Attribute	Adds a new **Attribute** to the content model
	Annotation	Insert **Annotation**
	@ID	**ID** of the **Add** rule
	@idref	Reference to the **ID** of another **Add** rule
Subtract		Subtract contains one or more of **BIE** or **Attribute**, and optionally an **@ID** or an **@idref**
	BIE	Removes a **BIE** from the content model.
	Attribute	Removes a **Attribute** from the content model
	@ID	**ID** of the **Subtract** rule
	@idref	Reference to the **ID** of another **Subtract** rule
Occurs		Occurs contains a **MinOccurs**, followed by a **MaxOccurs**, followed by one or more **BIEs**, and optionally an **@ID** or an **@idref**
	BIE	Changes an optional **BIE** to required.
	MinOccurs	Overrides the minimum number of occurrences for this **BIE**

Construct	Component Constructs	Description
Occurs (Continued)	**MaxOccurs**	Overrides the maximum number of occurrences for this **BIE**
	@ID	**ID** of the **Occurs** rule
	@idref	Reference to the **ID** of another **Occurs** rule
BIE		A **BIE** contains a **Name**, followed by an optional **Type**, followed by zero or more **Attribute**, followed by zero or more **Annotation**, and optionally an **@ID** or an **@idref**
	Name	Name of **BIE** to be modified
	Type	Type of **BIE** – the *Core Component* - required only if contained in an **Add** tag
	Attribute	**Attribute(s)** of this **BIE**
	Annotation	Insert **Annotation**
	@ID	**ID** of the **BIE** rule
	@idref	Reference to the **ID** of another **BIE** rule
Attribute		An **Attribute** contains an optional **Name** followed by an optional **Type**, followed by an optional **Use**, followed by an optional **Value**, followed by zero or more **Annotation**, and optionally an **@ID** or an **@idref**, and an optional **@apply**To
	Annotation	Insert **Annotation**
	Name	Name of **Attribute** to be modified
	Type	Type of the **Attribute** (representation class)
	Use	Indicates whether required or optional, and if the latter whether required or optional. If optional, indicates the presence of a default. May supply a fixed value instead.
	Value	Indicates whether required or optional, and if the latter whether required or optional. If optional, indicates the presence of a default. May supply a value to be modified
	@apply To	Node to apply action to
	@ID	**ID** of the **Attribute** rule
	@idref	Reference to the **ID** of another **Attribute** rule

Construct	Component Constructs	Description
UseBIE		A **UseBIE** contains zero or more of Annotation or **CreateGroup** or **UseBIE**, and optionally an **@ID** or an **@idref**. An **@name** is required in any **UseBIE** that does not use a **CreateGroup**.
	@name	Name of the **BIE** being used
	CreateGroup	Create a group of **BIEs**
	UseBIE	Use the named **BIE** from among the children of the **BIE** being created.
	Annotation	Insert **Annotation**. This design is intended to mirror the annotation functionality found in the W3C Schema Specification.
	@ID	**ID** of the **UseBIE** rule
	@idref	Reference to the **ID** of another **UseBIE** rule
Comment		Ubiquitous. Records comments about the rules document at the location it appears. It is not intended to be output in the resulting semantic model.
MinOccurs		Minimum number of occurrences in the output
MaxOccurs		Maximum number of occurrences in the output
Annotation		An **Annotation** contains zero or more of either **Documentation** or **Appinfo**, and optionally an **@ID** or an **@idref**
	Documentation	Used to include documentation
	Appinfo	Used to include application specific information
	@ID	**ID** of the **Annotation**
	@idref	Reference to the **ID** of another **Annotation**
Documentation		**Documentation** contains optionally an **@ID** or an **@idref**
	@ID	**ID** of the **Documentation**
	@idref	Reference to the **ID** of another **Annotation**
Appinfo		Documentation contains optionally an **@ID** or an **@idref**
	@ID	**ID** of the **Appinfo**
	@idref	Reference to the **ID** of another **Appinfo**

[Note]

Table Key: **@** indicates *Properties* of the construct being defined. For example, **@ID**, **@idref** and **@version** are *Properties* of **Assembly**.

6.2.4.1 Assembly Construct

The **MinOccurs** and **MaxOccurs** constructs in the **CreateBIE** construct specify the occurrence that the created *Business Information Entity* will have in the resulting semantic model.

[C66] A *Business Information Entity* created with **MinOccurs** = **1** and **MaxOccurs** = **1** shall be specified in the resulting semantic model as occurring only once.

[C67] An **Assembly** may contain more than one assembled top-level semantic model.

6.2.4.2 ContextRules Construct

Several built-in variables are used to access *Context* information. These variables correspond to the identified *Context Categories*. All of these variables have string values.

[C68] The **Apply Attribute** of the **ContextRules** construct type shall be used for determining the behaviour of rules that use hierarchical values.

[C69] Allowed **Apply Attribute** values are:

- **exact** - match only if the value in the provided *Context* is precisely the same as that specified in the rule
- **hierarchical** - match if the value provided is the same or a child of that specified in the rule.

[Example]

If the **ContextRules** specifies the region **Europe**, the value **France** would match only if the **Apply** *Attribute* is set to **hierarchical** (**exact** being the default).

[C70] The *Attribute* construct has four optional children in its content model, of which at least one must be present.

[C71] When the *Attribute* construct is used to refine an existing *Attribute*, then a value must be specified for **@apply**To on that *Attribute* construct.

[C72] **ContextRules** must refer to the names of the *Core Components*, and not the names given to the resulting *Business Information Entities* elsewhere in the Rules.

[Example]

Given a source that contains an optional child type named '**X**', a rule can be applied to rename '**X**' to '**Y**', but a rule to make '**Y**' required, rather than '**X**', would be illegal.

6.2.4.3 Output Constraints

[C73] Semantic models and document definitions produced through the application of ***Assembly*** and *Context Rules* must contain the metadata about the rules and *Context* that produced them.

6.2.4.4 Ordering and Application

There is an explicit *Order Property* on the *Rule* construct that applies a sequence to the application of a set of rules. It is an error for two *Rule* constructs to have the same value for the *Order Property*. In a single set of *ContextRules*, users should be careful not to sequence rules in a way that would preclude their execution—such as adding an *Attribute* to a *Business Information Entity* that has not been added yet by the rules.

[C74] The *Order Property* on the *Rule* construct shall determine the sequence for the application of the applicable set of rules.

[C75] Two *Rule* constructs must not have the same value for the *Order Property*.

7 Technical Details - Core Component Registry/Repository Storage

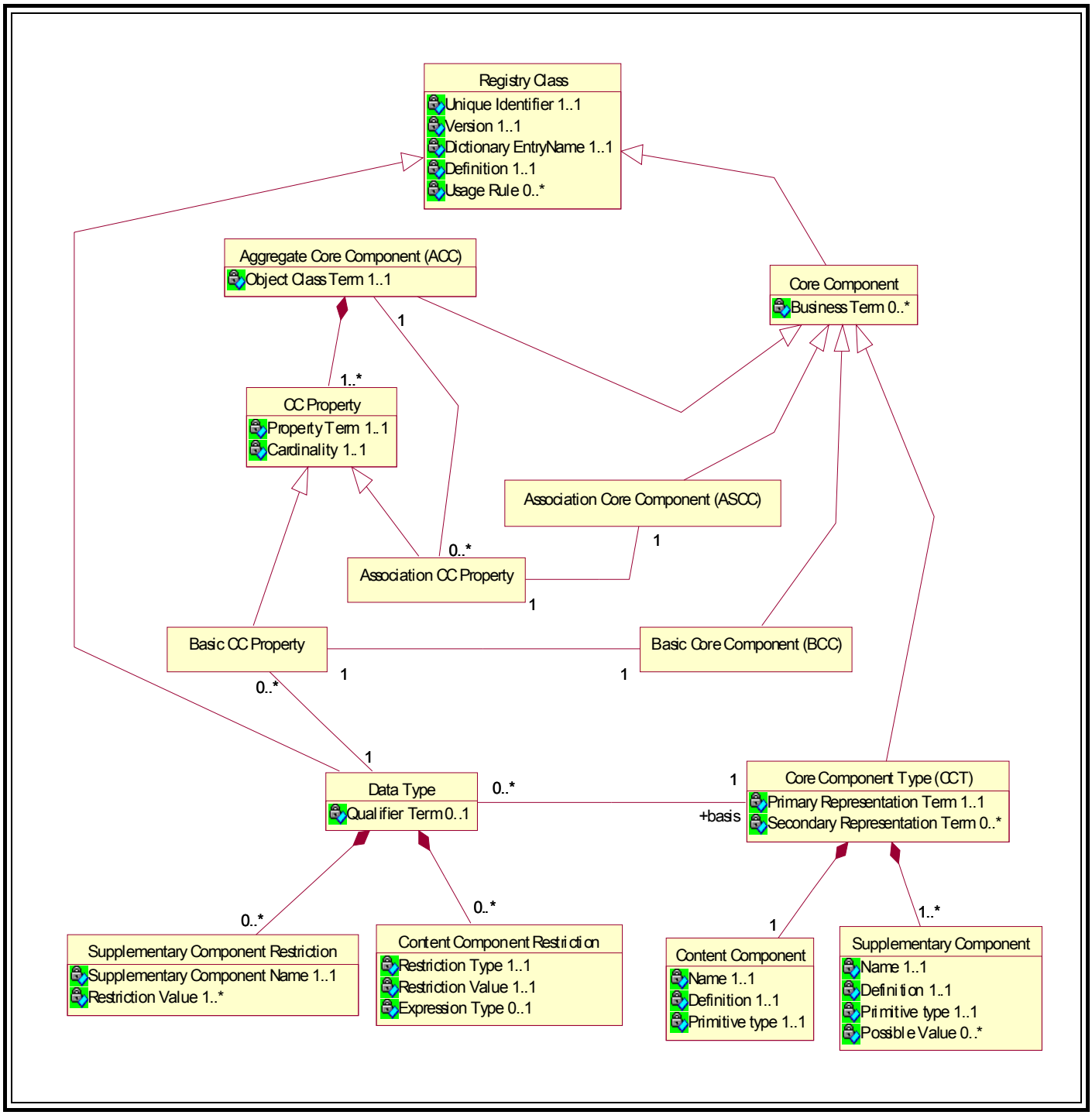
Section 6 specified the basic definitions for *Core Components*, *Data Types*, *Business Information Entities* and *Context*.

