INTERNATIONAL STANDARD

ISO/IEC 15476-4

First edition
2005-12-15

Information technology — CDIF semantic metamodel —

Part 4: Data models

Technologies de l'information — Métamodèle sémantique CDIF —

Partie 4: Modèles de données

Reference number
ISO/IEC 15476-4:2005(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Table of Illustrations

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 15476-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and system engineering.*

ISO/IEC 15476 consists of the following parts, under the general title *Information technology — CDIF semantic metamodel*:

— *Part 1: Foundation*

— *Part 2: Common*

— *Part 3: Data definitions*

— *Part 4: Data models*

— *Part 5: Data flow models*

— *Part 6: State/event models*

Introduction

This International Standard will assist the vendors and users of modelling tools and meta-data repositories in developing mechanisms for interchanging information. This International Standard specifies an element of a family of related International Standards. When used together, these International Standards specify a mechanism for transferring information between tools.

ISO/IEC 15474-1:2002**,** *Information technology - CDIF framework - Part 1: Overview* and ISO/IEC 15474-2:2002**,** *Information technology - CDIF framework - Part 2: Modelling and extensibility* should be read first when initially exploring CDIF. The first explains the overall CDIF architecture and how the family of standards fits together. The second explains the scope, and modelling approach in CDIF. The CDIF meta-metamodel and extensibility mechanism are also defined in that document.

This International Standard explains the Data models subject area. The CDIF semantic metamodel is used to ensure that the information held by tools communicating using CDIF is transferred with an agreed meaning. It covers the information required to express Entity-Relationship-Attribute Modelling and Logical Database Design concepts.

This International Standard has been developed with the wide support and participation of vendors, users, academia and government involved in or familiar with the CASE industry, its products and the general requirements associated with interchanging information between these products.

This document is organized into the following Clauses.

— Clause 1 to 5 are prescribed ISO/IEC Clauses;

— Clause 6: Subject area overview:

  This Clause gives an overview of the coverage of this subject area.

— Clause 7: Subject area summary:

  This Clause gives an overview of the content of this subject area.

— Clause 8: Subject area specification:

  This Clause gives the formal specification of all the objects defined in the subject area, and the formal reference to those used, but not defined in the subject area.

This document is intended to be used by anyone wishing to understand and/or use CDIF. This document provides a definition of a single subject area of the CDIF semantic metamodel. It is suitable for:

— Those evaluating CDIF;

— Those who wish to understand the principles and concepts of a CDIF transfer; and

— Those developing importers and exporters.

This document, ISO/IEC 15474-1:2002*, Information technology - CDIF framework - Part 1: Overview,* and the framework document ISO/IEC 15474-2:2002*, Information technology - CDIF framework - Part 2: Modelling and extensibility,* should be read first when initially exploring CDIF and before attempting to read other documents in the CDIF family of standards.

While there are no specific prerequisites for reading this document, it will be helpful for the reader to have familiarity with the following:

— Entity-Relationship-Attribute modelling;

— Modelling (CASE) tools;

— Information repositories;

— Data dictionaries;

— Multiple meta-layer modelling.

Information technology — CDIF semantic metamodel —

Part 4:
Data models

1 Scope

The CDIF family of International Standards is primarily designed to be used as a description of a mechanism for transferring information between modelling tools. It facilitates a successful transfer when the authors of the importing and exporting tools have nothing in common except an agreement to conform to CDIF. The language that is defined for the transfer format also has applicability as a general language for import/export from repositories. The CDIF semantic metamodel defined for CASE also has applicability as the basis of standard definitions for use in repositories.

The International Standards which form the complete family of CDIF standards are documented in ISO/IEC 15474-1. These International Standards cover the overall framework, the transfer format and the CDIF semantic metamodel.

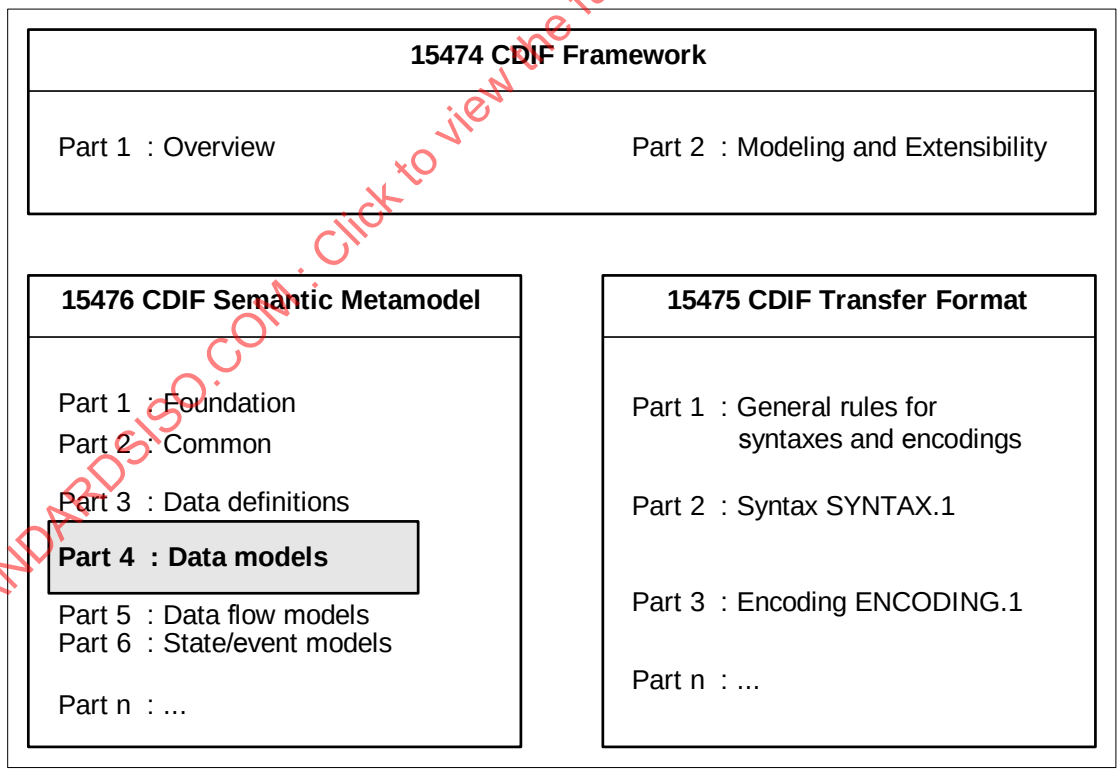

Figure 1 – CDIF family of International Standards

The diagram in Figure 1 depicts the various International Standards that comprise the CDIF family of standards. The shaded box depicts this Standard and its position in the CDIF family of standards.

This International Standard defines the portion of the CDIF semantic metamodel that supports general Data Modelling. This includes Entity-Relationship-Attribute Modelling as well as Logical Database Design.

2 Conformance

2.1 General

A product is fully standards conformant to a CDIF subject area standard if and only if it is input-conformant, output-conformant and round-trip conformant to each and every *MetaEntity*, *MetaRelationship*, *MetaAttribute*, and *AttributableMetaObject* which is defined and/or used in that standard, and it is also CDIF architecture conformant. A product may be partially input-conformant, and/or partially output-conformant, and/or partially round-trip conformant to a CDIF subject area standard.

2.2 Input conformance

Input conformance for a specific *MetaEntity*, *MetaRelationship*, *MetaAttribute*, or *AttributableMetaObject* (short: *CollectableMetaObject*) is determined by applying the following test:

A set of meta-data containing all meanings and structures standardized by a CDIF subject area is imported by the product under test. Then the meta-data which has arrived in the product is examined. The following options exist for the relation between the input (CDIF) meta-data and the imported (product) meta-data:

For a specific *CollectableMetaObject*:

1 The product is input conformant if each instance of the specific *CollectableMetaObject* has arrived in the product without change of meaning or structure. If the *CollectableMetaObject* is a meta-entity or meta-relationship, its structural relationships to other *CollectableMetaObjects* have been preserved. If the *CollectableMetaObject* is a meta-attribute, the value of the meta-attribute has been preserved.

2 The product is input morphing conformant if each instance of the specific *CollectableMetaObject* has arrived in the product, but with some changes in meaning or structure. If the *CollectableMetaObject* is a meta-attribute, the value(s) for some instances of the meta-attribute have changed.

3 The product is not input conformant for that *CollectableMetaObject* if neither of the previous tests is satisfied.

2.3 Output conformance

Output conformance for a specific *CollectableMetaObject* is determined by applying the following test:

For the product being tested, a set of meta-data that includes all possible meanings and structures representable in that product is exported. Then the meta-data that has been exported is examined. The following options exist for the relation between the product's meta-data and the exported (CDIF) meta-data:

For a specific *CollectableMetaObject*:

1 The product is output conformant if all of the meaning and structure for the specific *CollectableMetaObject* has been represented as meta-data in the product and has been exported as one or more instances of that *CollectableMetaObject*. If the *CollectableMetaObject* is a meta-attribute, the correct value of the meta-attribute has been exported.

2 The product is output morphing conformant if each instance of meta-data in the product that has the same meaning and structure as the *CollectableMetaObject* has been exported, but some instances have been exported as a different *CollectableMetaObject* or some of the meaning and structure has been changed.

3 If the product does not represent the meaning and structure associated with the *CollectableMetaObject*, output conformance for that *CollectableMetaObject* is not applicable to the product.

4 In all other cases, the product is not output conformant for that *CollectableMetaObject*.

2.4 Round-trip conformance

Round-trip conformance for a specific *CollectableMetaObject* is determined by applying the following test:

A set of meta-data containing all meanings and structures standardized by a CDIF subject area is imported by the product under test. Then the meta-data is exported again. The following options exist for the relation between the input meta-data and the output meta-data:

For a specific *CollectableMetaObject*:

1 The product is round-trip conformant if the meaning and structure of each instance of the *CollectableMetaObject* is preserved without changes during the round-trip. For a vendor to claim round-trip conformance, it is also necessary for the tool to be able to perform create, read, update, and delete operations on the imported (product) meta-data corresponding to the instances of the *CollectableMetaObject*.

2 The product is round-trip morphing conformant if each instance of the input *CollectableMetaObject* is preserved, but with some changes in meaning and/or structure. If the *CollectableMetaObject* is a meta-entity or meta-relationship, some of its instances' structural relationships to other *CollectableMetaObjects* have changed, or some instances have been transformed into other *CollectableMetaObjects*, or instances of other *CollectableMetaObjects* have been transformed into instances of the *CollectableMetaObject.* If the *CollectableMetaObject* is a meta-attribute, the values of some instances of the meta-attribute have changed or the domain of the meta-attribute has changed.

3 In all other cases, the product is not round-trip conformant for that CollectableMetaObject.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15474-1, *Information technology — CDIF framework — Part 1: Overview*

ISO/IEC 15474-2, *Information technology — CDIF framework — Part 2: Modelling and extensibility*

ISO/IEC 15476-1, *Information technology — CDIF semantic metamodel — Part 1: Foundation*

ISO/IEC 15476-2, *Information technology — CDIF semantic metamodel — Part 2: Common*

ISO/IEC 15476-3, *Information technology — CDIF semantic metamodel — Part 3: Data definitions*

4 Terms and definitions

For the purposes of this document, the following definitions apply. Unless otherwise noted, the definitions are specific to this International Standard.

4.1 From other International Standards

4.1.1 ISO/IEC 15474-1

This part of ISO/IEC 15476 makes use of the following terms defined in ISO/IEC 15474-1:

CDIF
CDIF family of standards
CDIF semantic metamodel
CDIF meta-metamodel
CDIF transfer
Instance
Meta-attribute
Meta-entity
Metamodel
Meta-object
Meta-relationship
Model
Subject area
Transfer
Transfer format

4.1.2 ISO/IEC 13238-1

This part of ISO/IEC 15476 makes use of the following terms from ISO/IEC 13238-1:

Exporter
Importer

4.1.3 For this International Standard

For the purpose of this part of ISO/IEC 15476 new terms are defined when introduced. Double quotes are used to introduce new terms (e.g., "model layer")

5 Symbols (and abbreviated terms)

5.1 Naming, diagramming and definition conventions

Conventions for naming, diagramming, describing and defining meta-objects can be found in Clause 7 of the framework document (ISO/IEC 15474-2:2002*, Information technology - CDIF framework - Part 2: Modelling and extensibility*).

5.2 Abbreviations

The following abbreviation is used in this International Standard:

CDIF CASE Data Interchange Format (originally)

5.3 Notation for this subject area

The instance diagrams appearing in Clause 6 use various techniques to depict relationships. Relationships are shown as lines and arrow-heads, or as diamonds. Keys and key components are shown as in Figure 2.

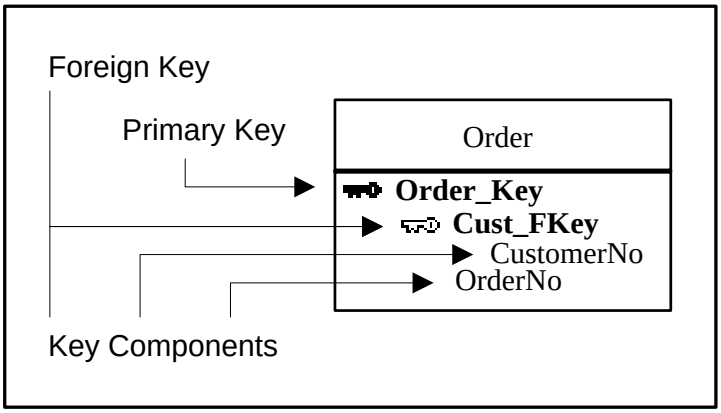

Figure 2 – Diagrammatic Conventions - Keys

Solid keys are primary keys; clear keys are foreign keys and indented items are key components. For example, Figure 2 reads, "***Orders*** have a primary key called ***Order_Key*** that consists of a foreign key called ***Cust_FKey*** and an attribute ***OrderNo***. ***Cust_FKey*** consists of a single attribute called ***CustomerNo***."

6 Data models subject area overview

6.1 Introduction

The Data Models subject area addresses the major forms of entity-relationship-attribute modelling and logical database design. This subject area is technique independent but offers broad coverage, striving for a superset of the concepts represented in various techniques.

The following sub-clauses explain the major concepts supported in this subject area.