This section details exact information required for design of *Unified Modeling Language* objects to store *Core Components*, *Data Types*, *Business Information Entities*, *Context* and relevant associated metadata in the registry/repository. Both parts contain requirements that must be addressed by developers and users of *Core Components*. Further, both parts contain requirements that must be satisfied in the supported registry and repository suite of technical specifications and any corresponding overarching information technology framework that uses *Core Components* as the linchpin between process modelling and trade.

7.1 Storing Core Components

This section fully describes *Core Component* storage details. Figure 7-1 is the *Unified Modeling Language* model of all aspects of *Core Components* and fully describes the types of *Core Components* and their relationships as a requirement of storage.

Figure 7-1. Core Components and Data Types - Full Definition



7.1.1 Stored Core Components

[S1] *Core Components* are a particular category of *Registry Classes*. As such, all stored *Core Components* shall include the following *Attributes*:

- ***Unique Identifier* (mandatory):** The identifier that references a *Core Component* instance in a unique and unambiguous way.

- ***Version* (mandatory):** An indication of the evolution over time of a *Core Component* instance.

- ***Dictionary Entry Name* (mandatory):** The official name of a *Core Component*.

- ***Definition* (mandatory):** The semantic meaning of a *Core Component*.

- ***Usage Rule* (optional, repetitive):** A constraint that describes specific conditions that are applicable to the *Core Component*.

[S2] Stored *Core Components* shall always be defined as one of the four recognized types—*Basic Core Component, Association Core Component, Aggregate Core Component* or *Core Component Type*.

[S3] Stored *Core Components* shall include the following *Attributes*:

- ***Business Term* (optional, repetitive):** A synonym term under which the *Core Component* is commonly known and used in a business. B*usiness Terms* may be expressed in any language. A *Core Component* may have several *Business Terms* or synonyms.

7.1.2 Stored Aggregate Core Components

[S4] *Aggregate Core Components* are a particular category of *Core Components*. As such, stored *Aggregate Core Components* shall include all *Attributes* of stored *Core Components*.

[S5] Stored *Aggregate Core Components* shall contain one or more *Core Component Properties*.

[S6] Stored *Aggregate Core Components* can be referenced by one or more *Association Core Component Properties* of other *Aggregate Core Components*.

[S7] Stored *Aggregate Core Components* shall include the following *Attribute*:

- ***Object Class Term* (mandatory):** A semantically meaningful name for the *Object Class* that is represented by the *Aggregate Core Component*. It shall serve as basis for the *Dictionary Entry Name* of the *Aggregate Core Component* and for the *Dictionary Entry Name* of all *Basic* and *Association Core Components* that represent *Core Component Properties* of this *Aggregate Core Component*.

7.1.3 Stored Core Component Properties

[S8] Stored *Core Component Properties* shall be stored as part of the stored *Aggregate Core Component* to which they belong, i.e. they shall never exist independently of their owning *Aggregate Core Component*.

[S9] Stored *Core Component Properties* shall be defined as one of the two recognized types—*Basic Core Component Property* or *Association Core Component Property*.

[S10] Stored *Core Component Properties* shall include the following *Attributes*:

- ***Property Term* (mandatory):** A semantically meaningful name for the characteristic of the *Object Class* that is represented by the *Core Component Property*. It shall serve as basis for the *Dictionary Entry Name* of the *Basic* or *Association Core Component* that represents this *Core Component Property*.

- ***Cardinality* (mandatory)**: Indication whether the *Core Component Property* represents an optional, mandatory and/or repetitive characteristic of the *Aggregate Core Component*.

7.1.4 Stored Basic Core Component Properties

[S11] *Basic Core Component Properties* are a particular category of *Core Component Properties*. As such, stored *Basic Core Component Properties* shall include all *Attributes* of stored *Core Component Properties*.

[S12] Stored *Basic Core Component Properties* shall be linked to the *Data Type* that describes the possible values of the *Basic Core Component Property*.

7.1.5 Stored Association Core Component Properties

[S13] *Association Core Component Properties* are a particular category of *Core Component Properties*. As such, stored *Association Core Component Properties* shall include all *Attributes* of stored *Core Component Properties*.

[S14] Stored *Association Core Component Properties* shall be linked to the *Aggregate Core Component* that describes the structure of the *Association Core Component Property*.

7.1.6 Stored Basic Core Components

[S15] *Basic Core Components* are a particular category of *Core Components*. As such, stored *Basic Core Components* shall include all *Attributes* of stored *Core Components*.

[S16] Stored *Basic Core Components* shall represent a *Basic Core Component Property* of a particular *Aggregate Core Component*.

7.1.7 Stored Association Core Components

[S17] *Association Core Components* are a particular category of *Core Components*. As such, stored *Association Core Components* shall include all *Attributes* of stored *Core Components*.

[S18] Stored *Association Core Components* shall represent an *Association Core Component Property* of a particular *Aggregate Core Component*.

7.1.8 Stored Core Component Types

[S19] *Core Component Types* are a particular category of *Core Components*. As such, stored *Core Component Types* shall include all *Attributes* of stored *Core Components*.

[S20] Stored *Core Component Types* shall include one *Content Component* that defines the *Primitive Type* and one or more *Supplementary Components* that give meaning to the *Content Component*.

[S21] Stored *Core Component Types* shall not reflect business meaning.

[S22] Stored *Core Component Types* shall include the following *Attributes*:

- ***Primary Representation Term* (mandatory):** A semantically meaningful name that forms the basis for the *Dictionary Entry Name* of the *Core Component Type*. It can also form the basis for the *Dictionary Entry Name* of *Data Types* that are based on the *Core Component Type*.

- ***Secondary Representation Term* (optional, repetitive):** A semantically meaningful name that represents a meaningful subset of the *Core Component Type*. It can form the basis for the *Dictionary Entry Name* of *Data Types* that are based on the *Core Component Type*.

7.1.9 Stored Supplementary Components

[S23] Stored *Supplementary Component*s shall be stored as part of the stored *Core Component Type* to which they belong, i.e. they shall never exist independently of their owning *Core Component Type*.

[S24] Stored *Supplementary Component*s shall include the following *Attributes*:

- **`Name` (mandatory): `Name`** in the Registry of a *Supplementary Component* of a *Core Component Type*.

- ***Definition* (mandatory):** A clear, unambiguous and complete explanation of the meaning of a *Supplementary Component* and its relevance for the related *Core Component Type*.

- ***Primitive type* (mandatory)**: *PrimitiveType* to be used for the representation of the value of a *Supplementary Component*.

> [Note]
>
> Possible values for *Primitive Type* are **`String, Decimal, Integer, Boolean, Date`** and **`Binary`**.

- **`Possible Value` (optional, repetitive)**: one possible value of a *Supplementary Component*.

> [Note]
>
> **`Possible values`** shall only be stored if all possible values can be defined by an enumeration (e.g. list of quantity units).

7.1.10 Stored Content Components

[S25] Stored *Content Component*s shall be stored as part of the stored *Core Component Type* to which they belong, i.e. they shall never exist independently of their owning *Core Component Type*.

[S26] Stored *Content Components* shall include the following *Attributes*:

- **`Name` (mandatory): `Name`** in the Registry of a *Content Component* of a *Core Component Type*.
- **`Definition` (mandatory):** A clear, unambiguous and complete explanation of the meaning of a *Content Component*.
- ***Primitive type*** **(mandatory)**: *Primitive Type* to be used for the expression of the value of an instance of a *Basic Core Component* based on the associated *Core Component Type*.

7.2 Storing Data Types

This section fully describes *Data Type* storage details.

7.2.1 Stored Data Types

[S27] *Data Types* are a particular category of *Registry Classes*. As such, all stored *Core Components* shall include the following *Attributes*:

- ***Unique Identifier*** **(mandatory):** The identifier that references a *Data Type* instance in a unique and unambiguous way.
- ***Version*** **(mandatory):** An indication of the evolution over time of a *Data Type* instance.
- ***Dictionary Entry Name*** **(mandatory):** The official name of a *Data Type*.
- ***Definition*** **(mandatory):** The semantic meaning of a *Data Type*.
- ***Usage Rule*** **(optional, repetitive):** A constraint that describes specific conditions that are applicable to the *Data Type*.

[S28] Stored *Data Types* shall include the following *Attribute*:

- ***Qualifier Term*** **(mandatory)**: A semantically meaningful name that differentiates the *Data Type* from its underlying *Core Component Type*. It shall serve as basis for the *Dictionary Entry Name* of the *Data Type*.

[S29] Stored *Data Types* shall have a *Core Component Type* as their basis.

[S30] Stored *Data Types* may include one or more *Content Component Restrictions* and one or more *Supplementary Component Restrictions* to provide additional information on the relationship between the *Data Type* and its corresponding *Core Component Type*. They identify restrictions on the format of the *Content*

Component and/or restrictions on the possible values of the *Supplementary Components* of the corresponding *Core Component Type*.

[Example]

The *Core Component Type* **Quantity** has a *Supplementary Component* **Quantity Unit** with possible values like **gram** and **second**. A *Data Type* that is used for a *Basic Core Component* such as **Person. Weight. Quantity** will not accept **second** as quantity unit.

7.2.2 Stored Content Component Restrictions

[S31] Stored *Content Component Restrictions* shall only be used to define format restrictions on the *Primitive Type* of the *Content Component* of the *Core Component Type* on which the *Data Type* is based. The list of allowed format restrictions per *Primitive Type* is defined in Table 7-1.

Table 7-1. Primitive Types and their related facets

Primitive Type	Format Restriction	Definition
String	**Expression**	Defines the set of characters that can be used at a particular position in a string.
String	**Length**	Defines the required length of the string.
String	**Minimum Length**	Defines the minimum length of the string. [Note] This format restriction shall not be used in combination with the **Length** format restriction.
String	**Maximum Length**	Defines the maximum length of the string. [Note] This format restriction shall not be used in combination with the **Length** format restriction.
String	**Enumeration**	Defines the exhaustive list of allowed values.
Decimal, Integer	**Total Digits**	Defines the maximum number of digits to be used.
Decimal	**Fractional Digits**	Defines the maximum number of fractional digits to be used.
Decimal, Integer	**Minimum Inclusive**	Defines the lower limit of the range of allowed values. The lower limit is also an allowed value.
Decimal, Integer	**Maximum Inclusive**	Defines the upper limit of the range of allowed values. The upper limit is also an allowed value.

Primitive Type	Format Restriction	Definition
Decimal, Integer	**Minimum Exclusive**	Defines the lower limit of the range of allowed values. The lower limit is no allowed value. [Note] This format restriction shall not be used in combination with the **Minimum Inclusive** format restriction.
Decimal, Integer	**Maximum Exclusive**	Defines the upper limit of the range of allowed values. The upper limit is no allowed value. [Note] This format restriction shall not be used in combination with the **Maximum Inclusive** format restriction.
Date	**Minimum Inclusive**	Defines the lower limit of the range of allowed dates. The lower limit is also an allowed date.
Date	**Maximum Inclusive**	Defines the upper limit of the range of allowed dates. The upper limit is also an allowed date.
Date	**Minimum Exclusive**	Defines the lower limit of the range of allowed dates. The lower limit is no allowed date. [Note] This format restriction shall not be used in combination with the **Minimum Inclusive** format restriction.
Date	**Maximum Exclusive**	Defines the upper limit of the range of allowed dates. The upper limit is no allowed date. [Note] This format restriction shall not be used in combination with the **Maximum Inclusive** format restriction.

[S32] Stored *Content Component Restrictions* shall contain the following *Attributes*:

- **Restriction Type (mandatory)**: Defines the type of format restriction that applies to the *Content Component*.
- **Restriction Value (mandatory)**: The actual value of the format restriction that applies to the *Content Component*.
- **Expression Type (optional)**: Defines the type of the regular expression of the restriction value.

[Note]

The restriction values depend on the **`restriction type`** (e.g. integer for a **length `restriction type`**, list of possible values for an **enumeration `restriction type`**.).

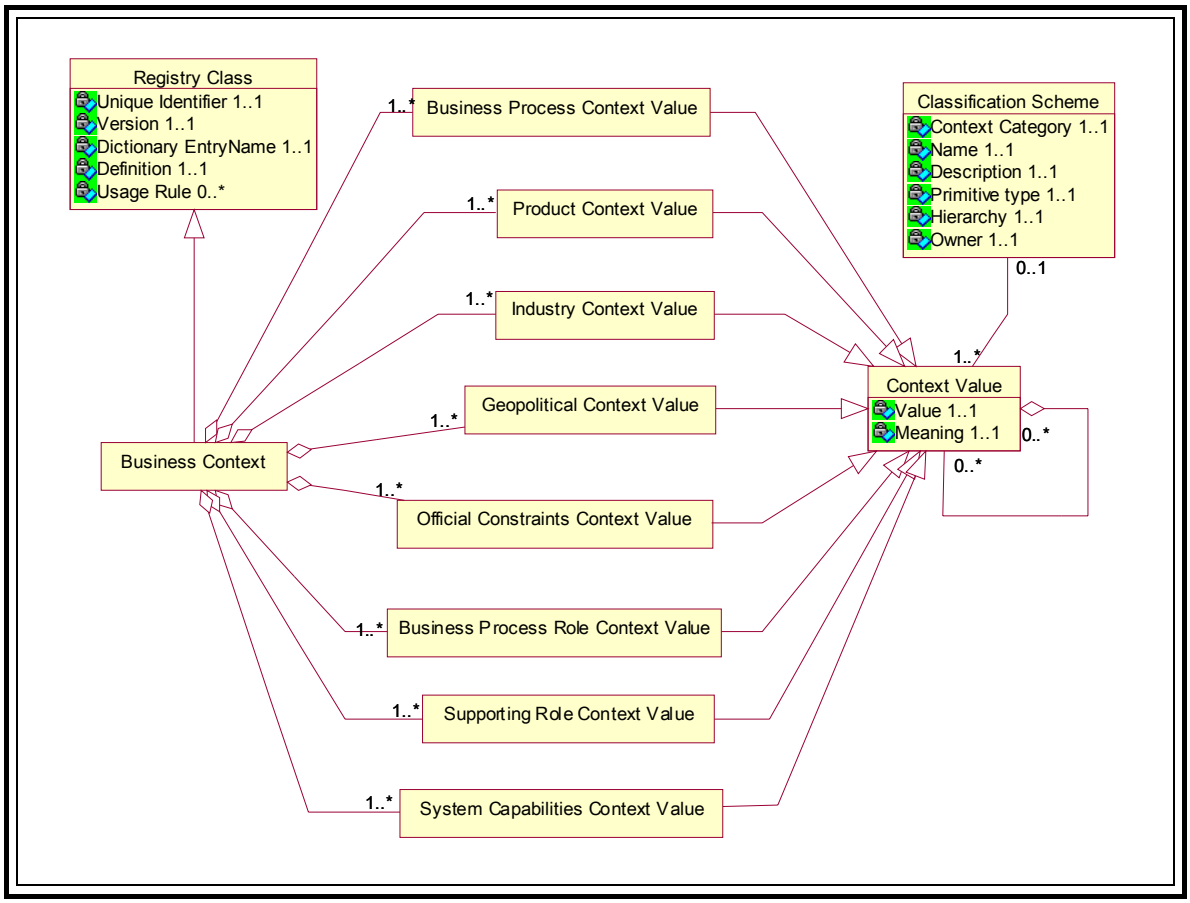
7.2.3 Stored Supplementary Component Restrictions

[S33] Stored *Supplementary Component Restrictions* shall only be used to restrict the possible values of the *Supplementary Component* of the *Core Component Type* on which the *Data Type* is based.

[S34] Stored *Supplementary Component Restrictions* shall contain the following *Attributes*:

- ***Supplementary Component* Name (mandatory):** Identifies the *Supplementary Component* on which the restriction applies.
- **`Restriction Value` (mandatory, repetitive)**: The actual value(s) that is (are) valid for the *Supplementary Component*.

Figure 7-2 Core Components Context Definition Model



7.3 Stored Context

This section fully describes *Context* storage details. Figure 7-2 is the *Unified Modeling Language* model of all aspects of *Context.* It shows that there are a number of *Context Categories* (e.g. Region, Product), which can each be described by one or more *Classification Schemes* (e.g. United Nations scheme for products, World Trade Organization scheme for products). For each *Classification Scheme* the list of possible values (and their meaning) is defined. A *Business Context* is then defined as a unique and meaningful combination of *Context* values.

7.3.1 Stored Business Contexts

[S35] *Business Contexts* are a particular category of *Registry Classes*. As such, all stored *Business Contexts* shall include the following *Attributes*:

- ***Unique Identifier*** **(mandatory):** The identifier that references a *Business Context* instance in a unique and unambiguous way.
- ***Version*** **(mandatory):** An indication of the evolution over time of *Business Context* instance.
- ***Dictionary Entry Name*** **(mandatory):** The official name of a *Business Context*.
- ***Definition*** **(mandatory):** The semantic meaning of a *Business Context*.
- ***Usage Rule*** **(optional, repetitive):** A constraint that describes specific conditions that are applicable to the *Business Context*.

[S36] Stored *Business Contexts* shall contain the combination of values for all approved *Context Categories* so as to define a unique and meaningful *Business Context*.

7.3.2 Stored Classification Schemes

[S38] Stored *Classification Schemes* shall include the following *Attributes*:

- ***Context Category*** **(mandatory):** Name used to identify the approved *Context Category* for which the *Classification Scheme* can be used.
- **`Name`** **(mandatory):** Name under which the *Classification Scheme* is known.
- ***Definition*** **(mandatory):** Definition of the *Classification Scheme*.
- ***Primitive type*** **(mandatory):** *Primitive Type* that is used for the representation of a *Context Value* in the *Classification Scheme.*
- **`Hierarchy`** **(mandatory):** Indicator describing whether the *Classification Scheme* supports a hierarchical description of the *Context*.

- **`Owner` (mandatory):** Organization that is responsible for the *Classification Scheme.*

7.3.3 Stored Context Values

[S39] Stored *Context* values shall describe a possible value of a particular *Context Category*.