6.2 Diagrams

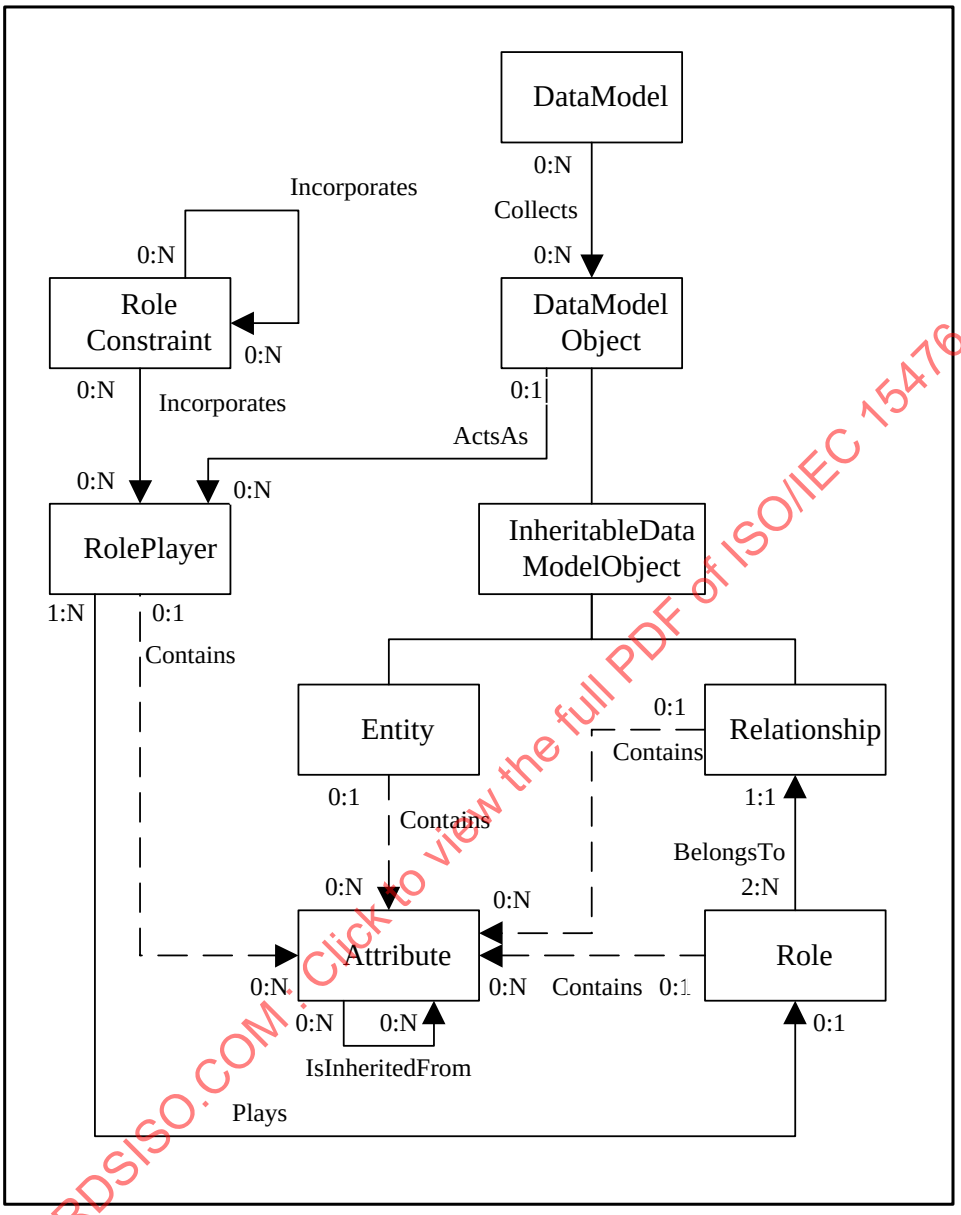

Figure 3 – Data Models subject area overview

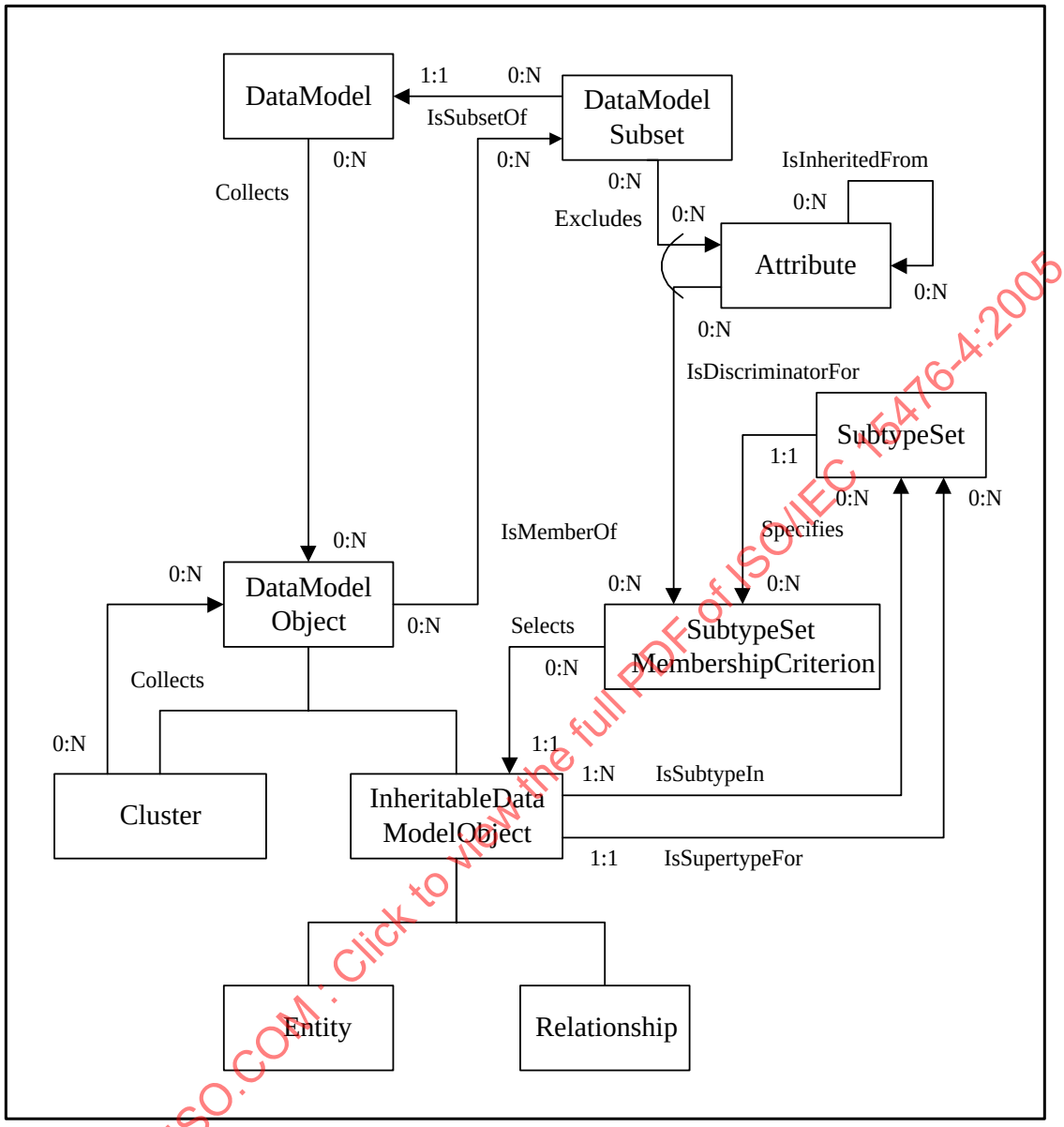

Figure 4 – Partitioning/View mechanisms

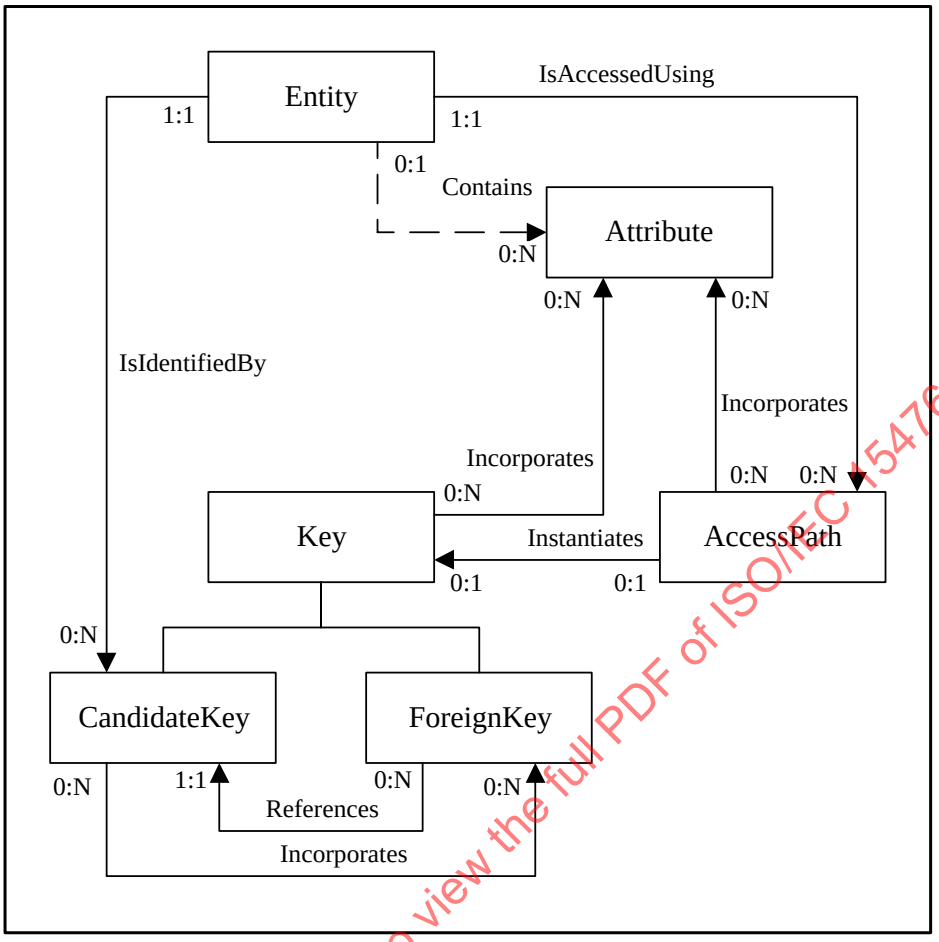

Figure 5 – *Keys* and *AccessPath*

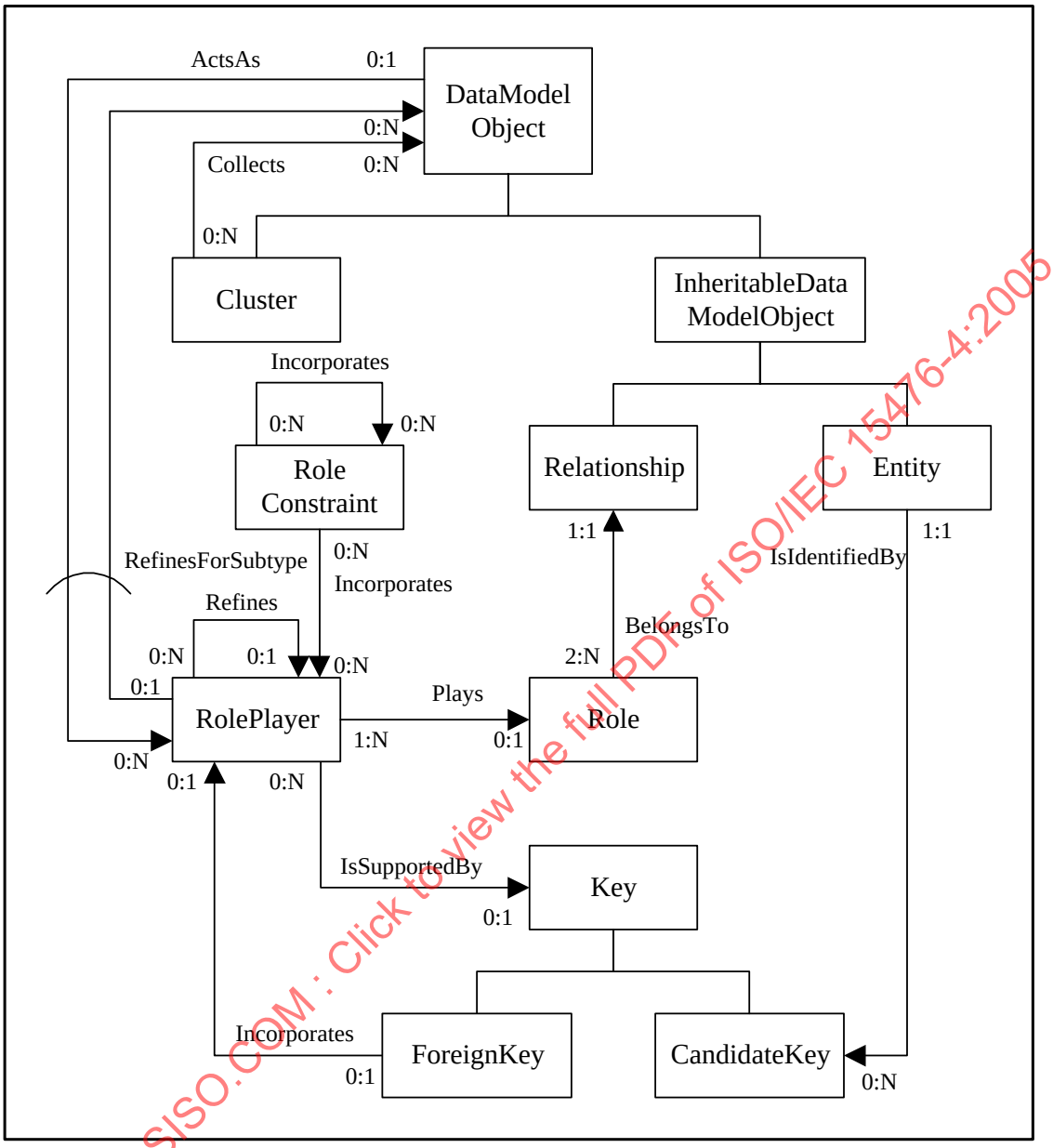

Figure 6 – *Roles*, *RolePlayers* and *RoleConstraints*

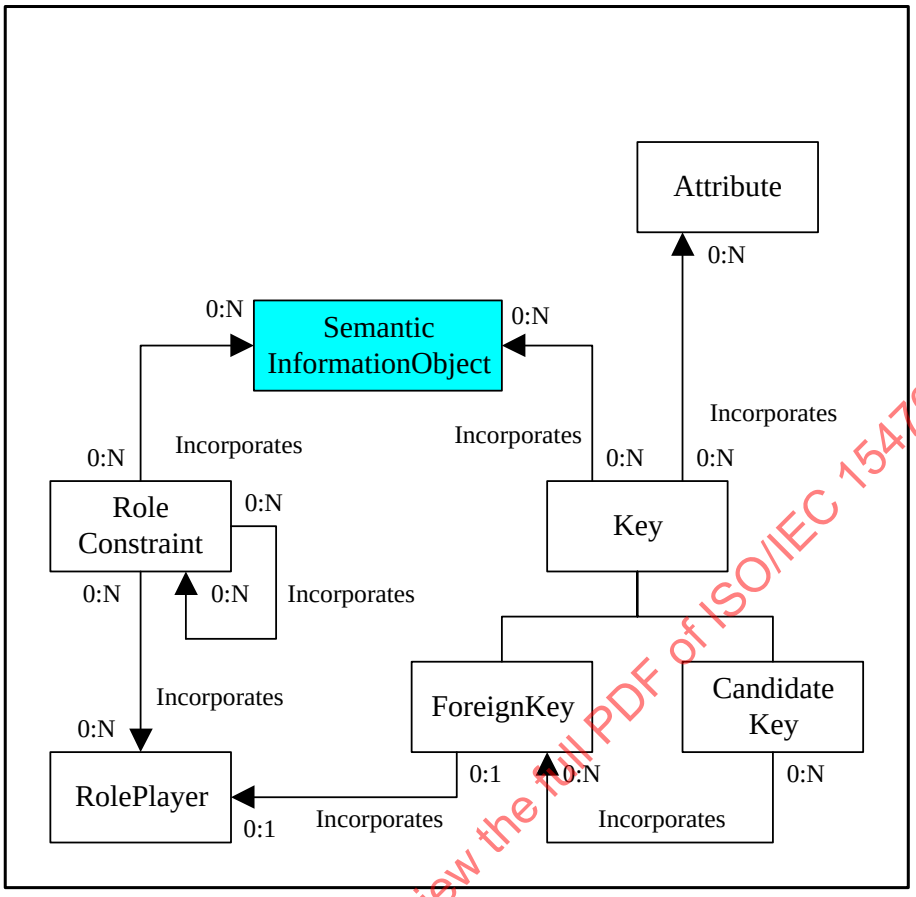

Figure 7 – Connection between Data Models and Common subject areas

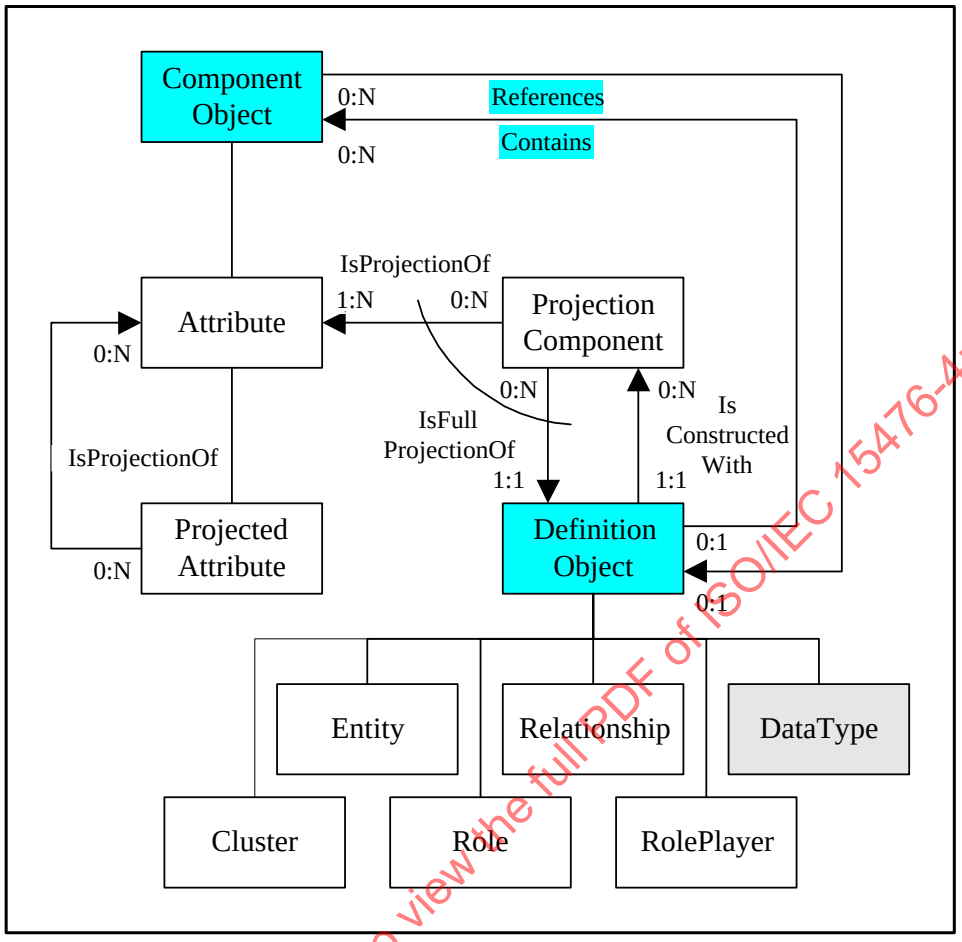

Figure 8 – Connection between Data Models and Data Definition subject areas

6.3 Entities

The concept of an "Entity," or object to be modeled, is represented directly by the meta-entity called *Entity*. A relational "Table" is also represented by *Entity*. A Table is a more concrete representation of an *Entity*.

6.4 Relationships

A relationship is an association between two or more *DataModelObjects*, expressed by the meta-entity *Relationship*. There are two orthogonal classifications of relationship, each with two subtypes:

- Arity expresses the degree or number of participants in a relationship. The two types are:
    - Binary
    - N-ary
- Intricacy categorizes the complexity of relationship association. The two types are:
    - Simple
    - Complex

6.4.1 Binary relationship

This is an association between two *DataModelObjects*. Figure 9 depicts a binary relationship between ***Orders*** and ***LineItems*** and reads, "**Order** contains **LineItem**."

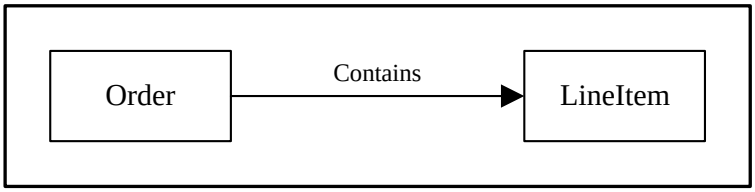

Figure 9 – Binary Relationship

6.4.2 N-ary relationship

This is an association between three or more *DataModelObjects*. Figure 10 depicts a ternary relationship which reads "An ***Employee*** is ***Recruited*** by an ***Employer*** using a ***Recruiter***."

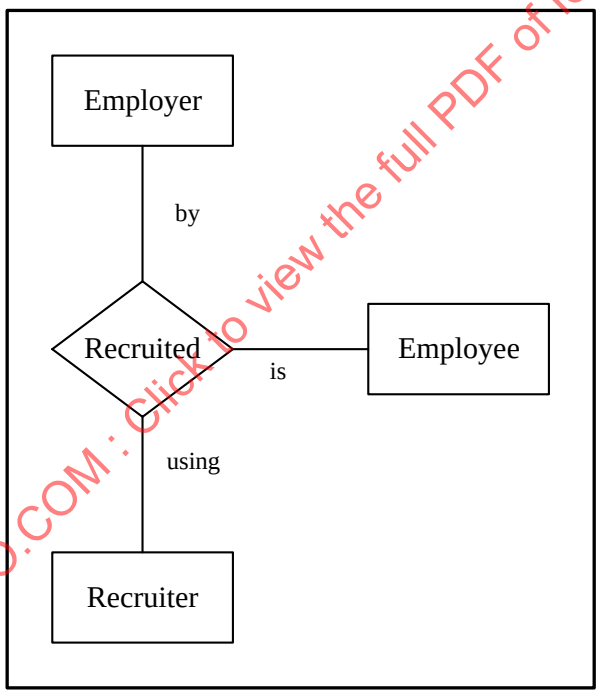

Figure 10 – Ternary Relationship

6.4.3 Simple relationship

This is an association between *Entities* or between *Clusters* or between *Entities* and *Clusters*. Figure 9 is an example of a Simple Binary Relationship and Figure 10 is an example of a Simple N-ary Relationship.

6.4.4 Complex relationship

This is an association between a *Relationship* and one or more *DataModelObjects*. Figure 11 depicts the association of an entity called ***Student*** and a binary relationship called ***Class*** using another binary relationship called ***Assigned***.

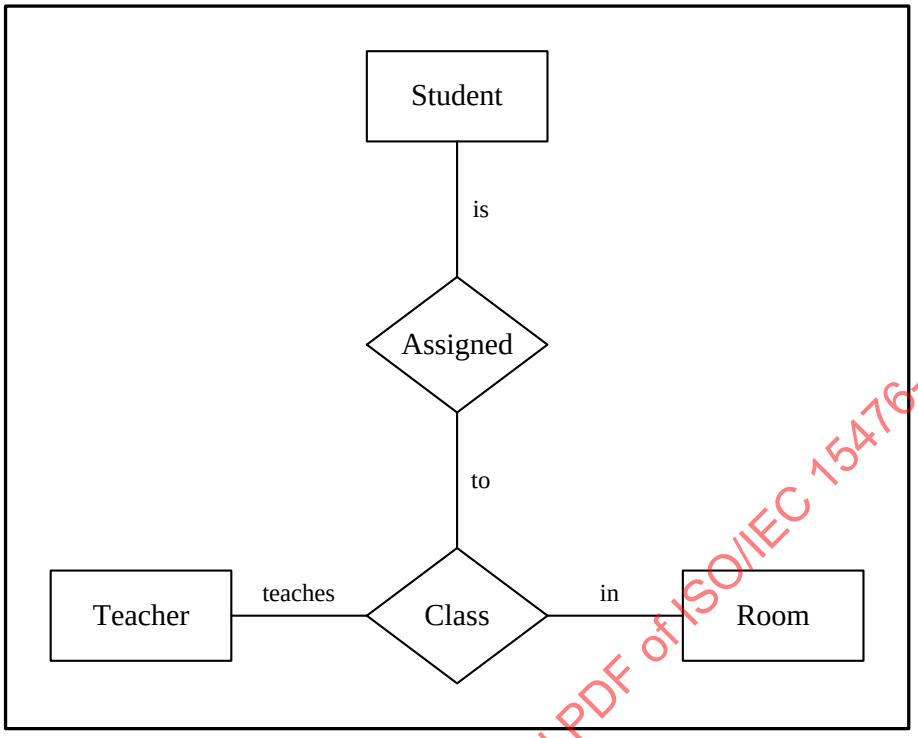

Figure 11 – Complex Binary Relationship

Figure 12 is an instance diagram for Figure 11, illustrating this concept. ***Captain Bligh*** is a student assigned to the class taught by the teacher ***Fletcher Christian*** in the room named ***Bounty***.

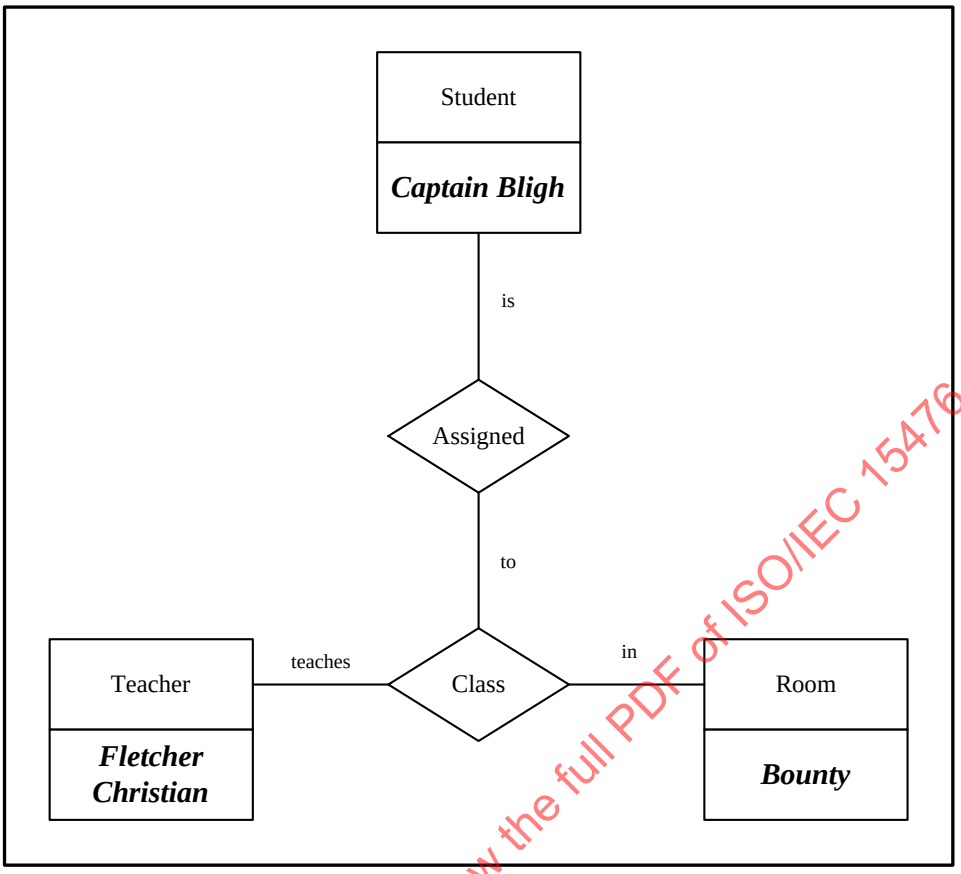

Figure 12 – Instance Diagram showing a Complex Relationship

6.5 Roles

A "Role" is the expression of an object playing a part in a relationship. The degree or arity of the relationship is a representation of the number of roles in the relationship. In a binary relationship there are two roles (Figure 13); in a ternary relationship there are three roles (Figure 10); and in an n-ary relationship there are "n" roles. In Figure 13, ***Order*** plays the role of ***set*** and ***LineItem*** plays the role of ***member***.