[S40] Stored *Context* values shall be defined as one of the eight recognized types—*Business Process Context* value, *Product Context* value, *Industry Context* value, *Geopolitical Context* value, *Official Constraints Context* value, *Business Process Role Context* value, *Supporting Role Context* value or *System Capabilities Context* value.

[S41] Stored *Context* values may belong to a particular *Classification Scheme*.

[S42] Stored *Context* values that belong to a particular *Classification Scheme* that allows a hierarchy, may have a hierarchical **contains** relation with another *Context Value* belonging to the same *Classification Scheme*.

[S43] Stored *Context* value *(s)* shall include the following *Attributes*:

- **`Value` (mandatory)**: Value describing a particular *Context*.
- **`Meaning` (mandatory)**: Description of the meaning of the corresponding value.

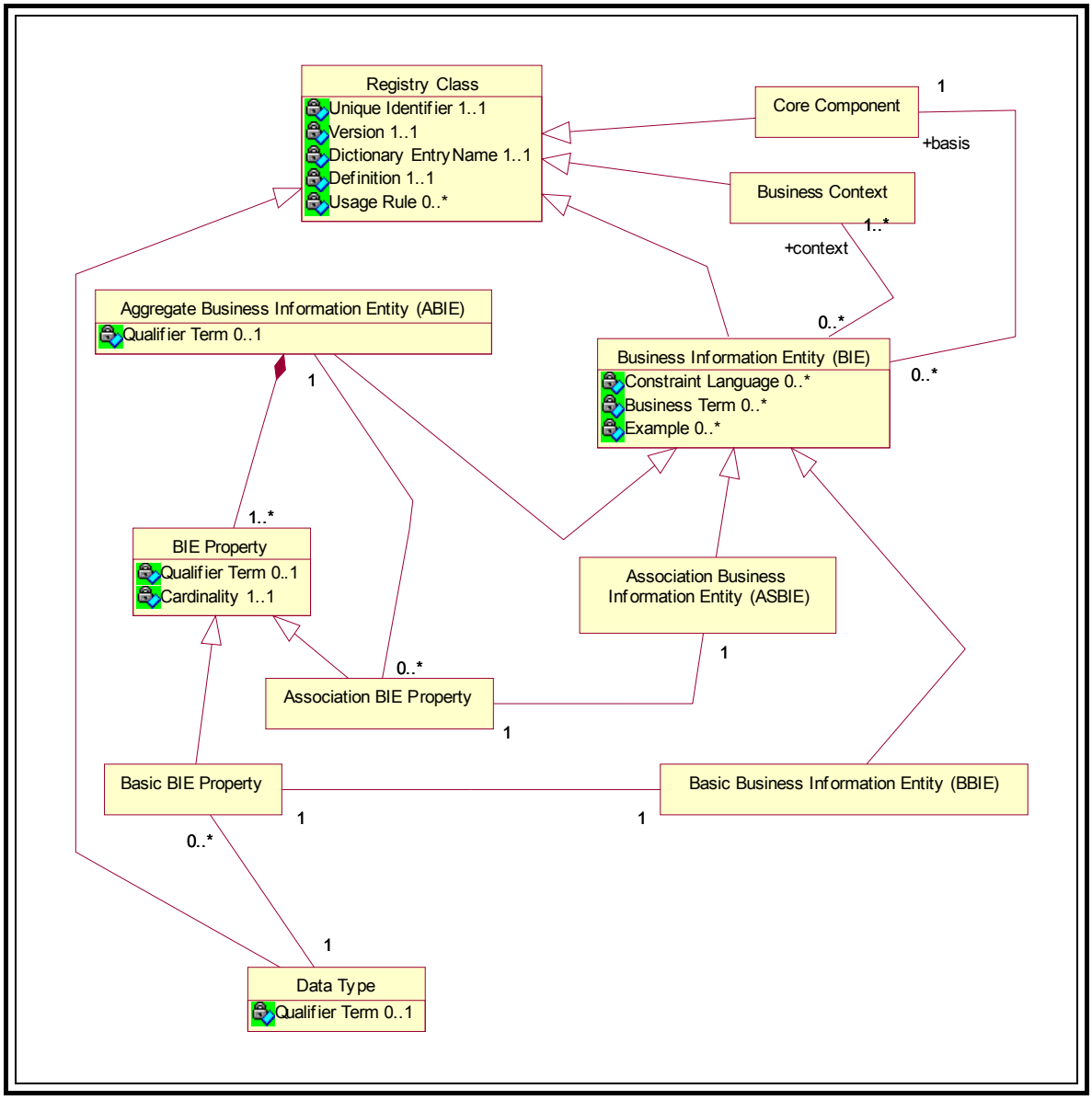
[Note]

The *Context* value is derived from a *Business Process* model which presumably uses values that have their meaning defined somewhere. For example, if the value is taken from a code list (specified in the *Classification Scheme*), then the meaning of the code should be provided by the code list specification. As an alternative solution, the meaning could optionally be a Uniform Resource Identifier that points to the definition.

7.4 Stored Business Information Entities

This section fully describes *Business Information Entity* storage details. Figure 7-3 is the *Unified Modeling Language* model of all aspects of *Business Information Entity* and fully describes the types of *Business Information Entities* and their relationships as a requirement of storage.

Figure 7-3. Business Information Entities – Full Definition



[Note]

Figure 7-3 does not show any details related to *Core Components*, *Data Types* and *Business Contexts* as these details can be found in Figures 7-1 and 7-2.

7.4.1 Stored Aggregate Business Information Entities

[S44] *Business Information Entities* are a particular category of *Registry Classes*. As such, all stored *Business Information Entities* shall include the following *Attributes*:

- ***Unique Identifier*** **(mandatory):** The identifier that references a *Business Information Entity* instance in a unique and unambiguous way.

- ***Version* (mandatory):** An indication of the evolution over time of a *Business Information Entity* instance.

- ***Dictionary Entry Name* (mandatory):** The official name of a *Business Information Entity*.

- ***Definition* (mandatory):** The semantic meaning of a *Business Information Entity*.

- ***Usage Rule* (optional, repetitive):** A constraint that describes specific conditions that are applicable to the *Business Information Entity*.

[S45] Stored *Business Information Entities* shall be based on a stored *Business Context*.

[S46] Stored *Business Information Entities* shall be based on a stored *Aggregate Core Component*, *Basic Core Component* or *Association Core Component*. They shall never be based on a *Core Component Type*.

[S47] Stored *Business Information Entities* shall be defined as one of the three recognized types—*Basic Business Information Entity, Association Business Information Entity* or *Aggregate Business Information Entity*. The type of *Business Information Entity* shall be the same as the type of its related *Core Component*:

- An *Aggregate Business Information Entity* is based on an *Aggregate Core Component*.

- A *Basic Business Information Entity* is based on a *Basic Core Component*.

- An *Association Business Information Entity* is based on an *Association Core Component*.

[S48] Stored *Business Information Entities* shall include the following *Attributes*:

- ***Constraint Language* (optional, repetitive)**: a formal description of a way the *Business Information Entity* is derived from the corresponding stored *Core Component* and stored *Business Context*.

- ***Business Term* (optional, repetitive)**: A synonym term under which the *Business Information Entity* is commonly known and used in the business. *Business Terms* may be expressed in any language. A *Business Information Entity* may have several *Business Terms* or synonyms.

- **Example (optional, repetitive)**: Example of a possible value of a *Business Information Entity*

7.4.2 Stored Aggregate Business Information Entities

[S49] *Aggregate Business Information Entities* are a particular category of *Business Information Entities*. As such, stored *Aggregate Business Information Entities* shall include all *Attributes* of stored *Business Information Entities*.

[S50] Stored *Aggregate Business Information Entities* shall contain one or more *Business Information Entity Properties*.

[S51] Stored *Aggregate Business Information Entities* can be referenced by one or more *Association Business Information Entity Properties* of other *Aggregate Business Information Entities*.

[S52] Stored *Aggregate Business Information Entities* shall include the following *Attribute*:

- ***Qualifier Term*** **(mandatory)**: Qualifies the *Object Class Term* of the associated *Aggregate Core Component*.

7.4.3 Stored Business Information Entity Properties

[S53] Stored *Business Information Entity Properties* shall be stored as part of the stored *Aggregate Business Information Entity* to which they belong, i.e. they shall never exist independently of their owning *Aggregate Business Information Entity*.

[S54] Stored *Business Information Entity Properties* shall be based on a *Core Component Property* that is stored as part of the *Aggregate Core Component* on which the owning *Aggregate Business Information Entity* is based.

[S55] Stored *Business Information Entity Properties* shall be defined as one of the two recognized types—*Basic Business Information Entity Property* or *Association Business Information Entity Property*. The type of *Business Information Entity Property* shall be the same as the type of its related *Core Component Property*:

- A *Basic Business Information Entity Property* is based on a *Basic Core Component Property*.

- An *Association Business Information Entity Property* is based on an *Association Core Component Property*.

[S56] Stored *Business Information Entity Properties* shall include the following *Attributes*:

- ***Qualifier Term*** **(optional):** Qualifies the *Property Term* of the associated *Core Component Property* in the associated *Aggregate Core Component*.

- ***Cardinality*** **(mandatory)**: Indication whether the *Business Information Entity Property* represents an optional, mandatory and/or repetitive characteristic of the *Aggregate Business Information Entity*.

7.4.4 Stored Basic Business Information Entity Properties

[S57] *Basic Business Information Entity Properties* are a particular category of *Business Information Entity Properties*. As such, stored *Basic Business Information Entity Properties* shall include all *Attributes* of stored *Business Information Entity Properties*.