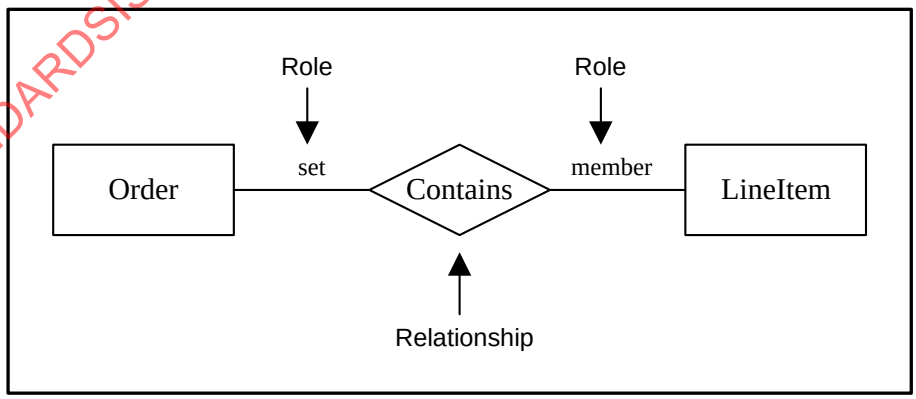

Figure 13 – Roles

The object playing a role in the relationship is represented by *DataModelObject*. The *DataModelObject* can be another *Relationship*, an *Entity* or a *Cluster*. A relationship must have at least two roles to be meaningful (a binary relationship). Incomplete relationship definitions are not supported.

The same *DataModelObject* can play both (binary) or many (n-ary) *Roles* in a *Relationship* as depicted in Figure 14. This is called a "Reflexive" or "Recursive" relationship. In this example, the relationship being modeled is that of a ***Person*** managing another ***Person***. However, each ***Person*** instance plays only one role with respect to another ***Person*** instance - either taking the role of ***Manager*** or that of ***Subordinate***.

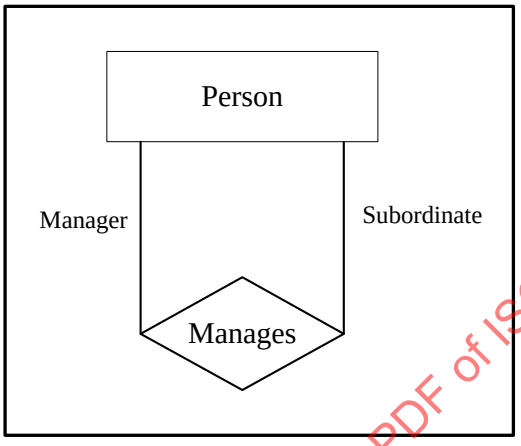

Figure 14 – Reflexive or Recursive Relationship

6.6 Cardinalities

Cardinalities are used to define static limits on the number of *DataModelObject* instances that can be associated using a *Relationship*. The limits are generally expressed as a range in the form "m:x", where "m" expresses the minimum number of object instances, and "x" expresses the maximum number of object instances. These pairings are visually associated with the object at each end of the relationship. For example, "0:N" expresses a range where the minimum participation is zero (optional participation) to a maximum value of "N" (unlimited). The combined cardinality of all participating objects expresses the type of relationship as illustrated in Figure 15.

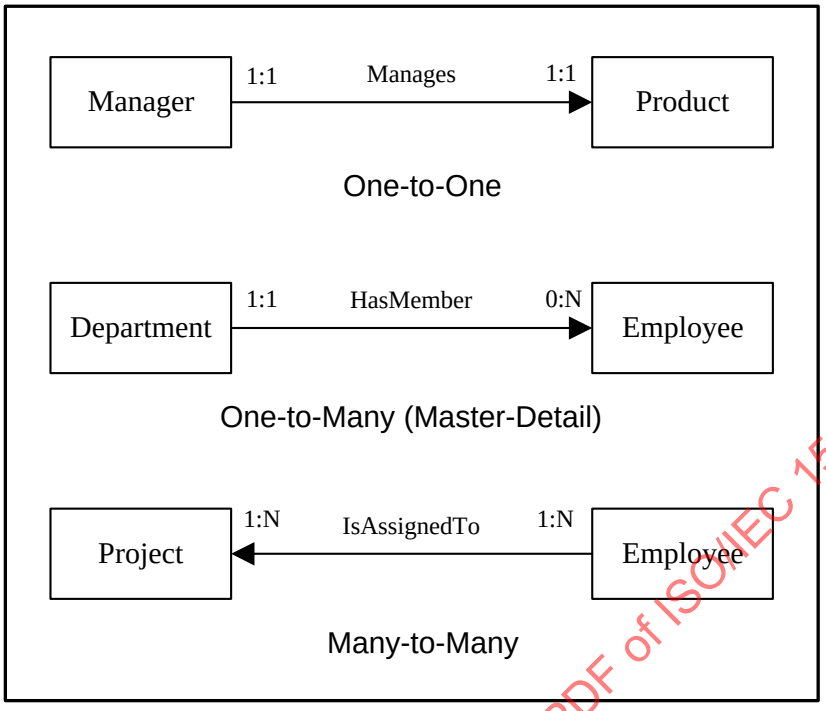

Figure 15 – Examples of Types of Binary Relationships

The Data Model Subject Area differentiates between inner and outer cardinality values for *DataModelObjects* participating in relationships. These concepts are described in the next two sub-clauses.

6.6.1 Outer cardinalities

The "Outer" cardinality defines the number of allowed instances of a participating *DataModelObject* from the viewpoint of the other participants in the *Relationship*. In the example in Figure 16, the outer cardinalities define the minimum and maximum number of instances of both the *Entity* called ***LineItem*** and the *Entity* called ***Order*** for the relationship called ***Contains***. This example reads: "For a given ***Order*** there may be at minimum zero ***LineItems*** and at maximum an unlimited number of ***LineItems*** (0:N). For a given ***LineItem*** there may be one and only one ***Order*** (1:1)." It is assumed that for each ***LineItem*** there is a corresponding instance of the relationship ***Contains.***

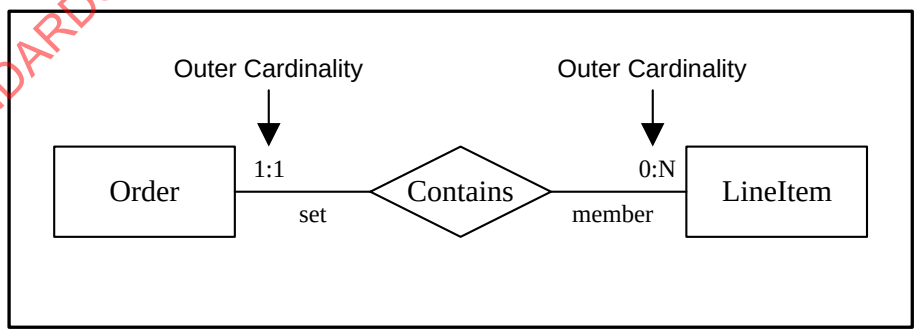

Figure 16 – Outer Cardinalities

6.6.2 Inner cardinalities

"Inner" cardinality defines the number of allowed instances of the *Relationship* from the viewpoint of a single instance of the *DataModelObject* playing a *Role*. In the context of un-attributed binary relationships, with which most people are familiar, the issue, of how many relationship instances are allowed, does not arise. It is assumed that a single instance of the relationship exists for each pair of related *DataModelObjects*. If outer cardinalities are not provided they are assumed to be ***1:1***. However, in more complex Modelling, when both inner and outer cardinality values are used, the concept is important. In the example in Figure 17, the model reads "One or more ***Bonuses*** may be given by one ***Employer*** to one ***Employee***. We can see that each ***Bonus*** may be given only once and that ***Employers*** and ***Employees*** may participate in many instances of ***Given***."

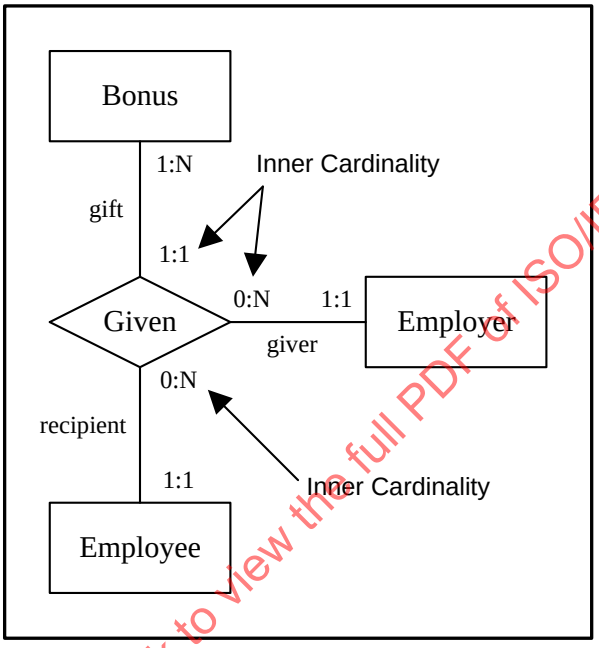

Figure 17 – Inner Cardinalities

Even where attributed relationships (either binary or n-ary) are used, a physical implementation would provide multiple instances of the relationship. But in conceptual terms, one must be able to define whether one or many relationship instances are required for any situation, since there are situations in which each is applicable.

Figure 18 depicts the concept of relationship instances. A given relationship has either one instance or multiple instances and each has its uses. Inner cardinality specifies which case is being used. In this example we have two depictions of the relationship between an ***Order*** and its ***LineItems***. The left hand side depicts a model that allows multiple relationship instances and the right, a single instance.

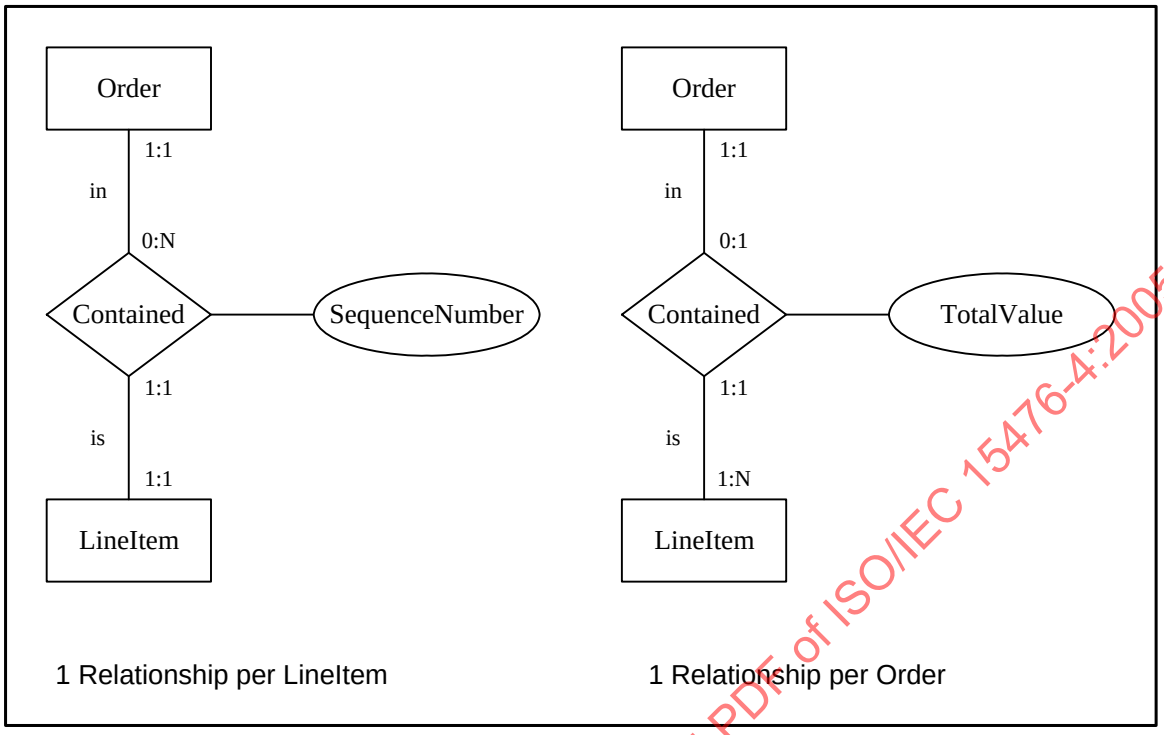

Figure 18 – Inner Cardinalities and Attributed Binary Relationships

In the case of the left hand side of Figure 18, the relationship holds an attribute called ***SequenceNumber*** to order the ***Lineitems***. The inner cardinality for ***Order*** is ***0:N*** and the outer cardinality for ***LineItems*** of 1:1, giving one instance of the relationship for each ***LineItem***.

In the right hand side the ***TotalValue*** attribute on the relationship shall occur only once for the ***Order***, rather than once per ***LineItem*** so there shall only be one instance of the relationship for each ***Order***. The ***TotalValue*** attribute can be considered as a property of the relationship between the two entities, rather than a property of the ***Order***, since its value is dependent on the ***LineItem***, rather than solely on the ***Order*** itself. Thus the inner cardinality from the viewpoint of an ***Order*** instance is 0:1, and the outer cardinality for ***LineItem*** is 1:N, giving only one instance of the relationship for each ***Order***, and thus only one instance of ***TotalValue***.

In both cases the inner cardinality for ***LineItem*** is ***1:1*** as each ***LineItem*** can only relate to a single ***Order***.

In the absence of inner cardinality values, the assumed semantic is multiple relationship instances. In the absence of outer cardinality values, the assumed semantic is mandatory participation of exactly one instance from each of the associated *DataModelObjects*.

6.7 Multi-player roles

For each object participating in a relationship there shall be a *Role*. However, several different types of object may play the same *Role* in a *Relationship*. This is called a multi-player *Role* and it is modeled using the meta-entity *RolePlayer*. It is this mechanism that enables heterogeneous collections to be modeled. For example, in Figure 19 an ***Office*** is furnished with a ***Desk***, ***Chair***, ***BookCase*** and ***FilingCabinet***. Note the specification of outer cardinalities on each *RolePlayer*, as each may have a different participation. There shall be only one ***Desk*** in an ***Office*** but one or more ***Chairs*** and zero or more ***BookCases*** and ***FilingCabinets***.

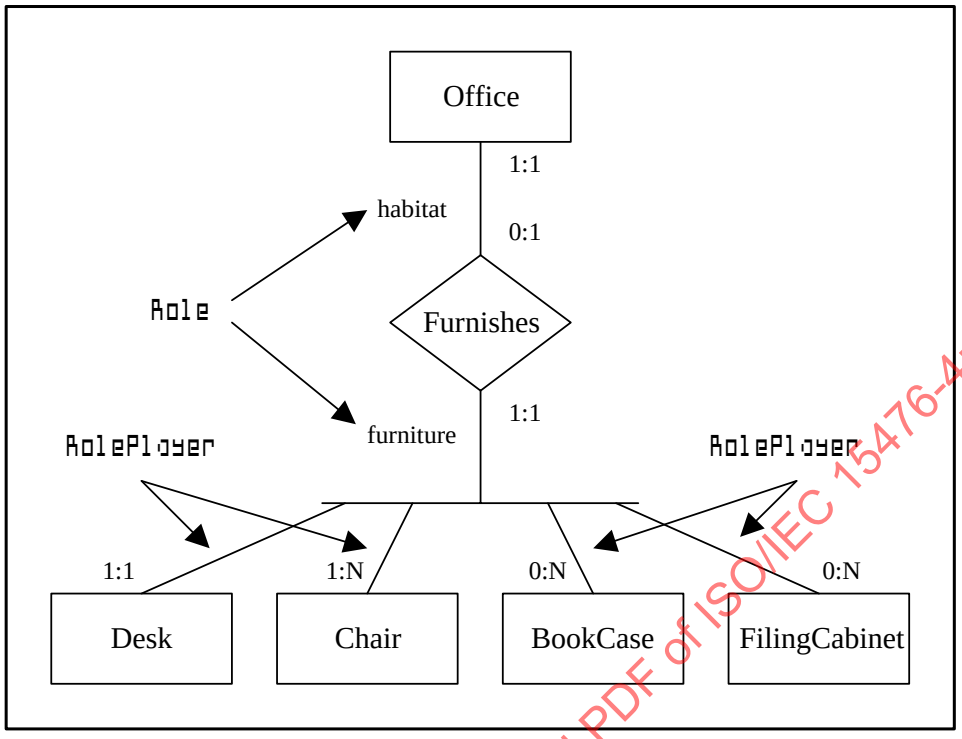

Figure 19 – Multi-player Roles and Heterogeneous Collections

In contrast, in Figure 20 the *Role* ***recruitingAgent*** specifies the cardinalities of the *RolePlayers* ***Recruiter*** and ***Employee***. Note that this is merely a graphical convention; cardinality values are properties of the *RolePlayer* meta-entity, and thus in this case the *RolePlayers* called ***serviceProvider*** and ***reference*** both contain the same cardinality values.

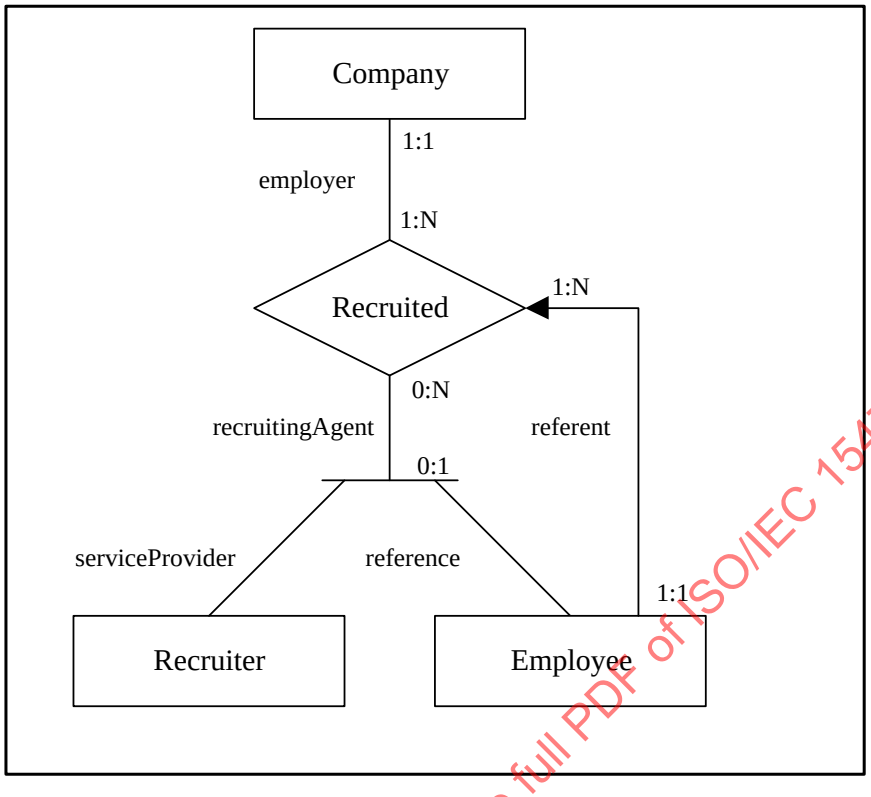

Figure 20 – Multi-player Roles and Common Outer Cardinality

All relationships use *RolePlayer*, even for simple binary relationships. The *RolePlayer* meta-entity contains meta-attributes that describe cardinalities and volumetrics concerning the participating object. The volumetric meta-attributes specify the number of insertions, deletions, updates and reads over a given time period that are expected for the participating object.

6.8 Role constraints

RoleConstraints provide a mechanism for expressing complex permissible groupings of relationships. This constrains the participation of an instance of a *DataModelObject* (*Entity, Relationship* or *Cluster*) in a set of relationships. The meta-attribute *Operator* in the meta-entity *RoleConstraint* expresses the constraint, whose values are the boolean operators inclusive ***OR***, exclusive ***XOR*** and ***AND***. The groupings are expressed on the *RolePlayers* which make up the relationships, rather than the relationships themselves, as multiple *DataModelObjects* may play a single role. The meta-relationship *RoleConstraint.Incorporates.RolePlayer* allows constraint groups to be constructed. Note that a role may participate in more than one constraint group. Figure 21 illustrates these concepts.