[S58] Stored *Basic Business Information Entity Properties* shall be linked to the *Data Type* that describes the possible values of the *Basic Business Information*

Entity Property. This *Data Type* shall either be the same as the *Data Type* that is linked to the corresponding *Basic Core Component Property* or it shall be a more restricted *Data Type* (i.e. additional and/or more restrictive *Content Component Restrictions* and/or additional and/or more restrictive *Supplementary Component Restrictions*).

7.4.5 Stored Association Core Component Properties

[S59] *Association Business Information Entity Properties* are a particular category of *Business Information Entity Properties*. As such, stored *Association Business Information Entity Properties* shall include all *Attributes* of stored *Business Information Entity Properties*.

[S60] Stored *Association Business Information Entity Properties* shall be linked to the *Aggregate Business Information Entity* that describes the structure. This *Aggregate Business Information Entity* shall be based on the *Aggregate Core Component* that describes the structure of the corresponding *Association Core Component Property*.

7.4.6 Stored Basic Business Information Entities

[S61] *Basic Business Information Entities* are a particular category of *Business Information Entities*. As such, stored *Basic Business Information Entities* shall include all *Attributes* of stored *Business Information Entities*.

[S62] Stored *Basic Business Information Entities* shall represent a *Basic Business Information Entity Property* of a particular *Aggregate Business Information Entity*.

7.4.7 Stored Association Business Information Entities

[S63] *Association Business Information Entities* are a particular category of *Business Information Entities*. As such, stored *Association Business Information Entities* shall include all *Attributes* of stored *Business Information Entities*.

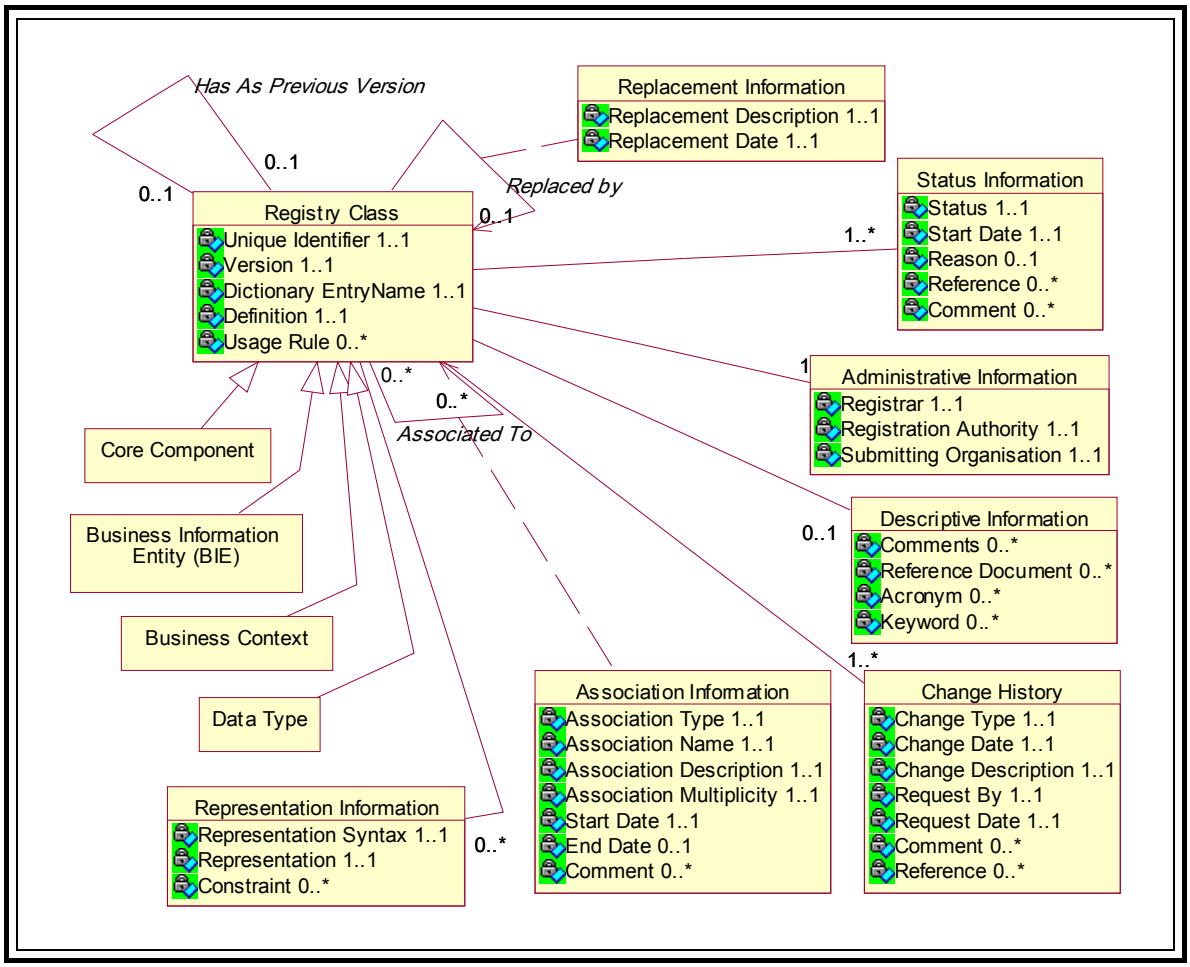
[S64] Stored *Association Business Information Entities* shall represent an *Association Business Information Entity Property* of a particular *Aggregate Business Information Entity*.

7.5 Core Component Storage Metadata

Core Components, *Data Types*, *Business Contexts* and *Business Information Entities* are used to design business documents and document components. To facilitate re-usability, it is important that these artefacts be searchable and retrievable.

Figure 7-4 focuses on the meta-information that needs to be defined for *Registry Metadata and Registry Classes* (i.e. all information needed to store for *Core Components*, *Data Types*, *Business Contexts* and *Business Information Entities*). To simplify the diagram all information regarding the structure of a *Core Component* and a *Business Information Entity* has been hidden.

Figure 7-4. Registry Metadata



As shown in Figure 7-4, the following metadata categories will be required:

- **Version Information**: even though at any given point in time only one *Version* of a *Registry Class* can be valid, multiple previous versions may have existed and a future *Version* may be in preparation. The **`Version`** association makes it possible to link the consecutive versions of a *Registry Class*. There will not be branches in the versioning; only a linear versioning will be supported.

- **Replacement Information**: a *Registry Class* may be replaced by another *Registry Class* at some point in time (e.g. because a duplicate is discovered). The **`Replaced by`** association makes it possible to do this and **`Replacement Information`** makes it possible to document the date of and reason for replacement.

- **Status Information**: information about the live status of a *Registry Class.*

- **Administrative Information**: information about the registration of the *Registry Class*.

- **Descriptive Information**: additional descriptive information about a *Registry Class*, giving further clarification about its meaning.
- **Change History**: information about all changes that are made to a *Registry Class*.
- **Association Information**: a *Registry Class* may be associated to multiple other *Registry Classes*.
- **Representation Information**: information about the physical representation of a *Registry Class* in a particular syntax (e.g. to document the XML-tag).

7.5.1 General Metadata Storage Rules

[S65] Stored *Registry Classes* shall include a *Unique Identifier*.

[S66] Stored *Registry Classes* shall include a *Version* number to keep track of the evolution over time of a *Registry Class*.

[S67] Stored *Registry Classes* shall include a *Dictionary Entry Name*.

[S68] Stored *Registry Classes* shall include a *Definition*.

[S69] Stored *Registry Classes* may include one or more *Usage Rules*, describing how and/or when to use the *Registry Class*.

[S70] Except for the first *Version* of a *Registry Class*, each stored *Version* shall be linked to its previous *Version*.

[S71] Except for the last *Version* of a *Registry Class*, each stored *Version* shall be linked to its next *Version*.

[S72] Stored *Registry Classes* shall include the history of the status lifecycle of each *Version*.

7.5.2 Management Information

7.5.2.1 Administrative Information

[S73] Stored *Registry Classes* shall contain administrative information and shall include the following *Attributes*:

- **`Registrar` (mandatory)**: Name of the responsible person who has created the *Registry Class* in the registry
- **`Registration Authority` (mandatory)**: Organisation authorised to register the *Registry Class*.
- **`Submitting Organization` (mandatory)**: The organisation that has submitted / requested the *Registry Class.*

7.5.2.2 Status Information

[S74] Stored *Registry Classes* shall contain status information to include the following *Attributes*:

- **`Status` (mandatory)**: Status of the *Registry Class* (i.e. **`draft, provisionally registered, registered, to be retired, retired`**, ...)
- **`Start Date` (mandatory)**: Date on which the status comes into effect
- **`Reason` (optional)**: Description of why the *Registry Class* status has been changed.
- **`Reference` (optional, repetitive)**: External Document(s) containing relevant information about the status change.
- **`Comment` (optional, repetitive)**: Remark about the *Registry Class* status.

7.5.2.3 Change History

[S75] Stored *Registry Classes* shall include the history of all modifications related to each *Version* to include the following *Attributes*:

- **`Change Type` (mandatory)**: Purpose of the Change—such as **`new element, new version, element modification, status modification, element replacement`**.
- **`Change Date` (mandatory)**: Date on which the modification has been made.
- **`Change Description` (mandatory)**: Description of why and how the *Registry Class* has been modified.
- **`Request By` (mandatory)**: Name of the organisation that has requested the modification of the *Registry Class.*
- **`Request Date` (mandatory)**: Date on which the modification was requested.
- **`Comment` (optional, repetitive)**: Remark about the *Registry Class* modification.
- **`Reference` (optional, repetitive)**: External Document(s) containing relevant information about the modification.