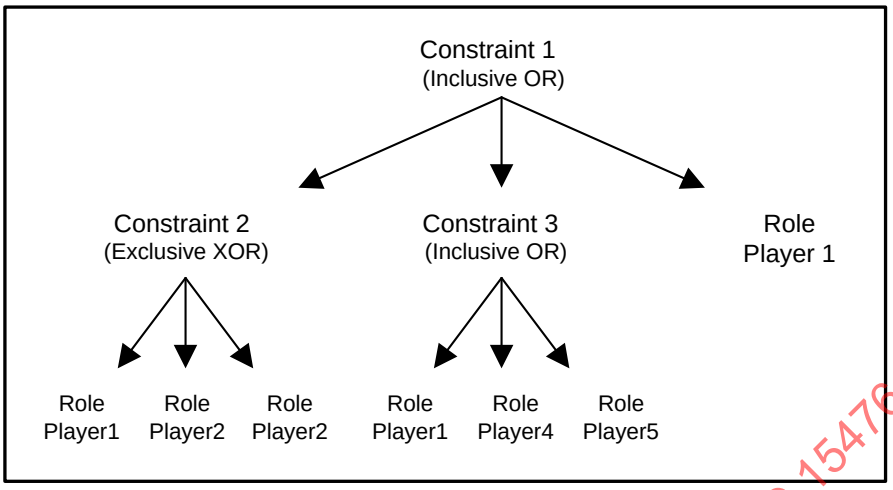

Figure 21 – Role Constraint Network

A single object can act as more than one *RolePlayer* in a *Relationship* with a need to constrain the participation of its instances to only one *Role*. Figure 22 illustrates this point. In this example, ***Employee*** is constrained to being either a player in the role ***recruitingAgent*** or a player in the role ***referent***. This example models employee recruitment and is read "An ***Employee*** participates in one or more acts of recruitment. For each recruitment, there is one ***Company***, and optionally, either a single ***Recruiter*** or ***Employee*** is used. ***Recruiters*** and ***Employees*** can be used in zero or more acts of recruitment. An ***Employee*** is constrained from recruiting him/herself."

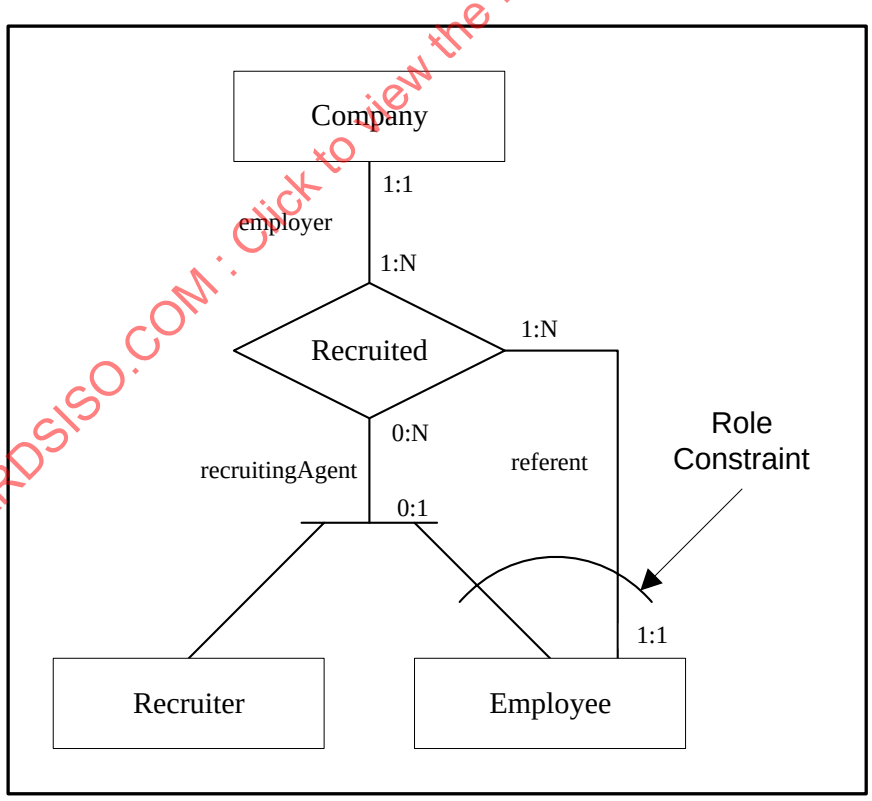

Figure 22 – Constraints on Role Participation

Figure 23 is an instance diagram of Figure 22, illustrating the concept of *Role* constraints.

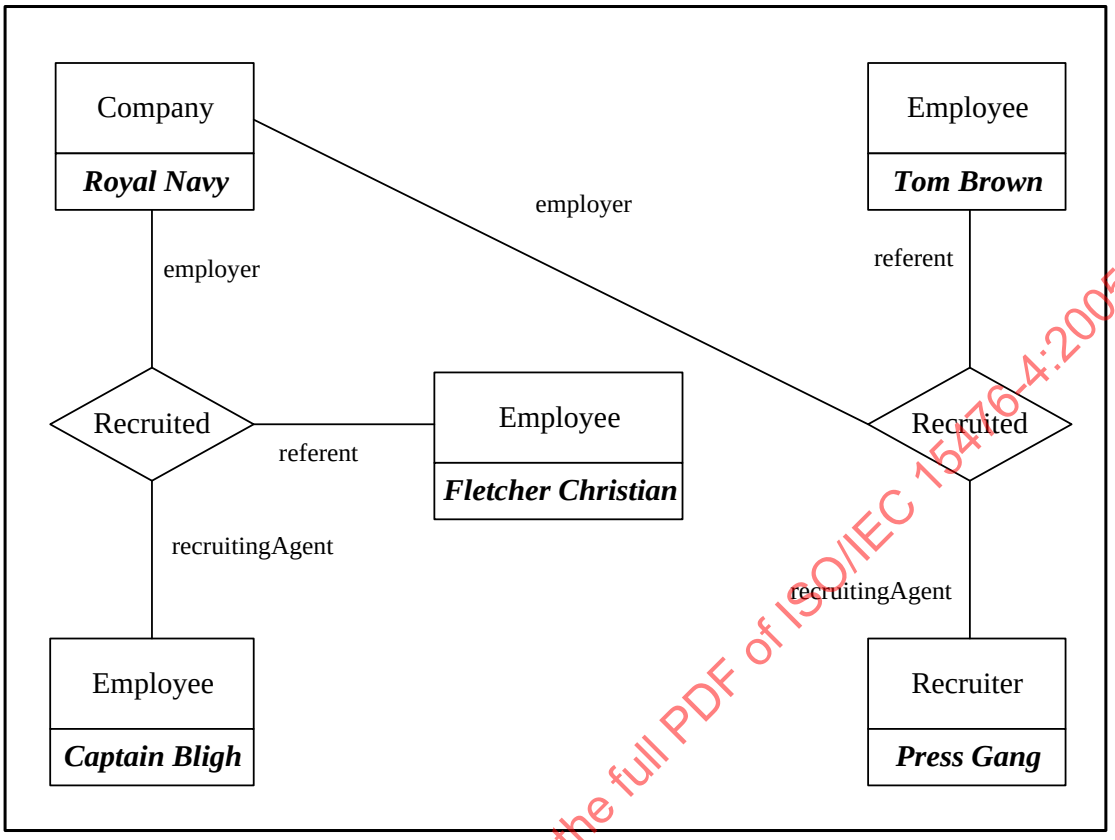

Figure 23 – Instance Diagram showing Role Constraints

6.9 Attributes

In this subject area the data content of *DefinitionObjects* is described by the use of *Attribute* or *ProjectedAttribute* (which is a subtype of *Attribute)*. These are assigned to the subtypes of *DefinitionObjects* through the meta-relationship *DefinitionObject.Contains.ComponentObject*. For example, an *Entity* called ***Customer*** contains attributes ***Name***, ***Address*** and ***PhoneNumber***.

ProjectedAttributes are dependent attributes that exist as a result of a projection operation such as computation. For example, the *ProjectedAttribute* called ***Total*** exists as a result of multiplying the *Attributes* called ***Price*** and ***Quantity***. For a detailed discussion on projection see Sub-clause Projections.

The detailed type information about attributes is not covered by this subject area; the Data Definition subject area is used to describe this. The connections between the Data Models subject area and the Data Definition (see *ISO/IEC DIS 15476-3:2000, Information Technology - CDIF Semantic Meta-model - Part 3: Data Definition*) subject area is shown in Figure 8.

Since the Data Definition subject area has its own decomposition mechanism of a DataType into its subtypes, an Entity should be modelled as it only contains top-level Attributes when the Data Models subject area is used along with the Data Definition subject area.

Roles, RolePlayers, and *Relationships* may also have attributes defined for them using the meta-relationship *DefinitionObject.Contains.ComponentObject*. Figure 24 illustrates the use of each of these. This models how an ***Employee*** is recruited by a ***Company***. The *Relationship* called ***Recruited*** has an *Attribute* of ***ApprovedBy***. Sometimes the services of a ***Recruiter*** are used and sometimes an existing ***Employee*** refers someone. In the case of the former the *RolePlayer* called ***serviceProvider*** is paid a ***Fee*** and the *RolePlayer* called ***reference*** is paid a ***Bonus***.

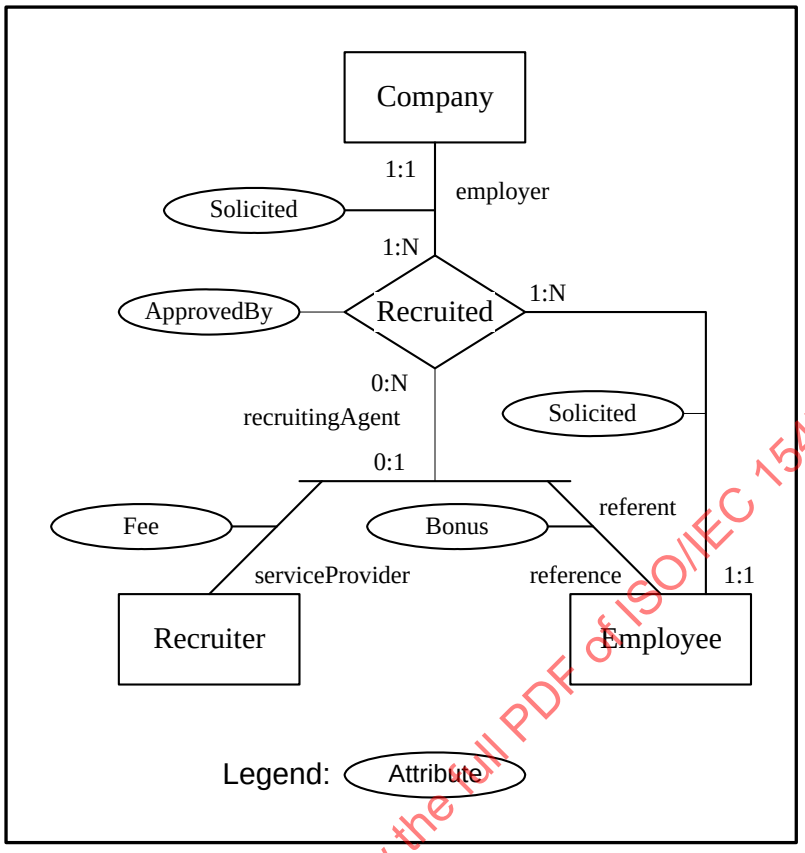

Figure 24 – Attributed *Relationships*, *Roles* and *RolePlayers*

6.10 Keys and access paths

Entity instances are identified by *Key* values and are accessed using *AccessPath* values. A *Key* is a means of identification of *Entity* instances, providing the basis of normalization and referential integrity (see 6.11). *Keys* are used to support relationships. *AccessPaths,* sometimes known as Alternate Keys, are used to specify efficient access to *Entity* instances and are often implementations of *Keys* (using the meta-relationships *Entity.IsAccessedUsing.AccessPath* and *AccessPath.Instantiates.Key*).

Keys are classified into two types:

a) Candidate Keys

CandidateKeys are a means of identification of the containing Entity. The concatenated values of *CandidateKey* components contain unique values and thus uniquely identify *Entity* instances. A primary key is simply one of the set of candidate keys for a given *Entity* and signifies that this is the key that is the major identifier. This concept of unique identification is the basis on which *Entity* normalization takes place. *CandidateKeys* used in conjunction with *ForeignKeys* provide the basis of referential integrity, which is the mechanism for maintaining consistency between related *Entities* after Delete, Update or Insert operations (See 6.11).

b) Foreign Keys

ForeignKeys are a replication of a *CandidateKey* from a related *Entity*. The *ForeignKey's* purpose is to provide a referential integrity constraint to ensure that any *ForeignKey* values have matching *CandidateKey* values affecting creation, modification and deletion of *Entity* instances. It is also used to support the concept of a relationship. The *ForeignKey* refers to the *CandidateKey* by using the meta-relationship called

ForeignKey.References.CandidateKey. For example, from Figure 25, in the relationship ***Customer.Places.Order*** if there is a *ForeignKey* value in ***Order*** that does not have an equivalent *CandidateKey* value in ***Customer*** then there is an integrity failure.

Keys contain one or more *Attributes* drawn from the set that describes the related *Entity*. Multiple attributes comprising a key characterize the key as a concatenated key. Note that *CandidateKeys* may also contain *ForeignKeys* as in the example in Figure 25. ***Order_Key*** consists of both a foreign key, ***Cust_FKey***, and an attribute ***OrderNo***. The foreign key is ***Cust_FKey*** which contains a single attribute ***CustomerNo***. This is modeled through the meta-relationship *CandidateKey.Incorporates.ForeignKey.* This implies that the attributes that describe the foreign keys are the same attributes (and attribute values) as those describing the associated candidate key, and thus the attributes are "global" in scope. However, *Attributes* are local in their ability to describe a *DefinitionObject* and thus *ForeignKeys* must contain *Attributes* that describe the entity that contains the *ForeignKey*. The association between a *CandidateKey's* attributes and those of its *ForeignKeys* is accomplished by defining for each *CandidateKey* attribute a *DataType* that is shared between the *CandidateKey* attribute and the corresponding *ForeignKey* attribute. Additionally, the sequence of attributes that comprise the *ForeignKey* must be the same sequence that comprise the *CandidateKey*.

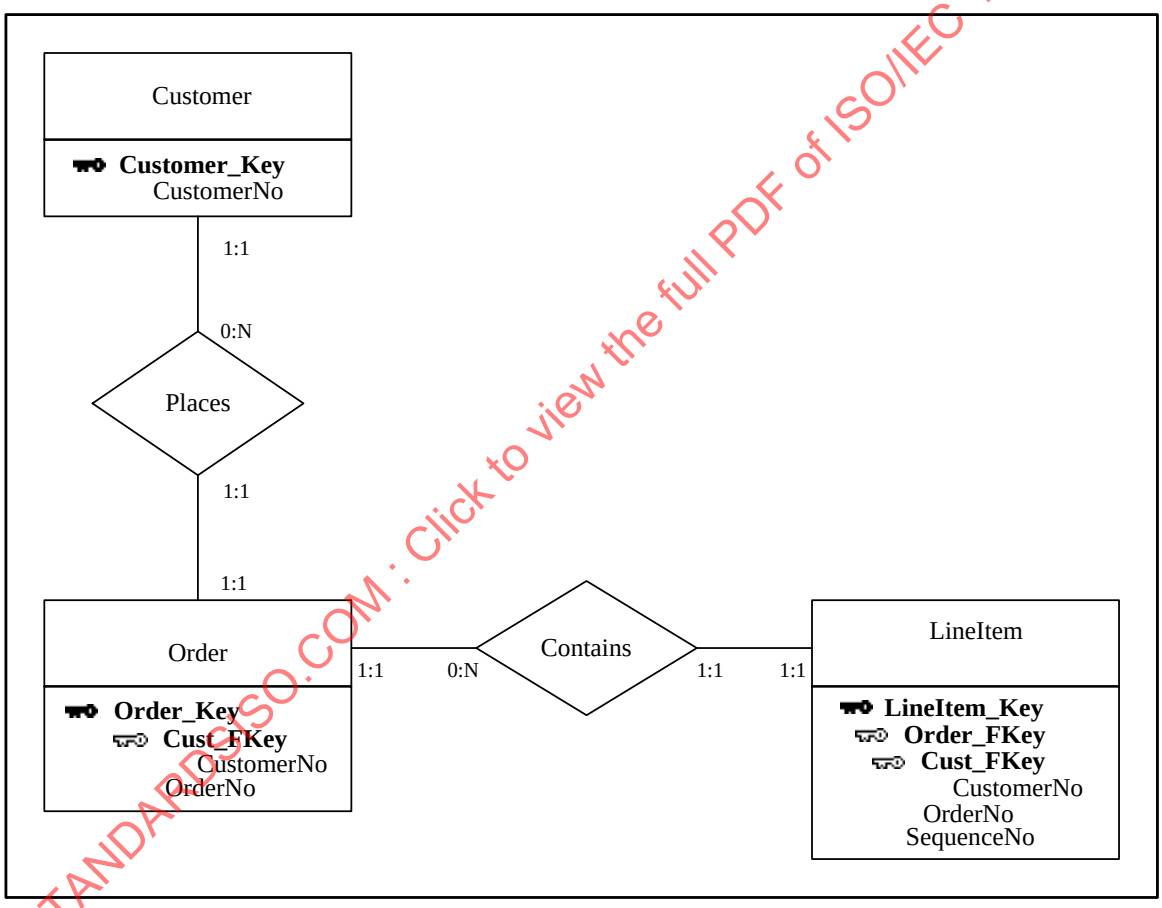

Figure 25 – Foreign Keys

In some techniques a relationship can be a component of a *ForeignKey*. This is modeled by relating the relevant *RolePlayer* to the *ForeignKey* using *ForeignKey.Incorporates.RolePlayer*. In this technique attributes describing the key are not specified - they are implied from the candidate key; the foreign key is simply composed of the *RolePlayer* reference. Figure 26 illustrates this where ***Employee_FKey*** in ***Software*** is described by the *RolePlayer* ***version*** and ***Employee_FKey*** in ***Hardware*** is described by the *RolePlayer* ***model***. This means that both ***Hardware*** and ***Software*** are not described by the attribute ***Name*** that is the description of the primary key in ***Employee***.

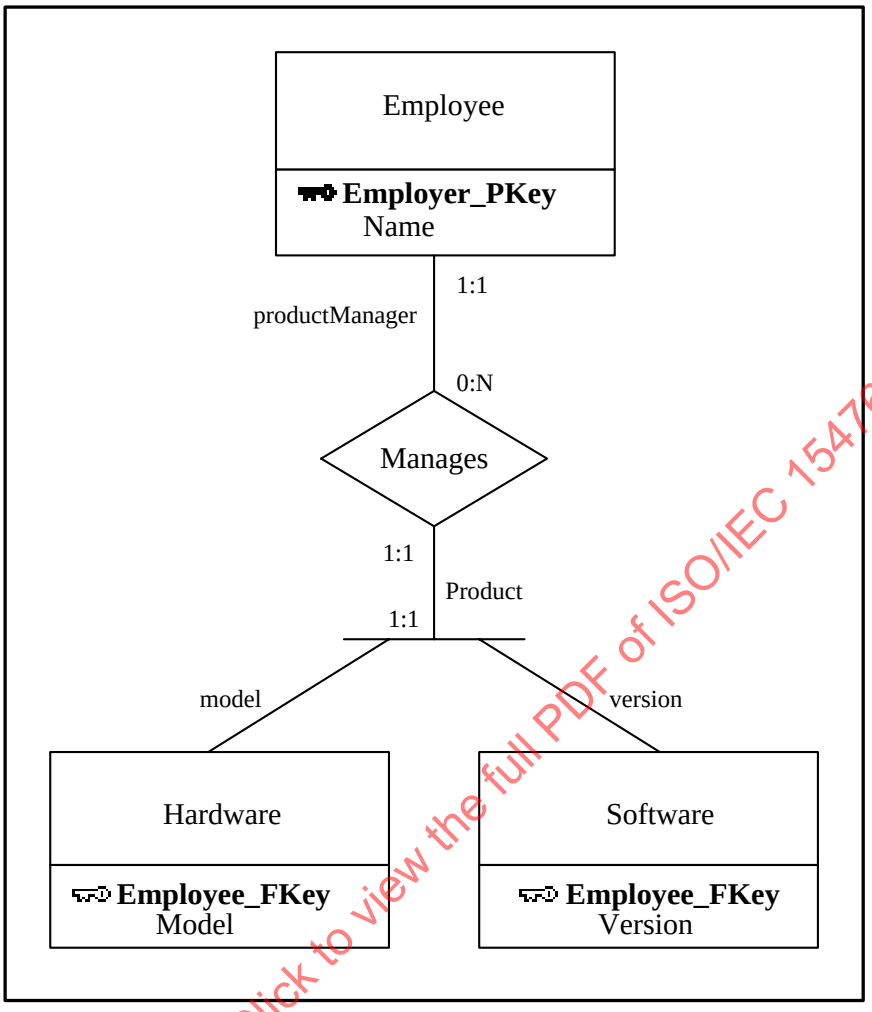

Figure 26 – *RolePlayer* as a component of a *ForeignKey*

Pairings of *CandidateKeys* and *ForeignKeys* are used to more concretely express the *Relationships* defined between *Entities*. The meta-relationship *RolePlayer.IsSupportedBy.Key* is used as the mechanism to express this, by selecting a particular *CandidateKey* (usually the primary key) or *ForeignKey* from an entity to be used in support of the entity's participation as a *RolePlayer* in a particular *Relationship*. Where more than one *Entity* can play the same *Role* in a *Relationship*, there is one *Key* for each *RolePlayer* in the *Role*. Figure 27 illustrates this concept, where ***Recruiter*** and ***Employee*** act as *RolePlayers*, playing the role of ***recruitingAgent*** in the relationship. In this case the primary key ***Recruiter_PKey*** supports the *RolePlayer* ***serviceProvider*** and ***Employee_PKey*** supports the *RolePlayer* ***reference***.

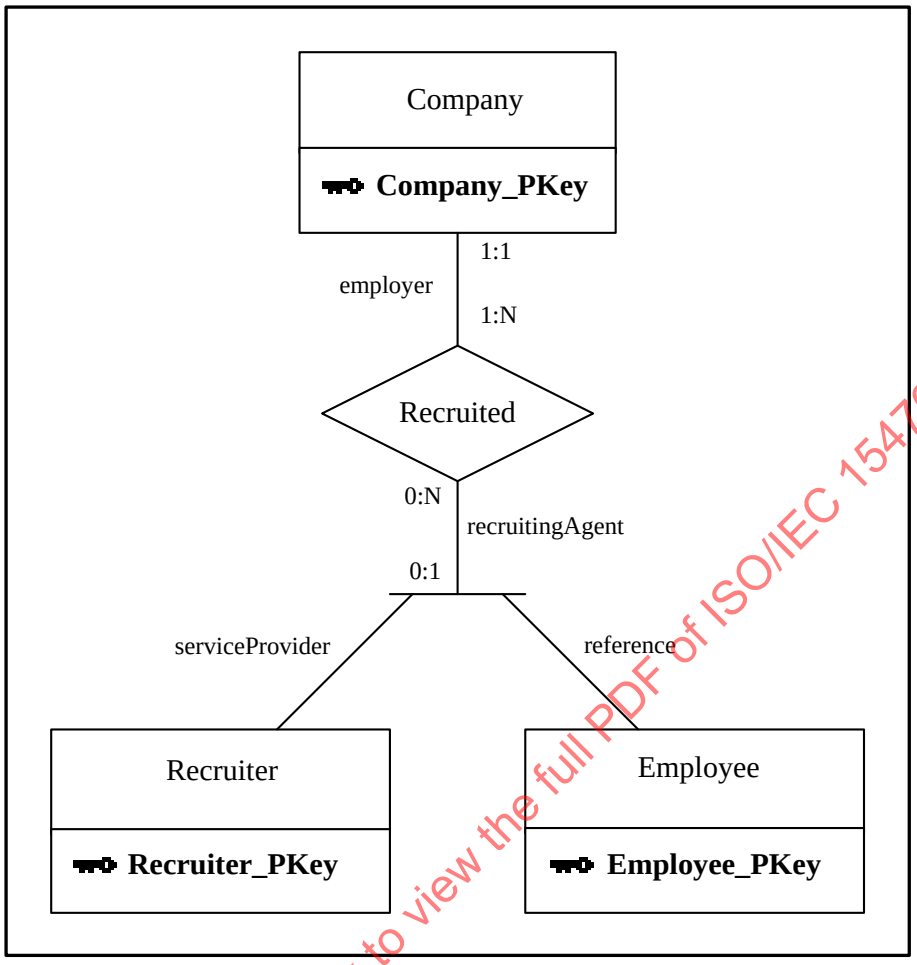

Figure 27 – *Key* support of *RolePlayer*

The same *DataModelObject* can play both (binary) or many (n-ary) *Roles* in a reflexive or recursive *Relationship* as depicted in Figure 14. In this example, the relationship being modeled is that of a ***Person*** managing another ***Person***. However, each ***Person*** instance plays only one role with respect to another ***Person*** instance - either taking the role of ***Manager*** or that of ***Subordinate***. The meta-relationship *RolePlayer.IsSupportedBy.Key* identifies which role is played by which instance - the *CandidateKey* ***Person_PKey*** would play the ***Manager*** role and the *ForeignKey* ***Manager_Key*** would play the ***Subordinate*** role.

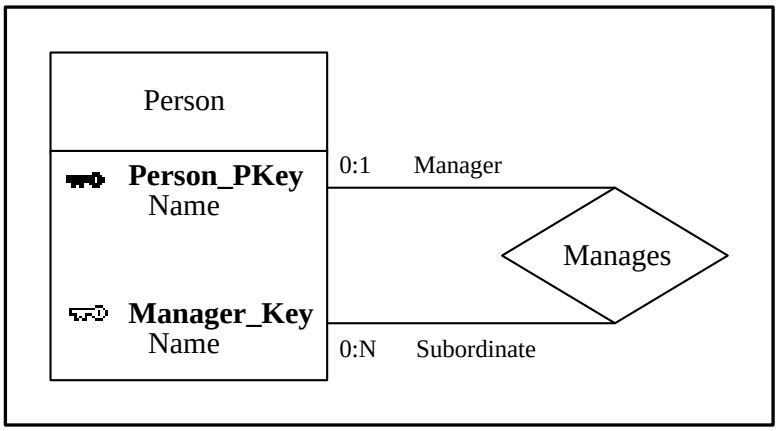

Figure 28 – *Key* support for *RolePlayer* in a Reflexive *Relationship*

6.11 Referential integrity

Keys are used to enforce referential integrity. Referential integrity rules are associated with *RolePlayer* as the mechanism for binding *CandidateKey* and *ForeignKey* pairs using the meta-relationship *RolePlayer.IsSupportedBy.Key*.

The meta-attributes *InsertEffect*, *DeleteEffect*, and *UpdateEffect*, belonging to *RolePlayer*, govern the behavior associated with Insert, Delete, and Update events on an *Entity* type. If the referenced *Entity* instance is inserted, deleted, or its *Key* is updated, then the rules outlined in Table 1 apply for each type of key when each event occurs:

Table 1 Referential Integrity Rules

Key	Insert	Delete	Update
Candidate	RESTRICTS CASCADES	RESTRICTS CASCADES SETNULL SETDEFAULT	RESTRICTS CASCADES SETNULL SETDEFAULT
Foreign	RESTRICTS CASCADES	RESTRICTS CASCADES	RESTRICTS CASCADES

CandidateKey.InsertEffect

- ***RESTRICTS***

  The insert operation cannot complete if there are no matching *ForeignKey* entity instances. This occurs where the *ForeignKey* entity's participation in the relationship is mandatory. For example, ***Employee.Has.WorkSpace*** (0:1 to 1:1), where an instance of *Employee* is inserted.

- ***CASCADES***

  An instance of the *ForeignKey* entity shall be inserted. This is the equivalent of deferred checking of integrity. This applies to a mandatory relationship. For example, ***ProductManager.IsResponsibleFor.Product*** (0:N to 1:N), where an instance of ***ProductManager*** is inserted.

ForeignKey InsertEffect

- ***RESTRICTS***

  The insert operation cannot complete if there are no matching *CandidateKey* entity instances. This occurs where the *CandidateKey* entity's participation in the relationship is mandatory. For example, ***Order.Has.LineItem*** (1:1 to 0:N).

- ***CASCADES***

  An instance of the *CandidateKey* entity shall be inserted if none exists. This is the equivalent of deferred checking of integrity. This applies to a mandatory relationship. For example, ***Customer.Places.Order*** (1:1 to 0:N).

CandidateKey DeleteEffect

- ***RESTRICTS***

  The delete operation cannot complete if there are matching *ForeignKey* entity instances, This occurs where the *CandidateKey* entity's participation in the relationship is mandatory. For example, ***Customer.Places.Order*** (1:1 to 0:N).

  This is the equivalent of "NO ACTION" in SQL2.

- ***CASCADES***

  The delete operation will delete *ForeignKey* entity instances and this will cascade in a "domino" effect for entity instances that have the candidate key's *DeleteEffect* value of ***CASCADES***, until there are no more *ForeignKey* entity instances or the *ForeignKey* entity has a *DeleteEffect* value of ***RESTRICTS***. For example, in Figure 25 ***Order*** contains a foreign key, ***Cust_FKey***, that references ***Customer*** and ***LineItem*** contains a foreign key, ***Order_FKey***, that references ***Order***. Deleting a ***Customer*** instance will cause a cascading delete that will delete all referencing ***Order*** instances which will cause all referencing ***LineItem*** instances to be deleted (the *DeleteEffect* meta-attribute of the ***Customer*** *RolePlayer* and the *DeleteEffect* meta-attribute for the ***Order*** *RolePlayer* have values of ***CASCADES***).

- ***SETNULL***

  The delete operation will cause the values of referencing *ForeignKeys* to be set to NULL. This preserves the *ForeignKey* entity instances. For example, in a model where we have the relationship ***Supplier.Fulfills.Order*** (0:N to 0:N) and the ***Supplier*** instance is deleted, the ***Order*** instances have their *ForeignKeys* set to NULL.

- ***SETDEFAULT***

  The delete operation will cause the values of referencing *ForeignKeys* to be set to default values as specified by the meta-attribute *DefaultValue*. This preserves the *ForeignKey Entity* instances. For example, in a model where we have the relationship ***Supplier.Fulfills.Order*** and the ***Supplier*** instance is updated or deleted, the ***Order*** instances have their *ForeignKeys* set to a default value of ***UNKNOWN***.

ForeignKey DeleteEffect

- ***RESTRICTS***

  The delete operation cannot complete if the number of foreign key entity instances falls below the minimum cardinality value. For example, in the relationship ***Employee.WorksFor.Manager*** (1:N to 1:1) where a given ***Manager*** has only one ***Employee*** assigned, the delete operation cannot complete.

- ***CASCADES***

The delete operation will delete referenced *CandidateKey* entity instances when the last *ForeignKey* entity instance has been deleted. For example, for the relationship ***LineItem.IsContainedIn.Order*** (0:N to 1:1), when the last ***LineItem*** instance is deleted for a particular ***Order*** instance, the ***Order*** instance is deleted.

CandidateKey UpdateEffect

- ***RESTRICTS***

  The update operation cannot complete if there are matching *ForeignKey* entity instances.

- ***CASCADES***

  The update operation will update *ForeignKey* entity instances. For example, for the relationship ***Customer.Places.Order***, if the ***Customer's*** candidate key is updated then all associated ***Order*** instances will have their *ForeignKey* values correspondingly updated.

- ***SETNULL***

  The update operation will cause the values of referencing *ForeignKeys* to be set to NULL. This effectively breaks the relationship between the candidate key entity instance and the *ForeignKey* entity instances.

- ***SETDEFAULT***

  The update operation will cause the values of referencing *ForeignKeys* to be set to default values as specified by the meta-attribute *DefaultValue*.

ForeignKey UpdateEffect

- ***RESTRICTS***

  The update operation cannot complete if there are matching *CandidateKey* entity instances.

- ***CASCADES***

  The update operation will update the attribute components in the *CandidateKey* entity instances.

The meta-attributes *IsDeleteDeferred*, *IsInsertDeferred*, and *IsUpdateDeferred* of *RolePlayer* govern whether referential integrity checking is to be done immediately, or deferred to the completion of the transaction.

6.12 Supertypes and subtypes

A supertype is a generalization of one or more object types into a single more abstract object type. A subtype is a refinement of its supertype (specialization). In this Subject Area, only *InheritableDataModelObjects* have supertypes or subtypes. Subtypes are represented by the *SubtypeSet* meta-entity and the meta-relationship *InheritableDataModelObject.IsSubtypeIn.SubtypeSet*. The supertype is identified by an instance of the meta-relationship *InheritableDataModelObject.IsSupertypeFor.SubtypeSet*. Subtypes are grouped into one or more sets.

Each subtype set can be determined by an attribute or attributes of the supertype that acts as a discriminator. This is represented by the meta-relationship *Attribute.IsDiscriminatorFor.SubtypeSetMembershipCriterion.* Each value of the discriminator determines the appropriate subtype. This is represented by the meta-attribute *DiscriminatorValue* of the meta-entity *SubtypeSetMembershipCriterion* and the meta-relationship *SubtypeSetMembershipCriterion.Selects.InheritableDataModelObject*. For example, in Figure 29 the subtype set that includes ***Male*** and ***Female*** subtypes is discriminated using the ***Gender*** attribute, whose values are ***M*** that determines ***Male*** and ***F*** that determines ***Female***.

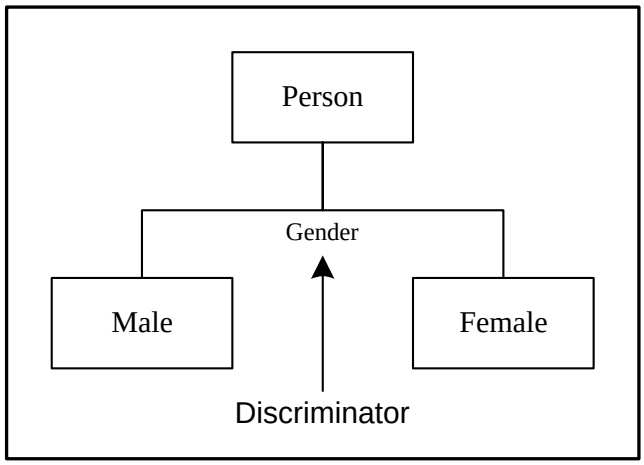

Figure 29 – Subtype Set Discriminator

There are four important characteristics of subtype sets:

- Coverage
- Exclusivity
- Orthogonality
- Inheritance

6.12.1 Coverage

Coverage is the concept that describes the instantiability of the supertype. When an *InheritableDataModelObject* serves as a supertype for one or more subtype sets, and it can be instantiated, it is said to be concrete. If it cannot be instantiated, it is said to be abstract. An abstract supertype requires that the subtypes be instantiated rather than the supertype.

Coverage is specified by the meta-attribute *IsAbstract* of the *InheritableDataModelObject* meta-entity. A value of ***TRUE*** indicates an abstract supertype, ***FALSE*** indicates a concrete supertype. For example, in Figure 30, any instance of ***Person*** is either ***Male*** or ***Female*** (***Person*** is abstract as indicated by italics). In Figure 31, the ***Employee*** supertype is concrete, hence, there can be instances of ***Employee*** that are neither ***Waged*** nor ***Salaried***. Note that in the case where there are multiple subtype sets, the status of the supertype (abstract or concrete) applies equally to all subtype sets - i.e. coverage for a supertype must be consistent for all of its subtype sets.

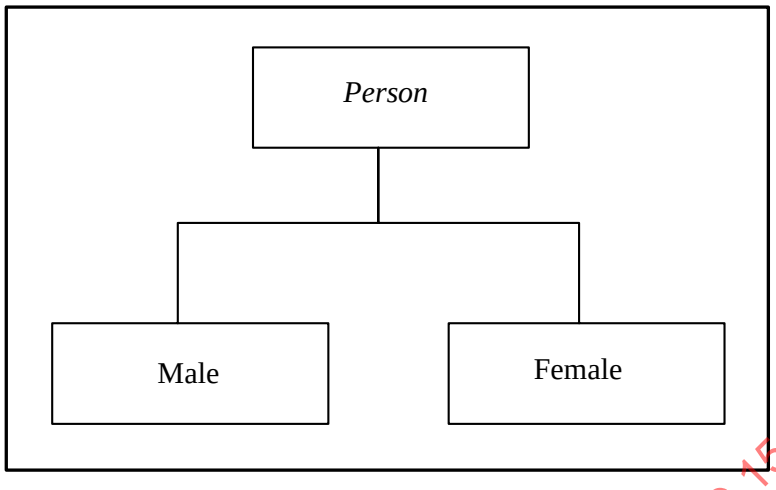

Figure 30 – Abstract Supertype

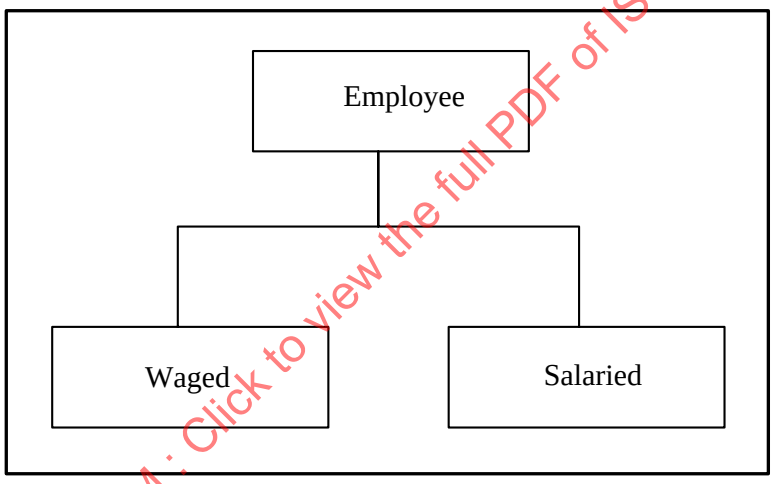

Figure 31 – Concrete Supertype

Abstract supertypes are sometimes used as categorization supertypes and may have little or no attributes.

6.12.2 Exclusivity

Exclusivity is the concept that subtype instances do not overlap. If the subtype set is not exclusive then an instance can exist as more than one subtype. For example, in Figure 30, a given instance of ***Person*** can be either ***Male*** or ***Female***. In contrast, in Figure 32, an instance of ***CDIFMember*** can be ***Chair***, ***ViceChair***, ***WGChair*** and ***Editor*** concurrently. The *IsExclusive* meta-attribute of the *SubtypeSet* meta-entity is used to specify exclusivity among the subtypes of the subtype set.

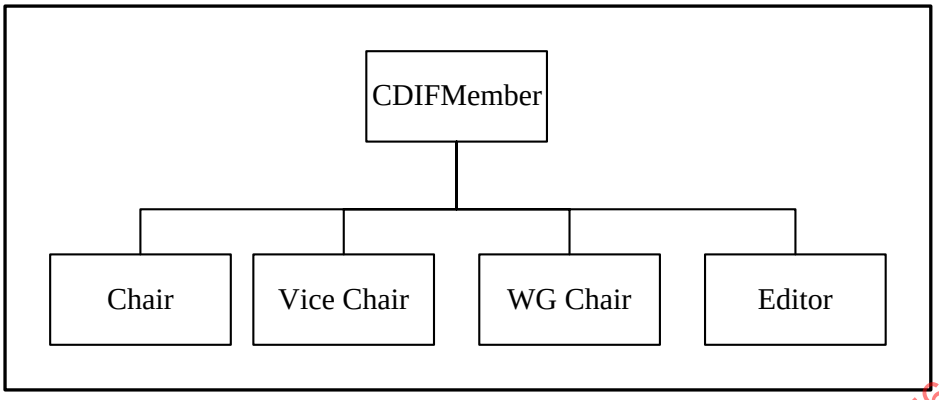

Figure 32 – Non-exclusive Subtype Set

6.12.3 Orthogonality

Multiple subtype sets with the same supertype are said to be orthogonal (or parallel). Orthogonality allows for multiple ways of specializing a supertype entity. For example, Figure 33 illustrates subtype orthogonality. The ***Employee*** supertype can be subtyped as ***FullTime*** or ***PartTime*** and also subtyped as ***Salaried*** or ***Waged***. Both specializations apply simultaneously.

Thus the combinations are:

FullTime-Salaried
PartTime-Salaried
FullTime-Waged
PartTime-Waged

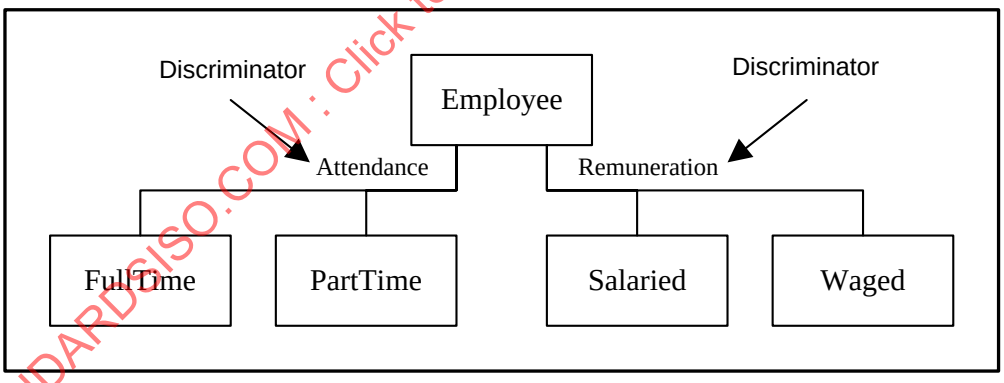

Figure 33 – Orthogonal Subtype Sets

Figure 34 is an instance diagram showing an instance of ***Employee*** and its subtype instances.