7.5.2.4 Replacement Information

[S76] For each stored pair of *Registry Classes* where one *Registry Class* replaces the other, the stored information shall specify **`Replacement Information`** to include the following *Attributes*:

- **`Replacement Description` (mandatory)**: Reason for the *Registry Class* being replaced

- **`Replacement Date` (mandatory)**: Date from which the replacement is effective.

[S77] If another *Registry Class* has replaced a *Registry Class*, it shall be linked to the *Registry Class* by which it has been replaced.

[S78] If a *Registry Class* replaces one or more other *Registry Class(es)*, it shall be linked to the *Registry Class(es)* it replaces

7.5.3 Content Information

7.5.3.1 Descriptive Information

[S79] Stored *Registry Classes* may include additional descriptive information to include the following *Attributes*:

- **`Comments` (optional, repetitive):** Comments is additional information about a *Registry Class*, which is not part of the *Definition* but that is considered relevant for clarification.

- **`Reference Document` (optional, repetitive):** Reference Document is a reference (e.g. a Uniform Resource Locator) to external documentation that contains relevant additional information about a *Registry Class*.

- **`Acronym` (optional, repetitive):** Acronym is an abbreviation or code under which the *Registry Class* is commonly known.

- **`Keyword` (optional, repetitive):** Keyword is one or more significant words used for the search and retrieval of a *Registry Class*.

7.5.3.2 Representation Information

[S80] Stored *Registry Classes* may optionally include information about the representation of the *Registry Class* in one or more syntaxes to include the following *Attributes*.

- **`Representation Syntax` (mandatory)**: Identification of the representation syntax

- **`Representation` (mandatory)**: Physical representation of the *Registry Class* (e.g. Extensible Markup Language tag)

- **`Constraint` (optional, repetitive)**: Description of additional constraints that apply to the representation of the *Registry Class* in the given syntax (e.g. **`maximum length`**)

7.5.3.3 Association Information

[S81] Stored *Registry Classes* shall include all associations they have with other stored *Registry Classes* and shall include the following *Attributes*:

- **`Association Name` (mandatory)**: Name of the association
- **`Association Description` (mandatory)**: Descriptive text explaining the meaning of the association
- **`Association Type` (mandatory)**: Type of association (e.g. aggregation, specialisation, generalization, simple association ...)
- **`Association Multiplicity` (mandatory)**: *Cardinality* of the association (i.e. optional/mandatory and repetition)
- **`Start Date` (mandatory)**: Date at which the association becomes valid
- **`End Date` (optional)**: Date from which the association is no longer valid
- **`Comment` (optional, repetitive)**: Relevant information about the association (e.g. reason why it has been removed, ...)

8 Approved Core Component Type, Content, and Supplementary Components; and Permissible Representation Terms

The following subsections contain tables that convey the currently approved *Core Component Type*s (Section 8.1), the approved *Core Component Type Content* and *Supplementary Component*s (Section 8.2), and permissible *Representation Terms* (Section 8.3).

8.1 Approved Core Component Types

Table 8-1 presents the currently approved set of *Core Component Types*.

Table 8-1 Approved Core Component Types (CCT)

CCT Dictionary Entry Name	Definition	Remarks	*Object Class*	*Property Term*	CCT Components
Amount. Type	A number of monetary units specified in a currency where the unit of currency is explicit or implied.		**Amount**	**Type**	• **Amount. Content** • **Amount Currency. Identifier** • **Amount Currency. Code List Version. Identifier**
Binary Object. Type	A set of finite-length sequences of binary octets.	Shall also be used for *Data Types* representing graphics (i.e., diagram, graph, mathematical curves or similar representations), pictures (i.e. visual representation of a person, object, or scene), sound, video, etc.	**Binary Object**	**Type**	• **Binary Object. Content** • **Binary Object. Format. Text** • **Binary Object. Mime. Code** • **Binary Object. Encoding. Code** • **Binary Object. Character Set. Code** • **Binary Object. Uniform Resource. Identifier** • **Binary Object. Filename. Text**

CCT Dictionary Entry Name	Definition	Remarks	*Object Class*	*Property Term*	CCT Components
Code. Type	A character string (letters, figures or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an *Attribute* together with relevant supplementary information.	Should not be used if the character string identifies an instance of an *Object Class* or an object in the real world, in which case the Identifier. Type should be used.	**Code**	**Type**	• **Code. Content** • **Code List. Identifier** • **Code List. Agency. Identifier** • **Code List. Agency Name. Text** • **Code List. Name. Text** • **Code List. Version. Identifier** • **Code. Name. Text** • **Language. Identifier** • **Code List. Uniform Resource. Identifier** • **Code List Scheme. Uniform Resource. Identifier**
Date Time. Type	A particular point in the progression of time together with relevant supplementary information.	Can be used for a date and/or time.	**Date Time**	**Type**	• **Date Time. Content** • **Date Time. Format. Text**
Identifier. Type	A character string to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects in the same scheme together with relevant supplementary information.		**Identifier**	**Type**	• **Identifier. Content** • **Identification Scheme. Identifier** • **Identification Scheme. Name. Text** • **Identification Scheme Agency. Identifier** • **Identification Scheme. Agency Name. Text** • **Identification Scheme. Version. Identifier** • **Identification Scheme Data. Uniform Resource. Identifier** • **Identification Scheme. Uniform Resource. Identifier**

CCT Dictionary Entry Name	Definition	Remarks	*Object Class*	*Property Term*	CCT Components
Indicator. Type	A list of two mutually exclusive Boolean values that express the only possible states of a *Property*.		**Indicator**	**Type**	• **Indicator. Content** • **Indicator. Format. Text**
Measure. Type	A numeric value determined by measuring an object along with the specified unit of measure.		**Measure**	**Type**	• **Measure. Content** • **Measure Unit. Code** • **Measure Unit. Code List Version. Identifier**
Numeric. Type	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	May or may not be decimal	**Numeric**	**Type**	• **Numeric. Content** • **Numeric. Format. Text**
Quantity. Type	A counted number of non-monetary units possibly including fractions.		**Quantity**	**Type**	• **Quantity. Content** • **Quantity. Unit. Code** • **Quantity Unit. Code List. Identifier** • **Quantity Unit. Code List Agency. Identifier** • **Quantity Unit. Code List Agency Name. Text**
Text. Type	A character string (i.e. a finite set of characters) generally in the form of words of a language.	Shall also be used for names (i.e. word or phrase that constitutes the distinctive designation of a person, place, thing or concept).	**Text**	**Type**	• **Text. Content** • **Language. Identifier** • **Language. Locale. Identifier**

8.2 Approved Core Component Type Content and Supplementary Components

Table 8-2 presents the currently approved set of *Core Component Type Content* and *Supplementary Components*.

Table 8-2. Approved Core Component Type Content and Supplementary Components

Name	Primitive data-type	Definition	Remarks
Amount. Content	decimal	A number of monetary units specified in a currency where the unit of currency is explicit or implied	
Amount Currency. Code List Version. Identifier	string	The *Version* of the UN/ECE Rec. 9 code list.	
Amount Currency. Identifier	string	The currency of the amount	Reference UN/ECE Rec. 9, using 3-letter alphabetic codes. The UN/ECE Rec. 9 is also published as ISO 4217, but is available in electronic form and free of charge.
Binary Object. Content	binary	A set of finite-length sequences of binary octets.	
Binary Object. Format. Text	string	The format of the binary content.	
Binary Object. Mime.Code	string	The mime type of the binary object.	Reference IETF RFC 2045, 2046, 2047
Binary Object. Character Set. Code	string	The character set of the binary object if the mime type is text.	Reference IETF RFC 2045, 2046, 2047
Binary Object. Encoding. Code	string	Specifies the decoding algorithm of the binary object.	Reference IETF RFC 2045, 2046, 2047
Binary Object. Uniform Resource. Identifier	string	The Uniform Resource Identifier that identifies where the Binary Object is located.	
Binary Object. Filename. Text	String	The filename of the binary object.	Reference IETF RFC 2045, 2046, 2047
Code. Content	string	A character string (letters, figures or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an *Attribute*.	
Code List. Agency. Identifier	string	An agency that maintains one or more code lists.	Defaults to the UN/EDIFACT data element 3055 code list.
Code List. Agency Name. Text	string	The name of the agency that maintains the code list.	
Code List. Name. Text	string	The name of a list of codes.	
Code List. Identifier	string	The identification of a list of codes	Can be used to identify the URL of a source that defines the set of currently approved permitted values
Code List Scheme. Uniform Resource. Identifier	string	The Uniform Resource Identifier that identifies where the code list scheme is located.	
Code List. Uniform Resource. Identifier	string	The Uniform Resource Identifier that identifies where the code list is located.	
Code List. Version. Identifier	string	The *Version* of the code list.	Identifies the *Version* of the UN/EDIFACT data element 3055 code list.

Name	Primitive data-type	Definition	Remarks
Code. Name. Text	string	The textual equivalent of the code content	If no code content exists, the code name can be used on its own
Date Time. Content	string	The particular point in the progression of time	For times use an ISO 8601 compliant format that includes the UTC offset
Date Time. Format. Text	string	The format of the date/time content	Reference ISO 8601 and W3C note on date time
Identification Scheme Agency. Identifier	string	The identification of the agency that maintains the identification scheme.	Defaults to the UN/EDIFACT data element 3055 code list.
Identification Scheme Agency. Name. Text	string	The name of the agency that maintains the identification scheme	
Identification Scheme Data. Uniform Resource. Identifier	string	The Uniform Resource Identifier that identifies where the identification scheme data is located	
Identification Scheme. Identifier	string	The identification of the identification scheme.	
Identification Scheme. Name. Text	string	The name of the identification scheme.	
Identification Scheme. Uniform Resource. Identifier	string	The Uniform Resource Identifier that identifies where the identification scheme is located.	
Identification Scheme. Version. Identifier	string	The *Version* of the identification scheme.	Identifies the *Version* of the UN/EDIFACT data element 3055 code list.
Identifier. Content	string	A character string to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects within the same scheme	
Indicator. Content	string	The value of the indicator	For example on, off, true, false
Indicator. Format. Text	String	Whether the indicator is numeric, textual or binary	
Language. Identifier	string	The identifier of the language used in the corresponding text string	Reference ISO 639: 1998
Language. Locale. Identifier	string	The identification of the locale of the language.	
Measure. Content	decimal	The numeric value determined by measuring an object.	For example, 24.387 kilograms (24.387 is the Measure. Content)
Measure Unit. Code	string	The type of unit of measure	Reference UN/ECE Rec. 20 and X12 355.
Measure Unit. Code List Version. Identifier	string	The *Version* of the measure unit code list.	
Numeric. Content	As defined by **Numeric. Format. Text**	Numeric information that is assigned or is determined by calculation, counting or sequencing.	May be decimal
Numeric. Format. Text	string	Whether the number is an integer, decimal, real number or percentage	

Name	Primitive data-type	Definition	Remarks
Quantity. Content	decimal	A counted number of non-monetary units possibly including fractions.	For example 7 bales (7 is the Quantity. Content)
Quantity. Unit. Code	string	The unit of the quantity	May use UN/ECE Rec. 20
Quantity Unit. Code List Agency. Identifier	string	The identification of the agency which maintains the quantity unit code list	
Quantity Unit. Code List. Identifier	string	The quantity unit code list.	Defaults to the UN/EDIFACT data element 3055 code list.
Quantity Unit. Code List Agency Name. Text	string	The name of the agency which maintains the quantity unit code list.	
Text. Content	string	A character string (i.e. a finite set of characters) generally in the form of words of a language.	

8.3 Permissible Representation Terms

Table 8-3 presents the set of *Permissible Representation Terms*.

Table 8-3. Permissible Representation Terms

Primary ***Representation Term***	**Definition**	**Related** ***Core Component Type***	**Secondary** ***Representation Terms***
Amount	A number of monetary units specified in a currency where the unit of currency is explicit or implied.	Amount. Type	
Binary Object	A set of finite-length sequences of binary octets. [Note: This *Representation Term* shall also be used for *Data Types* representing graphics (i.e. diagram, graph, mathematical curves, or similar representation), pictures (i.e. visual representation of a person, object, or scene), sound, video, etc.]	Binary Object. Type	Graphic, Picture, Sound, Video

Primary ***Representation Term***	**Definition**	**Related** ***Core Component Type***	**Secondary** ***Representation Terms***
Code	A character string (letters, figures or symbols) that for brevity and / or language independence may be used to represent or replace a definitive value or text of a *Property*. [Note: The term 'Code' should not be used if the character string identifies an instance of an *Object Class* or an object in the real world, in which case the *Representation Term* identifier should be used.]	**Code. Type**	
Date Time	A particular point in the progression of time (ISO 8601). [Note: This *Representation Term* shall also be used for *Data Types* only representing a Date or a Time.]	**Date Time. Type**	**Date, Time**
Identifier	A character string used to establish the identity of, and distinguish uniquely, one instance of an object within an identification scheme from all other objects within the same scheme.	**Identifier. Type**	
Indicator	A list of exactly two mutually exclusive Boolean values that express the only possible states of a *Property*. [Note: Values typically indicate a condition such as **on/off**; **true/false** etc.]	**Indicator. Type**	
Measure	A numeric value determined by measuring an object. Measures are specified with a unit of measure. The applicable unit of measure is taken from UN/ECE Rec. 20. [Note: This *Representation Term* shall also be used for measured coefficients (e.g. m/s).]	**Measure. Type**	

Primary *Representation Term*	**Definition**	**Related *Core Component Type***	**Secondary *Representation Terms***
Numeric	Numeric information that is assigned or is determined by calculation, counting or sequencing. It does not require a unit of quantity or a unit of measure. [Note: This *Representation Term* shall also be used for *Data Types* representing Ratios (i.e. rates where the two units are not included or where they are the same), Percentages, etc.)	**Numeric. Type**	**Value, Rate, Percent**
Quantity	A counted number of non-monetary units. Quantities need to be specified with a unit of quantity. [Note: This *Representation Term* shall also be used for counted coefficients (e.g. flowers/m²).]	**Quantity. Type**	
Text	A character string (i.e. a finite set of characters) generally in the form of words of a language. [Note: This *Representation Term* shall also be used for names (i.e. word or phrase that constitutes the distinctive designation of a person, place, thing or concept).]	**Text. Type**	**Name**

9 Definition of Terms

Aggregate Business Information Entity **(A**BIE**)**– A collection of related pieces of business information that together convey a distinct business meaning in a specific *Business Context*. Expressed in modelling terms, it is the representation of an *Object Class*, in a specific *Business Context*.

Aggregate Core Component **- (ACC)** – A collection of related pieces of business information that together convey a distinct business meaning, independent of any specific *Business Context*. Expressed in modelling terms, it is the representation of an *Object Class*, independent of any specific *Business Context*.

Assembly Rules - *Assembly Rules* group sets of unrefined *Business Information Entities* into larger structures. *Assembly Rules* are more fully defined and explained in the *Assembly Rules* Supplemental Document.

Association Business Information Entity (ASBIE) - A *Business Information Entity* that represents a complex business characteristic of a specific *Object Class* in a specific *Business Context*. It has a unique *Business Semantic* definition. An *Association Business Information Entity* represents an *Association Business Information Entity Property* and is therefore associated to an *Aggregate Business Information Entity*, which describes its structure. An *Association Business Information Entity* is derived from an *Association Core Component.*

Association Business Information Entity Property - A *Business Information Entity Property* for which the permissible values are expressed as a complex structure, represented by an *Aggregate Business Information Entity*.

Association Core Component (ASCC) - A *Core Component* which constitutes a complex business characteristic of a specific *Aggregate Core Component* that represents an *Object Class*. It has a unique *Business Semantic* definition. An *Association Core Component* represents an *Association Core Component Property* and is associated to an *Aggregate Core Component*, which describes its structure.

Association Core Component Property – A *Core Component Property* for which the permissible values are expressed as a complex structure, represented by an *Aggregate Core Component.*

Attribute – A named value or relationship that exists for some or all instances of some entity and is directly associated with that instance.

Basic Business Information Entity **(B**BIE**)** – A *Business Information Entity* that represents a singular business characteristic of a specific *Object Class* in a specific *Business Context*. It has a unique *Business Semantic* definition. A *Basic Business Information Entity* represents a *Basic Business Information Entity Property* and is therefore linked to a *Data Type*, which describes it values. A *Basic Business Information Entity* is derived from a *Basic Core Component*.

Basic Business Information Entity Property – A *Business Information Entity Property* for which the permissible values are expressed by simple values, represented by a *Data Type.*

Basic Core Component (BCC) –A *Core Component* which constitutes a singular business characteristic of a specific *Aggregate Core Component* that represents a *Object Class*. It has a unique *Business Semantic* definition. A *Basic Core Component* represents a *Basic Core Component Property* and is therefore of a *Data Type*, which defines its set of values. *Basic Core Components* function as the properties of *Aggregate Core Components*.

Basic Core Component (CC) Property – A *Core Component Property* for which the permissible values are expressed by simple values, represented by a *Data Type*.

Business Context – The formal description of a specific business circumstance as identified by the values of a set of *Context Categories*, allowing different business circumstances to be uniquely distinguished.

Business Information Entity (BIE) – A piece of business data or a group of pieces of business data with a unique *Business Semantic* definition. A *Business Information Entity* can be a *Basic Business Information Entity* (BBIE), an *Association Business Information Entity* (ASBIE), or an *Aggregate Business Information Entity* (ABIE).

Business Information Entity (BIE***) Property*** – A business characteristic belonging to the *Object Class* in its specific *Business Context* that is represented by an *Aggregate Business Information Entity*.

Business Libraries – A collection of approved process models specific to a line of business (e.g., shipping, insurance).

Business Process – The *Business Process* as described using the *UN/CEFACT Catalogue of Common Business Processes*.

Business Process Context – The *Business Process* name(s) as described using the *UN/CEFACT Catalogue of Common Business Processes* as extended by the user.

Business Process Role Context – The actors conducting a particular *Business Process*, as identified in the *UN/CEFACT Catalogue of Common Business Processes*.

Business Semantic(s) – A precise meaning of words from a business perspective.

Business Term – This is a synonym under which the *Core Component* or *Business Information Entity* is commonly known and used in the business. A *Core Component* or *Business Information Entity* may have several *Business Terms* or synonyms.

Cardinality – An indication whether a characteristic is optional, mandatory and/or repetitive.

Catalogue of Business Information Entities – This represents the approved set of *Business Information Entities* from which to choose when applying the *Core Component* discovery process

Catalogue of Core Components – see *Core Component Catalogue*.

CCL – see *Core Component Library.*

Child Core Component – A *Core Component* used as part of a larger aggregate construct.

Classification Scheme – This is an officially supported scheme to describe a given *Context Category*.