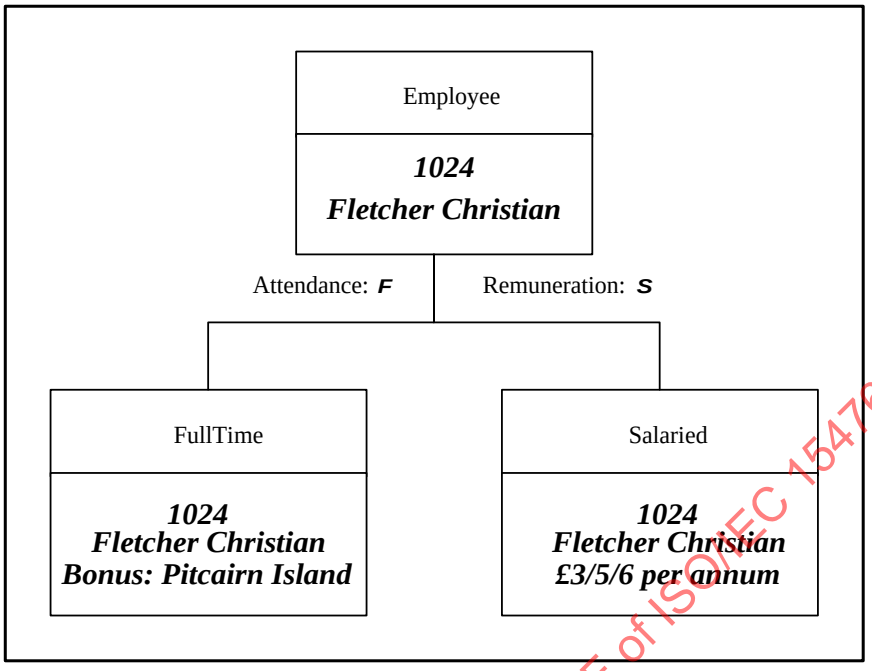

Figure 34 – Instance Diagram showing Orthogonal Subtype Sets

A subtype can be a member of more that one subtype set. In Figure 35, the subtype ***Amphibian*** is a member of the ***Vehicle*** subtype set and the ***Boat*** subtype set.

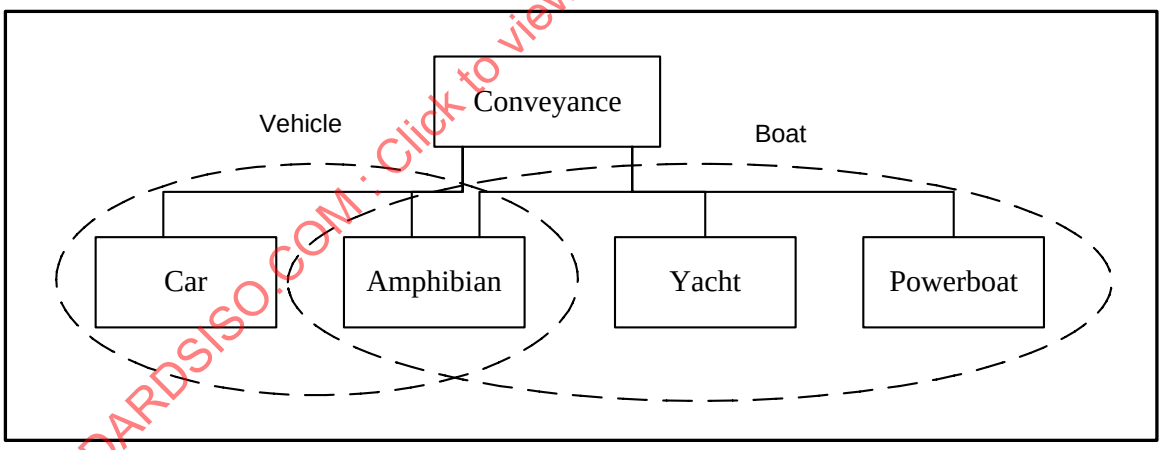

Figure 35 – Overlapping Subtype Sets

6.12.4 Inheritance

Inheritance imbues subtypes with the properties of their supertypes. Subtypes inherit all of the attributes and all of the relationships that describe their supertypes. Partial inheritance of attributes and relationships is not supported. Inheritance from a single supertype and inheritance from multiple supertypes is supported for both *Entities* and *Relationships*. For example, Figure 36 depicts an *Entity* super/subtype construct. ***Truck***, ***Car***, and ***Motorcycle***, ***Articulated*** and ***Non-Articulated*** inherit the ***SerialNumber*** and ***LicencePlateNumber*** attributes from ***Vehicle***. ***Articulated*** and ***Non-Articulated*** inherit the attribute ***Tare*** from ***Truck***. In addition, all inherit the relationship ***Vehicle.IsOwnedBy.Owner***.

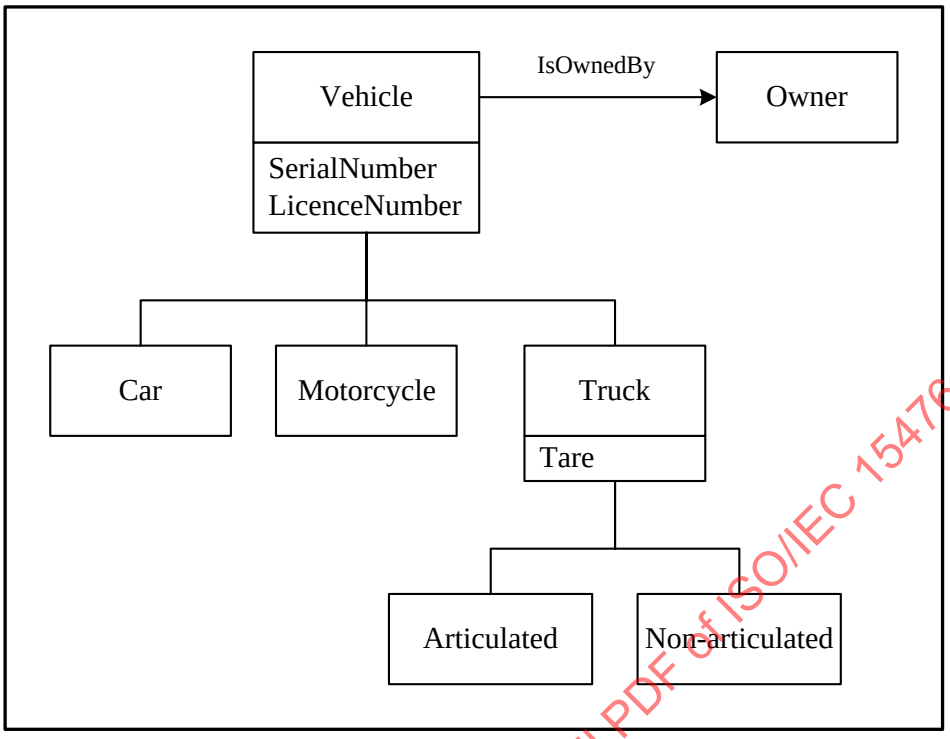

Figure 36 – Inheritance

The meta-relationship *Attribute.IsInheritedFrom.Attribute* is used to implement attribute inheritance. At each level of the subtyping hierarchy, there is a local copy of each inherited *Attribute* pointing to the corresponding *Attribute* in the immediate supertype. This local copy can be renamed. To determine any of the characteristics of an inherited *Attribute*, the attribute inheritance hierarchy must be navigated to find the root *Attribute* (by using the meta-relationship *Attribute.IsInheritedFrom.Attribute*). In the situation where an *Attribute* can be inherited from two supertypes, the definitions of the inherited *Attributes'* meta-attributes (except *Name*) and meta-relationships must be identical.

Inheritance of both *Attributes* and *Relationships* from a supertype cannot be selective. All the *Attributes* and *Relationships* are inherited. Once inherited, local override of an *Attribute* name can occur. See 6.13.

6.13 Refinement of inherited characteristics

6.13.1 Refinement of attribute characteristics

Both inherited *Attributes* and inherited *Relationships* can have their characteristics refined..

Inherited attributes can have their characteristics further refined to:

- Change the name of an inherited attribute. Each inherited attribute will have an instance of *Attribute* to represent it and can provide a new value for the meta-attribute *Name*. The meta-relationship *Attribute.IsInheritedFrom.Attribute* is used to relate the *Attribute* in the supertype to the renamed inherited *Attribute*.

- Alter the length of a data type. For example, ***Truck SerialNumbers*** may be shorter than ***Vehicle SerialNumbers***. The Data Definition subject area describes this in more detail.

- Further constrain the domain values. The Data Definition subject area describes this in more detail..

6.13.2 Refinement of relationship characteristics

An inherited relationship can have its cardinalities refined to reflect tighter constraints. Minimum cardinality values can be increased and maximum cardinality values can be decreased. For example, in Figure 37, the relationship ***Airplane.IsStaffedBy.FlightCrew***, the cardinality is 0:1 to 3:N. This indicates that in general, the minimum number of ***FlightCrew*** is three. ***JumboJet*** inherits this relationship and in so doing refines the cardinality so that the minimum number of ***FlightCrew*** is twenty.

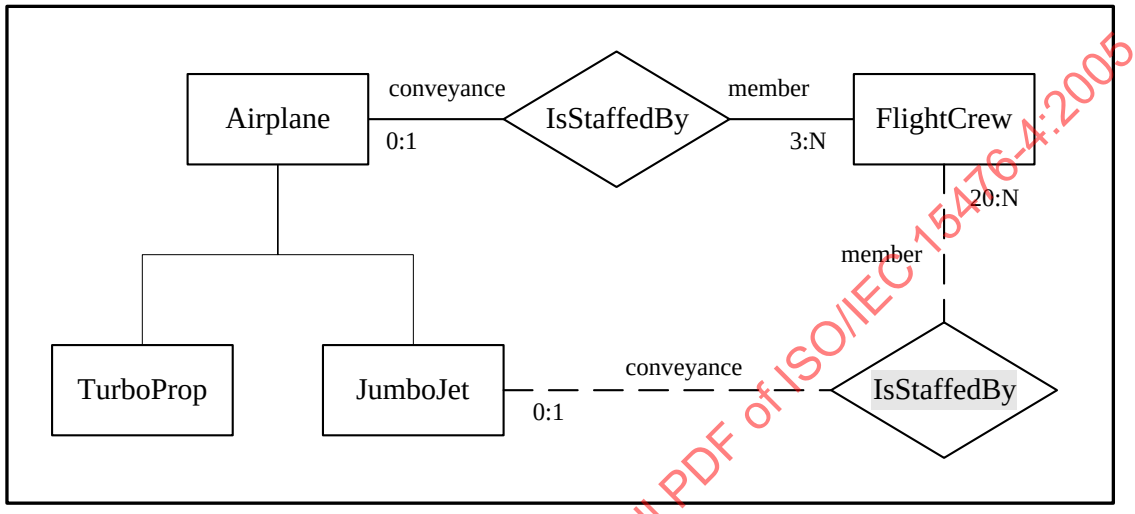

Figure 37 – Refinement of Cardinalities

This refinement mechanism is implemented using the meta-relationships *RolePlayer.Refines.RolePlayer* and *RolePlayer.RefinesForSubtype.DataModelObject*. *RolePlayer.Refines.RolePlayer* provides the mechanism to refine the cardinalities. *RolePlayer.RefinesForSubtype.DataModelObject* provides the identification of the particular subtypes that participate in the refined relationship. For example, ***JumboJet*** is identified as the subytpe of ***Airplane*** for which the role player ***FlightCrew*** is being refined.

Figure 38 is an instance diagram of part of Figure 37, showing the representation of the model.

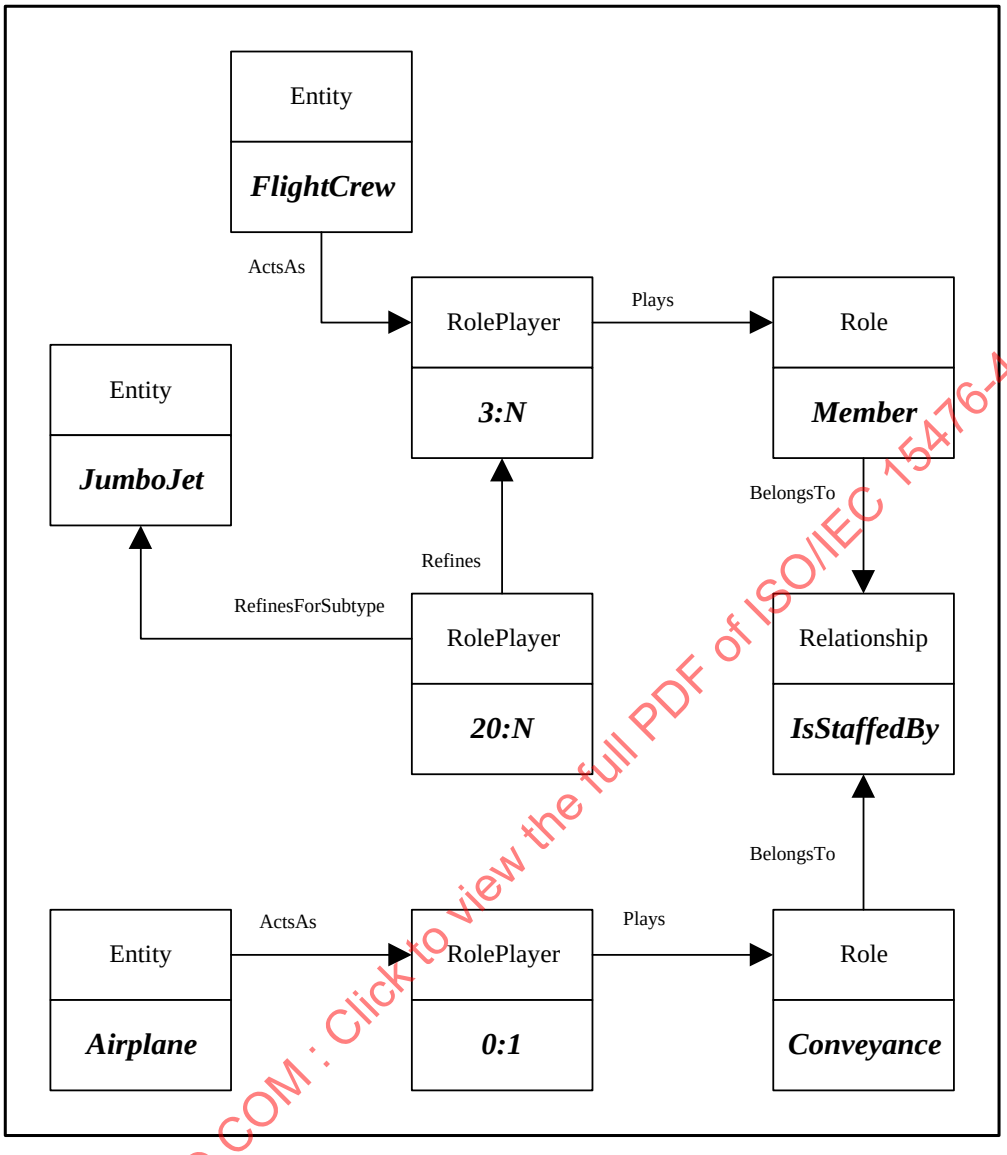

Figure 38 – Instance Diagram showing refinement of Cardinalities.

6.14 Subsets

Subsets are a means of defining a collection of objects taken from a larger model. They are a partition of the model.

The concept of defining subsets of a model (implemented by *DataModelSubset.IsSubsetOf.DataModel*), which may overlap each other, is supported by *DataModelSubset* which belongs to the parent *DataModel*. This Subject Area standard is an example of a subset of the single overall CDIF Semantic Meta-model.

Figure 39 depicts an overall data model that has been partitioned into two subsets - ***Subset 1*** and ***Subset 2***. Note that ***Subset 2*** overlaps ***Subset 1*** in that they both include *Entities* ***A*** and *B* and the relationship between ***A*** and ***B***.

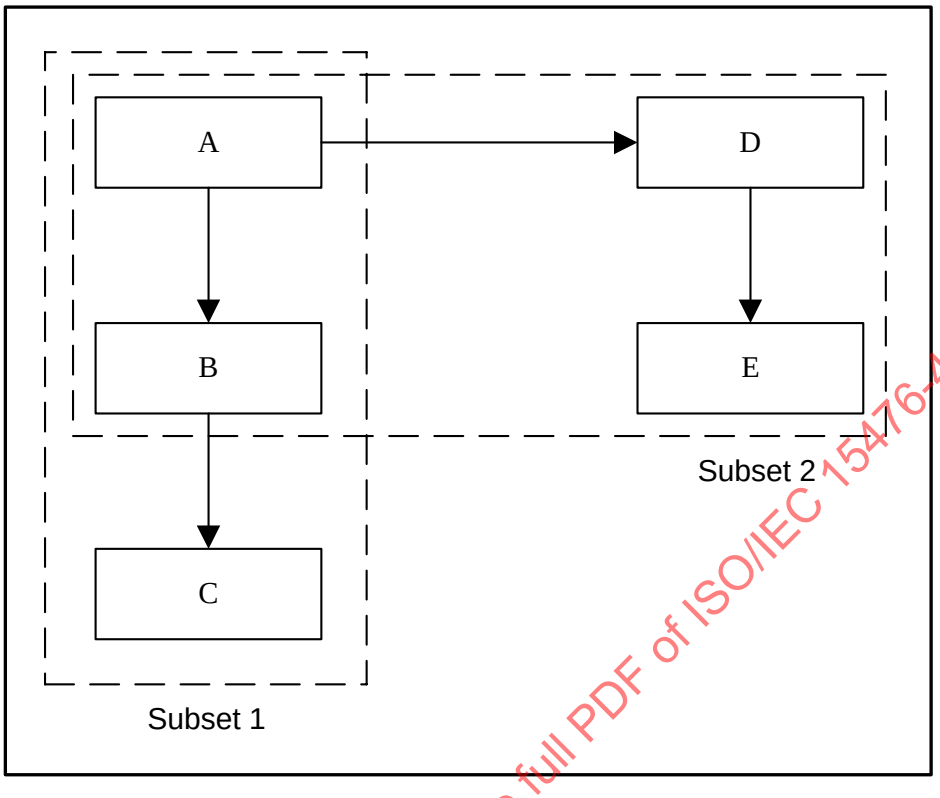

Figure 39 – Subsets

Further restriction of the subset to limit the *Attributes* that will be included is done by using the meta-relationship *DataModelSubset.Excludes.Attribute.* For example, in a human resources data model, a subset of the ***Employee*** entity could be created with the attribute ***Salary*** excluded, thereby limiting access to this information.

6.15 Clusters

Clusters provide an abstraction mechanism which can be used for providing high-level views of a data model by suppressing the lower level detail. They effectively provide a decomposition mechanism for data models, similar to process decomposition in data flow Modelling. This is sometimes referred to as subject database Modelling or as Aggregation.

Clusters represent collections of *Entities*, *Relationships* and other *Clusters*. This concept is usually used on high-level data models where the level of detail of a complete model is abstracted, showing the relationships between *Clusters*. Figure 40 depicts this process of abstraction. The *Clusters* named ***Stock Detail*** and ***Order Detail***, and the relationship ***Fulfills*** are the content of the model that is being viewed, rather than the full detail of all *Entities* and *Relationships* in the model that lie under the dotted line. *Clusters* are used to give a "summary" of the areas covered by a model and the relationships between the areas. It is distinct from inheritance in that *Clusters* are not attributed and they can contain both *Entities* and *Relationships*. *Clusters* can have relationships with *Clusters*, *Entities*, and *Relationships*.