Constraint Language – A formal expression of actions occurring in specific *Context*s to assemble, structurally refine, and semantically qualify *Core Components*. The result of applying the *Constraint Language* to a set of *Core Components* in a specific *Context* is a set of *Business Information Entities*.

Content Component – Defines the *Primitive Type* used to express the content of a *Core Component Type*.

Content Component Restrictions – The formal definition of a format restriction that applies to the possible values of a *Content Component*.

Context – Defines the circumstances in which a *Business Process* may be used. This is specified by a set of *Context Categories* known as *Business Context.*

Context Category – A group of one or more related values used to express a characteristic of a business circumstance.

Context Rules Construct – The overall expression of a single set of rules used to apply *Context* to *Core Components*.

Controlled Vocabulary – A supplemental vocabulary used to uniquely define potentially ambiguous words or *Business Terms*. This ensures that every word within any of the *Core Component* names and definitions is used consistently, unambiguously and accurately.

Core Component (CC) – A building block for the creation of a semantically correct and meaningful information exchange package. It contains only the information pieces necessary to describe a specific concept.

Core Component Catalogue – The temporary collection of all metadata about each *Core Component* discovered during the development and initial testing of this *Core Component Technical Specification*, pending the establishment of a permanent Registry/repository.

Core Component Dictionary – An extract from the *Core Component Catalogue* that provides a ready reference of the *Core Component* through its *Dictionary Entry Name*, component parts, and definition.

Core Component Library – The *Core Component Library* is the part of the registry/repository in which *Core Components* shall be stored as *Registry Classes*. *The Core Component Library* will contain all the *Core Component Types*, *Basic Core*

Components, *Aggregate Core Components*, *Basic Business Information Entities* and *Aggregate Business Information Entities*.

Core Component Property – A business characteristic belonging to the *Object Class* represented by an *Aggregate Core Component.*

Core Component Type **(CCT)** – A *Core Component*, which consists of one and only one *Content Component*, that carries the actual content plus one or more *Supplementary Components* giving an essential extra definition to the *Content Component*. *Core Component Types* do not have *Business Semantics*.

Data Type – Defines the set of valid values that can be used for a particular *Basic Core Component Property* or *Basic Business Information Entity Property*. It is defined by specifying restrictions on the *Core Component Type* that forms the basis of the *Data Type*.

Definition – This is the unique semantic meaning of a *Core Component, Business Information Entity, Business Context* or *Data Type.*

Dictionary Entry Name – This is the unique official name of a *Core Component*, *Business Information Entity*, *Business Context* or *Data Type* in the dictionary.

Geopolitical Context – Geographic factors that influence *Business Semantics* (e.g., the structure of an address).

Industry Classification Context – Semantic influences related to the industry or industries of the trading partners (e.g., product identification schemes used in different industries).

Information Entity – A reusable semantic building block for the exchange of business-related information.

Naming Convention – The set of rules that together comprise how the *Dictionary entry Name* for *Core Components* (See Section 6.1.4.1.4) and *Business Information Entities* (See Section 6.1.4.2.4) are constructed.

Object Class – The logical data grouping (in a logical data model) to which a data element belongs (ISO11179). The *Object Class* is the part of a *Core Component*'s *Dictionary Entry Name* that represents an activity or object in a specific *Context*.

Object Class Term – A component of the name of a *Core Component* or *Business Information Entity* which represents the *Object Class* to which it belongs.

Official Constraints Context – Legal and governmental influences on semantics (e.g. hazardous materials information required by law when shipping goods).

Order – In the *Constraint Language*, the *Property* on the *ContextRules Construct* that applies a sequence to the application of a set of rules. Two Rule constructs cannot have the same value for the *Property Order*.

Primitive Type – Used for the representation of a value. Possible values are String, Decimal, Integer, Boolean, Date and Binary.

Product Classification Context – Factors influencing semantics that are the result of the goods or services being exchanged, handled, or paid for, etc. (e.g. the buying of consulting services as opposed to materials)

Property – A peculiarity common to all members of an *Object Class*.

Property Term – A semantically meaningful name for the characteristic of the *Object Class* that is represented by the *Core Component Property*. It shall serve as basis for the *Dictionary Entry Name* of the *Basic* and *Association Core Components* that represents this *Core Component Property*.

Qualifier Term – A word or group of words that help define and differentiate an item (e.g. a *Business Information Entity* or *a Data Type)* from its associated items (e.g. from a *Core Component*, a *Core Compont Type,* another *Business Information Entity* or another *Data Type*).

Registry Class – The formal definition of all the information necessary to be recorded in the Registry about a *Core Component*, a *Business Information Entity*, a *Data Type* or a *Business Context*.

Representation Term – The type of valid values for a *Basic Core Component* or *Business Information Entity*.

Supplementary Component – Gives additional meaning to the *Content Component* in the *Core Component Type*.

Supplementary Component Restrictions – The formal definition of a format restriction that applies to the possible values of a *Supplementary Component*.

Supporting Role Context – Semantic influences related to non-partner roles (e.g., data required by a third-party shipper in an order response going from seller to buyer.)

Syntax Binding – The process of expressing a *Business Information Entity* in a specific syntax.

System Capabilities Context – This *Context category* exists to capture the limitations of systems (e.g. an existing back office can only support an address in a certain form).

UMM Information Entity – A *UMM Information Entity* realizes structured business information that is exchanged by partner roles performing activities in a business transaction. Information entities include or reference other information entities through associations."

Unique Identifier – The identifier that references a *Registry Class* instance in a universally unique and unambiguous way.

Usage Rules – *Usage Rules* describe how and/or when to use the *Registry Class.*

User Community – A *User Community* is a group of practitioners, with a publicised contact address, who may define *Context* profiles relevant to their area of business. Users within the community do not create, define or manage their individual *Context* needs but conform to the community's standard. Such a community should liase

closely with other communities and with general standards-making bodies to avoid overlapping work. A community may be as small as two consenting organisations.

Version – An indication of the evolution over time of an instance of a *Core Component*, *Data Type*, *Business Context*, or *Business Information Entity*.

XML schema – A generic term used to identify the family of grammar based XML document structure validation languages to include the more formal W3C XML Schema Technical Specification, Document Type Definition, Schematron, Regular Language Description for XML (RELAX), and the OASIS RELAX NG.

10 References

- *ebXML Technical Architecture Specification* v1.04
- *ebXML Business Process Specification Schema* v1.01
- *OASIS/ebXML Registry Information Model* v2.0
- *OASIS/ebXML Registry Services Specification* v2.0
- *ebXML Requirements Specification* v1.06
- *OASIS/ebXML Collaboration-Protocol Profile and Agreement Specification* v2.0
- *OASIS/ebXML Message Service Specification* v2.0
- *ebXML Technical Report, Business Process and Business Information Analysis Overview* v1.0
- *ebXML Technical Report, Business Process Analysis Worksheets & Guidelines* v1.0
- *ebXML Technical Report, E-Commerce Patterns* v1.0
- *ebXML Technical Report, Catalog of Common Business Processes* v1.0
- *ebXML Technical Report, Core Component* Overview v1.05
- *ebXML Technical Report, Core Component* Discovery and Analysis v1.04
- *ebXML Technical Report, Context* and Re-Usability of *Core Components* v1.04
- *ebXML Technical Report,* Guide to the *Core Components* Dictionary v1.04
- *ebXML Technical Report,* Naming Convention for *Core Components* v1.04
- *ebXML Technical Report,* Document Assembly and *Context* Rules v1.04
- *ebXML Technical Report, Catalogue of Context* Categories v1.04
- *ebXML Technical Report, Core Component* Dictionary v1.04
- *ebXML Technical Report, Core Component* Structure v1.04
- *Information Technology —Metadata registries: Framework for the Specification and Standardization of Data Elements,* International Standardization Organization, ISO 11179-1
- *Information Technology —Metadata registries: Classification of Concepts for the Identification of Domains,* International Standardization Organization, ISO 11179-2
- *Information Technology —Metadata registries: Registry Metamodel,* International Standardization Organization, ISO 11179-3
- *Information Technology —Metadata registries: Rules and Guidelines for the Formulation of Data Definitions, International Standardization Organization*, ISO 11179-4
- *Information Technology —Metadata registries: Naming and Identification Principles for Data Elements, International Standardization Organization,* ISO 11179-5

- Information Technology — *Metadata registries: Framework for the Specification and Standardization of Data Elements*, International Standardization Organization, ISO 11179-6
- *Information Technologies – Open-edi Reference Model,* ISO/IEC 14662
- *Information Technologies – Business Agreement Semantic Descriptive Techniques – Part 1: Operational Aspects of Open-edi for Implementation* , ISO/IEC 15944-1
- *UN/CEFACT Modelling Methodology*, UN/CEFACT TMWG N090
- *Information Technologies – IT Enablement for Widely Used Coded Domains*, ISO/IEC 18022
- *Information Technologies – Identification and Mapping of Various Categories of Jurisdictional Domains*, ISO/IEC 18038

11 Disclaimer

The views and specification expressed in this document are those of the authors and are not necessarily those of their employers. The authors and their employers specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this design.

12 Contact Information

Team Leader

Name	Hartmut Hermes
Company	Siemens AG
Street	Suedallee 1
City, state, zip/other	85326 Munich
Nation	Germany
Phone:	+49 89 636 718 580
Email:	hartmut.hermes@siemens.com

Editor

Name	Mark Crawford
Company	Logistics Management Institute
Street	2000 Corporate Ridge
City, state, zip/other	McLean, Virginia 22102
Nation	USA
Phone:	+01 703 917 7177
Email:	mcrawford@lmi.org

Copyright Statement