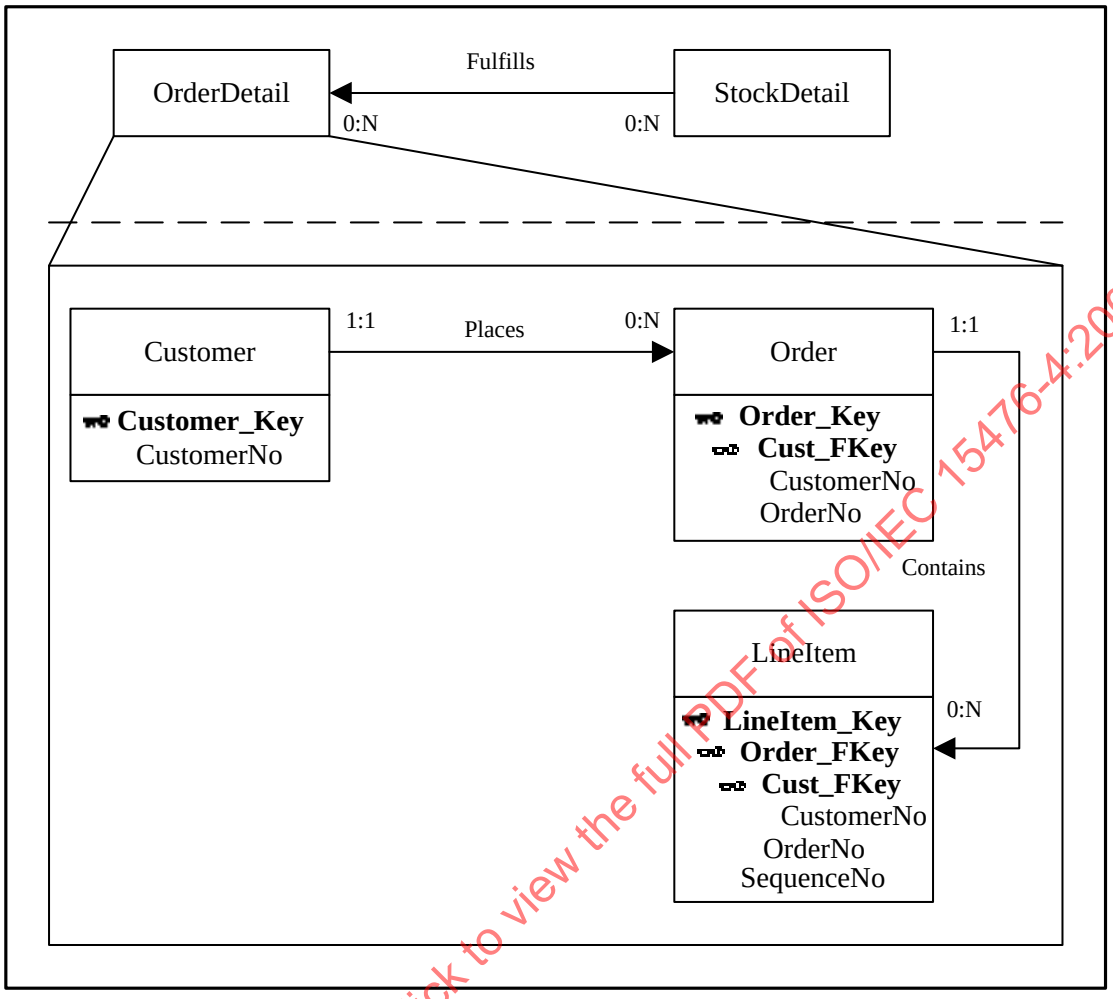

Figure 40 – *Clusters*

6.16 Projections

The capability to model "Virtual" entities (entities that are composed of one or more underlying "base" entities) is provided through the *ProjectionComponent* and *ProjectedAttribute* meta-entities. Entire sets or subsets of *Attributes* from the underlying *DefinitionObjects* are included in the resultant *DefinitionObject*. This provides the data modeler the capability to model users' views of data objects in the enterprise, while preserving a rigorous normalized data model. These views can be used at all stages of development, from conceptual data Modelling through to Database Design (e.g., SQL Views).

Simple projection is accomplished using the *ProjectedAttribute* meta-entity and the meta-relationships *DefinitionObject.Contains.ComponentObject* and *ProjectedAttribute.IsProjectionOf.Attribute*. Figure 41 is an instance diagram showing that an *Entity* instance is described by a mix of *Attributes* and a *ProjectedAttribute*.

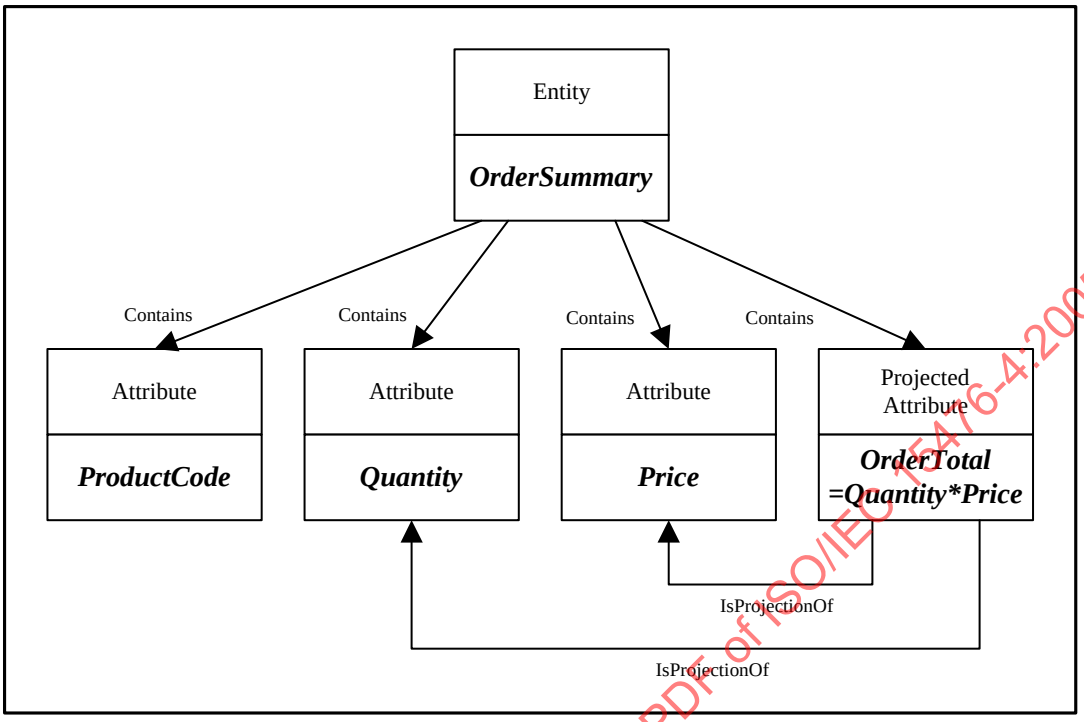

Figure 41 – Instance Diagram showing Simple Projection

More complex projections are constructed using the *ProjectionComponent* meta-entity and the meta-relationships *DefinitionObject.IsConstructedWith.ProjectionComponent* and *ProjectionComponent.IsProjectionOf.Attribute.* Figure 42 is an instance diagram showing a projection ***OrderSummary*** that subsets the *Attributes* from ***Customer*** and ***LineItem*** entities. The full descriptions of the entities are as follows:

OrderSummary
- Name
- Number
- OrderTotal (sum (Total)

Customer
- Name
- Address
- Number
- PhoneNumber
- FaxNumber

LineItem
- ProductCode
- Quantity
- Price
- Total (=Price*Quantity)

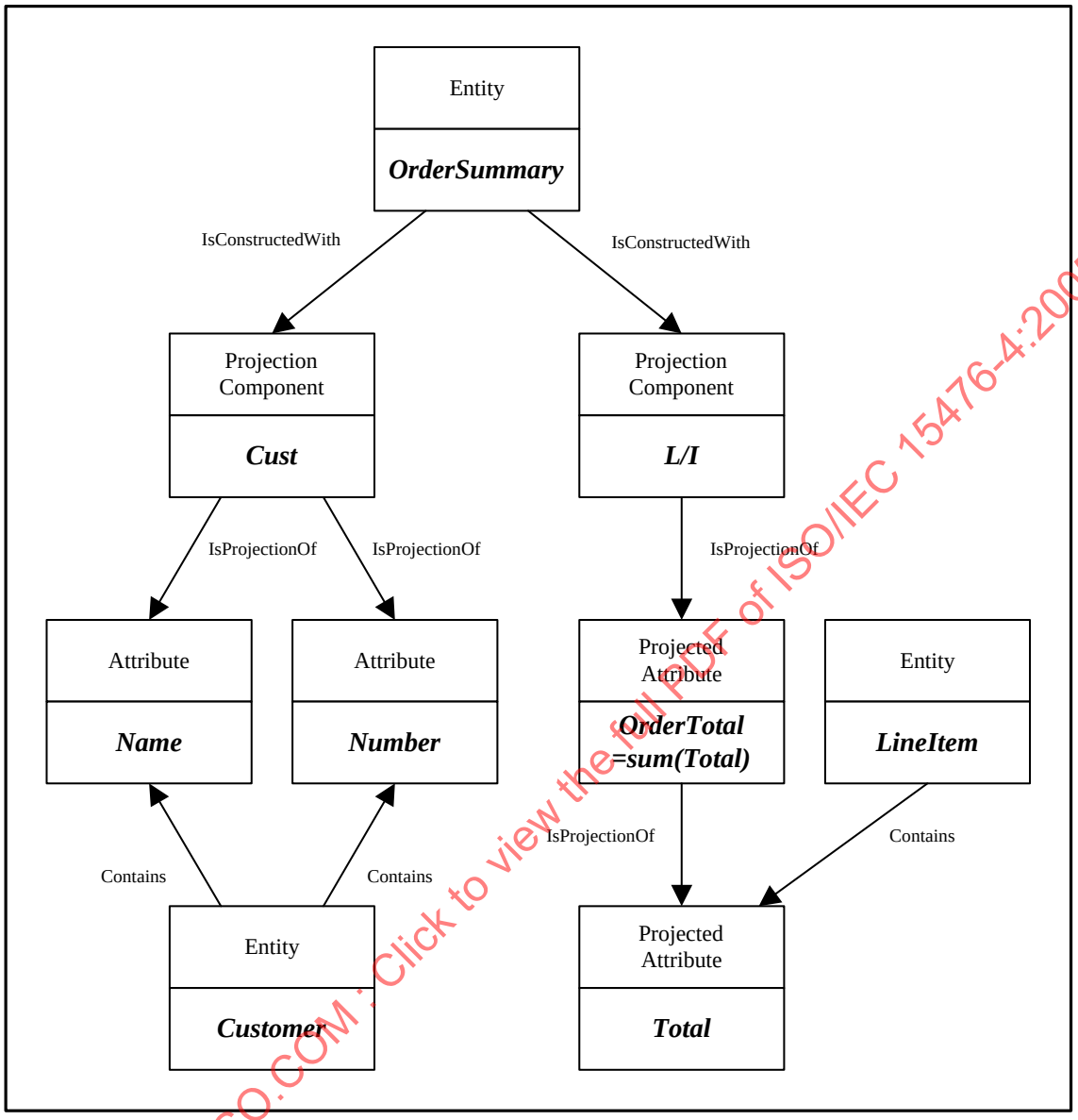

Figure 42 – Instance Diagram showing Complex Projection

A full projection, such as the "select *" in SQL is provided by the meta-relationship *ProjectionComponent.IsFullProjectionOf.DefinitionObject*. This provides the capability to fully project a *DefinitionObject* in terms of its currently defined *Attributes*, without the need to enumerate them.

Renaming of *Attributes* or *ProjectedAttributes* that are contained in a *ProjectionComponent* is accomplished by creating instances of *ProjectedAttribute* that contain new values for the *Name* meta-attribute. These are then related to the *ProjectionComponent* using the meta-relationship *ProjectionComponent.IsProjectionOf.Attribute* and to the renamed *Attributes* using the meta-relationship *ProjectedAttribute.IsProjectionOf.Attribute*. Figure 43 is an instance diagram showing renaming of *Attributes*.

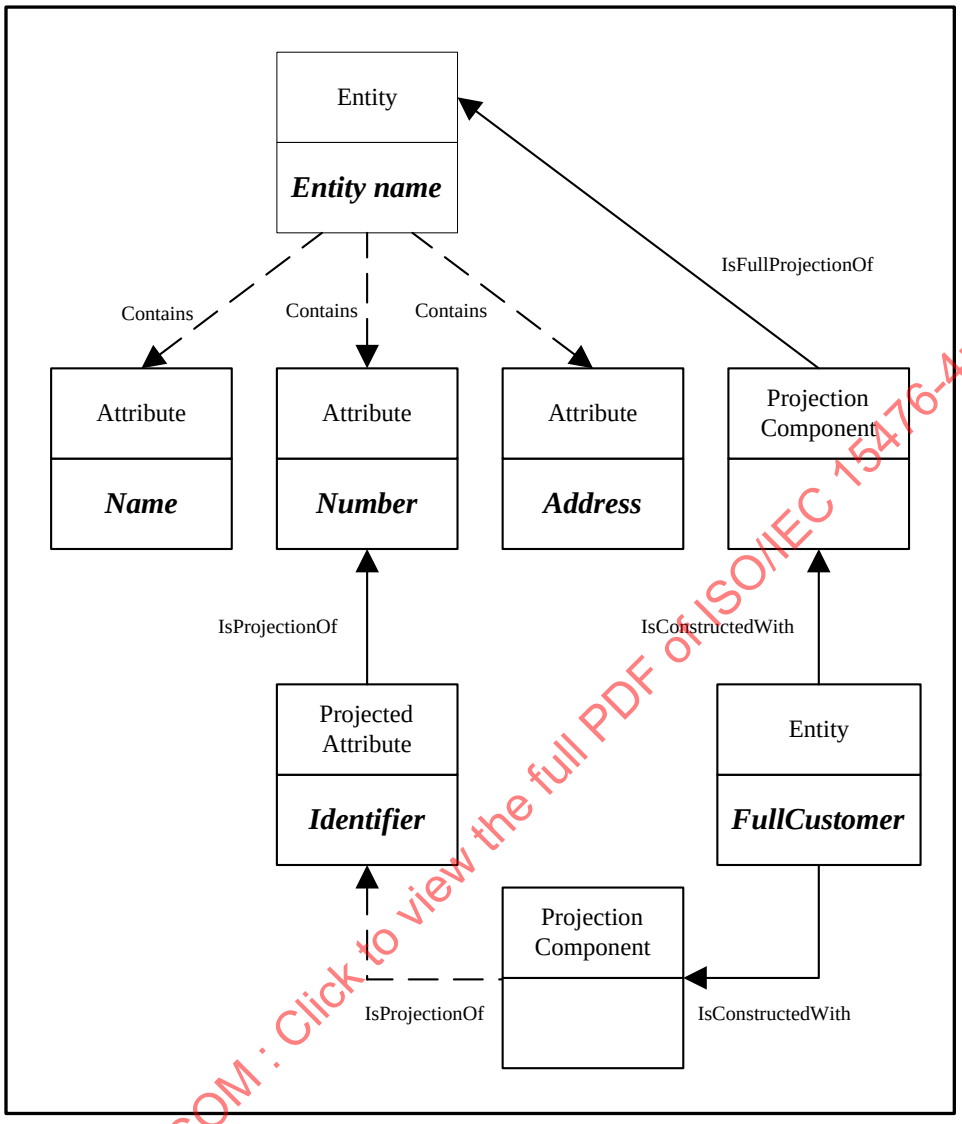

Figure 43 – Instance Diagram showing *Attribute* Renaming

6.17 Computable languages

The set of computable languages that can be supported is defined in *ISO/IEC 15476-2:2000, Information Technology - CDIF Semantic Meta-model - Part 2: Common*.

[Blank page]

7 Data models subject area summary

7.1 AttributableMetaObject hierarchy

MetaObject Name	SubjectArea Name
RootEntity	Foundation
SemanticInformationObject	Common
AccessPath	Data Models
ComponentObject	Common
Attribute	Data Models
ProjectedAttribute	Data Models
DataModel	Data Models
DataModelObject	Data Models
Cluster	Data Models
InheritableDataModelObject	Data Models
Entity	Data Models
Relationship	Data Models
DataModelSubset	Data Models
DefinitionObject	Common
Cluster	Data Models
Entity	Data Models
Relationship	Data Models
Role	Data Models
RolePlayer	Data Models
Key	Data Models
CandidateKey	Data Models
ForeignKey	Data Models
ProjectionComponent	Data Models
RoleConstraint	Data Models
SubtypeSet	Data Models
SubtypeSetMembershipCriterion	Data Models
RootEntity.IsRelatedTo.RootEntity	Foundation
AccessPath.Incorporates.Attribute	Data Models
AccessPath.Instantiates.Key	Data Models
Attribute.IsDiscriminatorFor.SubtypeSetMembershipCriterion	Data Models
Attribute.IsInheritedFrom.Attribute	Data Models
Cluster.Collects.DataModelObject	Data Models
ComponentObject.References.DefinitionObject	Common
DataModel.Collects.DataModelObject	Data Models
DataModelObject.ActsAs.RolePlayer	Data Models
DataModelObject.IsMemberOf.DataModelSubset	Data Models
DataModelSubset.Excludes.Attribute	Data Models
DataModelSubset.IsSubsetOf.DataModel	Data Models
DefinitionObject.Contains.ComponentObject	Common
DefinitionObject.IsConstructedWith.ProjectionComponent	Data Models
Entity.IsAccessedUsing.AccessPath	Data Models
Entity.IsIdentifiedBy.CandidateKey	Data Models

MetaObject Name	SubjectArea Name
Repeated for Clarity	
RootEntity.IsRelatedTo.RootEntity	Foundation
ForeignKey.References.CandidateKey	Data Models
InheritableDataModelObject.IsSubtypeIn.SubtypeSet	Data Models
InheritableDataModelObject.IsSupertypeFor.SubtypeSet	Data Models
Key.Incorporates.SemanticInformationObject	Data Models
CandidateKey.Incorporates.ForeignKey	Data Models
ForeignKey.Incorporates.RolePlayer	Data Models
Key.Incorporates.Attribute	Data Models
ProjectedAttribute.IsProjectionOf.Attribute	Data Models
ProjectionComponent.IsFullProjectionOf.DefinitionObject	Data Models
ProjectionComponent.IsProjectionOf.Attribute	Data Models
Role.BelongsTo.Relationship	Data Models
RoleConstraint.Incorporates.SemanticInformationObject	Data Models
RoleConstraint.Incorporates.RoleConstraint	Data Models
RoleConstraint.Incorporates.RolePlayer	Data Models
RolePlayer.IsSupportedBy.Key	Data Models
RolePlayer.Plays.Role	Data Models
RolePlayer.Refines.RolePlayer	Data Models
RolePlayer.RefinesForSubtype.DataModelObject	Data Models
SubtypeSet.Specifies.SubtypeSetMembershipCriterion	Data Models
SubtypeSetMembershipCriterion.Selects.InheritableDataModelObject	Data Models

7.2 MetaEntity summary

All MetaEntities	
All meta-entities contain the following inherited meta-attributes, which are omitted from the entries in this clause:	
CDIFIdentifier	*Mandatory*
DateCreated	*Optional*
DateUpdated	*Optional*
TimeCreated	*Optional*
TimeUpdated	*Optional*
AccessPath	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	Optional
SpecificationLanguage	Optional
SpecificationText	Optional
Attribute	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
DefaultValue	Optional
IsOptional	Optional
CandidateKey	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
IsPrimary	Optional

Cluster	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*

ComponentObject	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
DataModel	
BriefDescription	*Optional*
FullDescription	*Optional*
ModelType	Optional
Name	Optional
DataModelObject	
BriefDescription	*Optional*
FullDescription	*Optional*
DataModelSubset	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	Optional
DefinitionObject	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
Entity	
BriefDescription	*Optional*
FullDescription	*Optional*
IsAbstract	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
AvgNumberOfOccurrences	Optional
DeletionTimePeriod	Optional
EntityType	Optional
InsertionTimePeriod	Optional
MaxNumberOfOccurrences	Optional
MinNumberOfOccurrences	Optional
NormalizationState	Optional
NumberOfDeletions	Optional
NumberOfInsertions	Optional
NumberOfReads	Optional
NumberOfUpdates	Optional
ReadTimePeriod	Optional
UpdateTimePeriod	Optional

Usage	Optional
ForeignKey	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
InheritableDataModelObject	
BriefDescription	*Optional*
FullDescription	*Optional*
IsAbstract	Optional
Key	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	Optional
SpecificationLanguage	Optional
SpecificationText	Optional
ProjectedAttribute	
BriefDescription	*Optional*
FullDescription	*Optional*
DefaultValue	*Optional*
IsOptional	*Optional*
Name	*Optional*
SpecificationLanguage	Optional
SpecificationText	Optional
ProjectionComponent	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	Optional
SpecificationLanguage	Optional
SpecificationText	Optional
Relationship	
BriefDescription	*Optional*
FullDescription	*Optional*
IsAbstract	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*

InverseName	Optional
Role	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
IsMaster	Optional
IsSource	Optional
RoleConstraint	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	Optional
Operator	Optional
RolePlayer	
BriefDescription	*Optional*
FullDescription	*Optional*
Name	*Optional*
Operator	*Optional*
SpecificationLanguage	*Optional*
SpecificationText	*Optional*
AvgNumberOfOccurrences	Optional
DeleteEffect	Optional
DeletionTimePeriod	Optional
InsertEffect	Optional
InsertionTimePeriod	Optional
IsDeleteDeferrable	Optional
IsInsertDeferrable	Optional
IsUpdateDeferrable	Optional
MaxInnerCardinality	Optional
MaxNumberOfOccurrences	Optional
MaxOuterCardinality	Optional
MinInnerCardinality	Optional
MinNumberOfOccurrences	Optional
MinOuterCardinality	Optional
NumberOfDeletions	Optional
NumberOfInsertions	Optional
NumberOfReads	Optional
NumberOfUpdates	Optional
ReadTimePeriod	Optional
UpdateEffect	Optional
UpdateTimePeriod	Optional
SemanticInformationObject	
BriefDescription	*Optional*
FullDescription	*Optional*

SubtypeSet	
BriefDescription	*Optional*
FullDescription	*Optional*
IsExclusive	Optional
Name	Optional
SubtypeListIsClosed	Optional

SubtypeSetMembershipCriterion	
BriefDescription	*Optional*
FullDescription	*Optional*
DiscriminatorValue	Optional
SpecificationLanguage	Optional
SpecificationText	Optional

7.3 MetaRelationship summary

All MetaRelationships

All meta-relationships contain the following inherited meta-attributes, which are omitted from the entries in this clause:

CDIFIdentifier	*Mandatory*
DateCreated	*Optional*
DateUpdated	*Optional*
TimeCreated	*Optional*
TimeUpdated	*Optional*

AccessPath.Incorporates.Attribute	
IsAscending	Optional
SequenceNumber	Optional

AccessPath.Instantiates.Key

Attribute.IsDiscriminatorFor.SubtypeSetMembershipCriterion

Attribute.IsInheritedFrom.Attribute

CandidateKey.Incorporates.ForeignKey	
SequenceNumber	*Optional*

Cluster.Collects.DataModelObject
ComponentObject.References.DefinitionObject
DataModel.Collects.DataModelObject
DataModelObject.ActsAs.RolePlayer
DataModelObject.IsMemberOf.DataModelSubset
DataModelSubset.Excludes.Attribute
DataModelSubset.IsSubsetOf.DataModel
DefinitionObject.Contains.ComponentObject
DefinitionObject.IsConstructedWith.ProjectionComponent
SequenceNumber Optional
Entity.IsAccessedUsing.AccessPath
Entity.IsIdentifiedBy.CandidateKey
ForeignKey.Incorporates.RolePlayer
SequenceNumber *Optional*
ForeignKey.References.CandidateKey

InheritableDataModelObject.IsSubtypeIn.SubtypeSet	
SpecificationLanguage	Optional
SpecificationText	Optional
StoreWithSupertype	Optional
InheritableDataModelObject.IsSupertypeFor.SubtypeSet	
Key.Incorporates.Attribute	
SequenceNumber	*Optional*
IsAscending	Optional
Key.Incorporates.SemanticInformationObject	
SequenceNumber	Optional
ProjectedAttribute.IsProjectionOf.Attribute	
ProjectionComponent.IsFullProjectionOf.DefinitionObject	
ProjectionComponent.IsProjectionOf.Attribute	
SequenceNumber	Optional
Role.BelongsTo.Relationship	
RoleConstraint.Incorporates.RoleConstraint	
RoleConstraint.Incorporates.RolePlayer	
RoleConstraint.Incorporates.SemanticInformationObject	
RolePlayer.IsSupportedBy.Key	
RolePlayer.Plays.Role	
RolePlayer.Refines.RolePlayer	

RolePlayer.RefinesForSubtype.DataModelObject

SubtypeSet.Specifies.SubtypeSetMembershipCriterion

SubtypeSetMembershipCriterion.Selects.InheritableDataModelObject

8 Data models subject area specification

8.1 Introduction

This clause provides the full definition of each object used in the Data Models subject area of the CDIF semantic metamodel.

8.2 Subject area definition

SUBJECT AREA DEFINITION	
NAME	**DataModels**
VERSIONNUMBER	15476-4:2005
CDIFMETAIDENTIFIER	1000
DESCRIPTION	This Subject Area covers the semantics of Data Modelling and Logical Database Design.
USAGE	
ALIASES	
CONSTRAINTS	

8.3 Meta-entity definitions

8.3.1 AccessPath

META-ENTITY DEFINITION	
NAME	**AccessPath**
CDIFMETAIDENTIFIER	1001
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	An *AccessPath* provides an efficient means of accessing an entity instance.
USAGE	It is used to define all the means of direct access to an *Entity*.
ALIASES	Alternate Key, Index, Multi-valued Key, Duplicate Key
CONSTRAINTS	
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	AccessPath.Incorporates.Attribute AccessPath.Instantiates.Key Entity.IsAccessedUsing.AccessPath
LOCAL METAATTRIBUTES	Name SpecificationLanguage SpecificationText

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF AccessPath
NAME	**Name**
CDIFMETAIDENTIFIER	1081
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is the name of the *AccessPath.*
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF AccessPath
NAME	**SpecificationLanguage**
CDIFMETAIDENTIFIER	1082
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This states what language has been used for the *SpecificationText* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	A value shall be supplied when *SpecificationText* is present.
DATA TYPE	Enumerated
DOMAIN	See Clause 6 topic, Computable Languages.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF AccessPath
NAME	**SpecificationText**
CDIFMETAIDENTIFIER	1083
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is used to record text which specifies the *AccessPath.*
USAGE	
ALIASES	
CONSTRAINTS	If a *SpecificationText* is given, a *SpecificationLanguage* shall be given.
DATA TYPE	Text
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.2 Attribute

META-ENTITY DEFINITION	
NAME	**Attribute**
CDIFMETAIDENTIFIER	17
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	An *Attribute* represents a fact about an object. It is the lowest level component of data that describes an object. The value that it holds may or may not be typed. The type of the *Attribute* is defined through the inherited meta-relationship *ComponentObject.References.DefinitionObject*.
USAGE	For example, a ***Customer*** object has ***Name***, ***Address*** and ***PhoneNumber*** attributes.
ALIASES	Data Element, Instance Variable, Data Member, Column
CONSTRAINTS	
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	ProjectedAttribute
LOCAL METARELATIONSHIPS	Attribute.IsDiscriminatorFor.SubtypeSetMembershipCriterion Attribute.IsInheritedFrom.Attribute AccessPath.Incorporates.Attribute DataModelSubset.Excludes.Attribute Key.Incorporates.Attribute ProjectedAttribute.IsProjectionOf.Attribute ProjectionComponent.IsProjectionOf.Attribute
LOCAL METAATTRIBUTES	DefaultValue IsOptional

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Attribute
NAME	**DefaultValue**
CDIFMETAIDENTIFIER	18
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This meta-attribute specifies the default value to be used for the attribute when no user-supplied value is given. The value is given as a character string, although the actual type is dependent on the underlying *DataType* of the *Attribute*, if the type has been defined.
USAGE	When this meta-attribute has a value, and the *IsOptional* meta-attributes is ***FALSE*** this is equivalent to NOT NULL WITH DEFAULT in an SQL table column definition.
ALIASES	
CONSTRAINTS	The value supplied shall lie within the valid domain for the underlying type of the *Attribute*, if this has been defined. No value shall be supplied for *DefaultValue* if the value of *IsOptional* is ***TRUE***.
DATA TYPE	String
DOMAIN	
LENGTH	1024
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Attribute
NAME	**IsOptional**
CDIFMETAIDENTIFIER	19
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This describes whether or not a value for the *Attribute* must be given in every instance of the parent *DefinitionObject*. This meta-attribute is itself optional because the user may not have defined the optionality of the *Attribute* instance, and the value is therefore unknown.
USAGE	For example, within a relational database environment, columns of a table may be specified ***as NULL VALUES ALLOWED***; this indicates that the component of the table is optional.
ALIASES	
CONSTRAINTS	*IsOptional* shall not be specified with a value of ***TRUE*** when a value is supplied for *DefaultValue*.
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.3 CandidateKey

META-ENTITY DEFINITION	
NAME	**CandidateKey**
CDIFMETAIDENTIFIER	1003
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	A *CandidateKey* is a means of identification of an *Entity*. Candidate keys contain unique values and thus uniquely identify entity instances. A primary key is simply one of the set of candidate keys for a given *Entity* and signifies that this is the major identifier (meta-attribute *IsPrimary*).
USAGE	It identifies an *Entity*. It is also used as the basis of normalization and in conjunction with *ForeignKeys* provides referential integrity.
ALIASES	
CONSTRAINTS	
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	CandidateKey.Incorporates.ForeignKey Entity.IsIdentifiedBy.CandidateKey ForeignKey.References.CandidateKey
LOCAL METAATTRIBUTES	IsPrimary

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF CandidateKey
NAME	**IsPrimary**
CDIFMETAIDENTIFIER	1004
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	If ***TRUE***, this defines a particular *CandidateKey* as having been selected as the primary key for an *Entity*.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.4 Cluster

META-ENTITY DEFINITION	
NAME	**Cluster**
CDIFMETAIDENTIFIER	1005
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	A *Cluster* is a data aggregation that hides details of a *DataModel*.
USAGE	Clusters can be used to define decomposition in a *DataModel* by grouping together all the *Entities* and *Relationships* for a particular area. This enables "high-level" versions of a data model to be produced by abstracting out the detail and thus aiding the understanding of large, complex models.
ALIASES	Facet, Aggregate, SubModel
CONSTRAINTS	*Clusters* shall not contain *Attributes*.
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	Cluster.Collects.DataModelObject
LOCAL METAATTRIBUTES	

8.3.5 ComponentObject

META-ENTITY REFERENCE	
NAME	**ComponentObject**
CDIFMETAIDENTIFIER	8000
SUBJECTAREANAME	Common
SUBJECTAREAVERSION	15476-2:2000
LOCAL SUBTYPES	Attribute
LOCAL METARELATIONSHIPS	
LOCAL METAATTRIBUTES	

8.3.6 DataModel

META-ENTITY DEFINITION	
NAME	**DataModel**
CDIFMETAIDENTIFIER	1008
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is a named collection of data Modelling objects, modelled using an Entity-Relationship-Attribute Modelling technique. The data model is conceptual or logical, not physical.
USAGE	
ALIASES	InformationModel, ConceptualModel, LogicalModel, LogicalDataBaseDesign
CONSTRAINTS	
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	DataModel.Collects.DataModelObject DataModelSubset.IsSubsetOf.DataModel
LOCAL METAATTRIBUTES	ModelType Name

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF DataModel
NAME	**ModelType**
CDIFMETAIDENTIFIER	1010
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the type of the *DataModel*. Examples include ***Business Model***, ***Conceptual Model***, ***Third Normal Form Model***, and ***Composite Model***.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	64
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF DataModel
NAME	**Name**
CDIFMETAIDENTIFIER	1011
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This contains the name of the *DataModel*.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

8.3.7 DataModelObject

META-ENTITY DEFINITION	
NAME	**DataModelObject**
CDIFMETAIDENTIFIER	1012
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	The meta-entity called *DataModelObject* serves as a supertype of the *MetaEntities* called *InheritableDataModelObject* and *Cluster*.
USAGE	
ALIASES	
CONSTRAINTS	This meta-entity shall not be instantiated directly since all instances should be more precisely classified.
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	Cluster InheritableDataModelObject
LOCAL METARELATIONSHIPS	DataModelObject.ActsAs.RolePlayer DataModelObject.IsMemberOf.DataModelSubset Cluster.Collects.DataModelObject DataModel.Collects.DataModelObject RolePlayer.RefinesForSubtype.DataModelObject
LOCAL METAATTRIBUTES	

8.3.8 DataModelSubset

META-ENTITY DEFINITION	
NAME	**DataModelSubset**
CDIFMETAIDENTIFIER	1013
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	A *DataModelSubset* is a portion of the entire *DataModel*.
USAGE	Large data models are often partitioned as an aid to understanding and usability.
ALIASES	Subset, SubModel
CONSTRAINTS	
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	DataModelSubset.Excludes.Attribute DataModelSubset.IsSubsetOf.DataModel DataModelObject.IsMemberOf.DataModelSubset
LOCAL METAATTRIBUTES	Name

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF DataModelSubset
NAME	**Name**
CDIFMETAIDENTIFIER	1015
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This contains the name of the *DataModelSubset*.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

8.3.9 DefinitionObject

META-ENTITY REFERENCE	
NAME	**DefinitionObject**
CDIFMETAIDENTIFIER	8002
SUBJECTAREANAME	Common
SUBJECTAREAVERSION	15476-2:2000
LOCAL SUBTYPES	Cluster Entity Relationship Role RolePlayer
LOCAL METARELATIONSHIPS	DefinitionObject.IsConstructedWith.ProjectionComponent ProjectionComponent.IsFullProjectionOf.DefinitionObject
LOCAL METAATTRIBUTES	

8.3.10 Entity

META-ENTITY DEFINITION	
NAME	**Entity**
CDIFMETAIDENTIFIER	1016
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	A thing or concept about which information is recorded.
USAGE	
ALIASES	Object, Concept, Table, Relation
CONSTRAINTS	
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	Entity.IsIdentifiedBy.CandidateKey Entity.IsAccessedUsing.AccessPath
LOCAL METAATTRIBUTES	AvgNumberOfOccurrences DeletionTimePeriod EntityType InsertionTimePeriod MaxNumberOfOccurrences MinNumberOfOccurrences NormalizationState NumberOfDeletions NumberOfInsertions NumberOfReads NumberOfUpdates ReadTimePeriod UpdateTimePeriod Usage

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**AvgNumberOfOccurrences**
CDIFMETAIDENTIFIER	1017
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the average number of instances of the *Entity* that are expected to exist at any particular time.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**DeletionTimePeriod**
CDIFMETAIDENTIFIER	1018
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfDeletions* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***. It shall be valued if *NumberOfDeletions* is valued.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**EntityType**
CDIFMETAIDENTIFIER	1018
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates the type of the *Entity*.
USAGE	***Kernel***: This classification is used for an entity whose instances are meaningful without the occurrences of other entities. For example, an instance of an entity, having a name and a description, is significant without the knowledge of the data model to which it belongs. The fact that an entity can be kernel does not imply that it can be uniquely identified without other scoping information. ***Characteristic***: Each instance of a characteristic entity is logically related to a single instance of another entity. Therefore, the attributes could be considered to belong to the related entity, where the “strong” entity is related to the characteristic entity with a relationship with cardinality of 1:1. Characteristic entities are usually used to group together related attributes which are optional, and enable the kernel entity to have a smaller fixed set of attributes. ***Associative***: This classification is used for entities whose existence is dependent upon the existence of two or more other entities; they are used to resolve many-to-many relationships et al. *Entity* instances cannot exist without the existence of instances of the related entity types.
ALIASES	
CONSTRAINTS	
DATA TYPE	Enumerated
DOMAIN	Kernel, Characteristic, Associative
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**InsertionTimePeriod**
CDIFMETAIDENTIFIER	1020
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfInsertions* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***. It shall be valued if *NumberOfInsertions* is valued.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**MaxNumberOfOccurrences**
CDIFMETAIDENTIFIER	1022
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the maximum number of instances of the *Entity* that are expected to exist at any particular time.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***.
DATA TYPE	Integer
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**MinNumberOfOccurrences**
CDIFMETAIDENTIFIER	1023
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the minimum number of instances of the *Entity* that are expected to exist at any particular time.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***.
DATA TYPE	Integer
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**NormalizationState**
CDIFMETAIDENTIFIER	1024
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates the Normalization state of the *Entity*. This may be Un-normalized Form, First Normal Form, Second Normal Form, Boyce-Codd Normal Form, Third Normal Form, Fourth Normal Form or Fifth Normal Form.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Enumerated
DOMAIN	UNF, 1NF, 2NF, 3NF, BCNF, 4NF, 5NF.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**NumberOfDeletions**
CDIFMETAIDENTIFIER	1025
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of deletion instances of the *Entity* in the time period specified in the meta-attribute *DeletionTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE*** or if no value is supplied for *DeletionTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**NumberOfInsertions**
CDIFMETAIDENTIFIER	1026
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of insertion instances of the *Entity* in the time period specified in the meta-attribute *InsertionTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE*** or if no value is supplied for *InsertionTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**NumberOfReads**
CDIFMETAIDENTIFIER	1027
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of instances of the *Entity* read in the time period specified in the meta-attribute *ReadTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE*** or if no value is supplied for *ReadTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**NumberOfUpdates**
CDIFMETAIDENTIFIER	1028
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of instances of the *Entity* updated in the time period specified in the meta-attribute *UpdateTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE*** or if no value is supplied for *UpdateTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**ReadTimePeriod**
CDIFMETAIDENTIFIER	1029
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfReads* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***. It shall be valued if *NumberOfReads* is valued.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**UpdateTimePeriod**
CDIFMETAIDENTIFIER	1030
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfUpdates* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued if *IsAbstract* is ***TRUE***. It shall be valued if *NumberOfUpdates* is valued.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Entity
NAME	**Usage**
CDIFMETAIDENTIFIER	1031
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This describes the way in which the *Entity* is used.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Text
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.11 ForeignKey

META-ENTITY DEFINITION	
NAME	**ForeignKey**
CDIFMETAIDENTIFIER	1032
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	A *ForeignKey* is the embodiment of a *CandidateKey* in a referencing *Entity*; i.e. a *Key* that is a *CandidateKey* in one *Entity* that is instantiated in another related *Entity* is called a *ForeignKey*.
USAGE	This provides the mechanism for implementing referential integrity between two *Entities* (or the same *Entity* using a reflexive relationship).
ALIASES	
CONSTRAINTS	
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	ForeignKey.References.CandidateKey ForeignKey.Incorporates.RolePlayer CandidateKey.Incorporates.ForeignKey
LOCAL METAATTRIBUTES	

8.3.12 InheritableDataModelObject

META-ENTITY DEFINITION	
NAME	**InheritableDataModelObject**
CDIFMETAIDENTIFIER	1033
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	The *meta-entity* called *InheritableDataModelObject* serves as a supertype of the *meta-entities* called *Entity* and *Relationship*.
USAGE	
ALIASES	
CONSTRAINTS	This meta-entity shall never be instantiated in a CDIF Transfer; the objects shall be classified as one of the subtypes.
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	Entity Relationship
LOCAL METARELATIONSHIPS	InheritableDataModelObject.IsSubtypeIn.SubtypeSet InheritableDataModelObject.IsSupertypeFor.SubtypeSet SubtypeSetMembershipCriterion.Selects.InheritableDataModelObject
LOCAL METAATTRIBUTES	IsAbstract

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF InheritableDataModelObject
NAME	**IsAbstract**
CDIFMETAIDENTIFIER	1034
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates that a supertype *Entity* or *Relationship* is abstract, and can therefore never be instantiated.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.13 Key

META-ENTITY DEFINITION	
NAME	**Key**
CDIFMETAIDENTIFIER	1035
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is an abstract supertype for *CandidateKey* and *ForeignKey*.
USAGE	
ALIASES	Identifier
CONSTRAINTS	This object shall never be instantiated in a CDIF Transfer; the objects shall be classified as one of the subtypes.
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	CandidateKey ForeignKey
LOCAL METARELATIONSHIPS	Key.Incorporates.Attribute Key.Incorporates.SemanticInformationObject AccessPath.Instantiates.Key RolePlayer.IsSupportedBy.Key
LOCAL METAATTRIBUTES	Name SpecificationLanguage SpecificationText

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Key
NAME	**Name**
CDIFMETAIDENTIFIER	1077
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is the name of the *Key*.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Key
NAME	**SpecificationLanguage**
CDIFMETAIDENTIFIER	1078
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This states what language has been used for the *SpecificationText* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	A value shall be supplied when *SpecificationText* is present.
DATA TYPE	Enumerated
DOMAIN	See Clause 6 topic, Computable Languages.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Key
NAME	**SpecificationText**
CDIFMETAIDENTIFIER	1140
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is used to record text which specifies the *Key.*
USAGE	
ALIASES	
CONSTRAINTS	If a *SpecificationText* is given, a *SpecificationLanguage* shall be given.
DATA TYPE	Text
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.14 ProjectedAttribute

META-ENTITY DEFINITION	
NAME	**ProjectedAttribute**
CDIFMETAIDENTIFIER	37
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	An attribute that is selected or whose value is derived, rather than stored. It can be selected or derived from other *Attributes* or *ProjectedAttributes*. The values can be derived by other means, such as through a system function.
USAGE	
ALIASES	View Element, View Column, Derived Attribute
CONSTRAINTS	
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	ProjectedAttribute.IsProjectionOf.Attribute
LOCAL METAATTRIBUTES	SpecificationLanguage SpecificationText

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF ProjectedAttribute
NAME	**SpecificationLanguage**
CDIFMETAIDENTIFIER	39
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This identifies what language has been used for the *SpecificationText* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	A value shall be supplied when *SpecificationText* is present.
DATA TYPE	Enumerated
DOMAIN	See Clause 6 topic, Computable Languages.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF ProjectedAttribute
NAME	**SpecificationText**
CDIFMETAIDENTIFIER	38
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This provides a definition of how the value of a *ProjectedAttribute* is derived. This can provide a simple definition, such as a literal value, or more complex definitions, such as selection expressions, limited only by the power of the language used, which is defined in the meta-attribute *SpecificationLanguage*.
USAGE	For example, ***Total = Price * Quantity*** defines a ProjectedAttribute.
ALIASES	
CONSTRAINTS	When a value is supplied, the language used shall be passed using the meta-attribute *SpecificationLanguage*.
DATA TYPE	Text
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.15 ProjectionComponent

META-ENTITY DEFINITION	
NAME	**ProjectionComponent**
CDIFMETAIDENTIFIER	1036
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This describes a component of a projection of the owning *DefinitionObject*. It may consist of a projection of all of a single *DefinitionObject* or it may be a set of attributes.
USAGE	
ALIASES	
CONSTRAINTS	
TYPE	Characteristic
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	ProjectionComponent.IsFullProjectionOf.DefinitionObject ProjectionComponent.IsProjectionOf.Attribute DefinitionObject.IsConstructedWith.ProjectionComponent
LOCAL METAATTRIBUTES	Name SpecificationLanguage SpecificationText

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF ProjectionComponent
NAME	**Name**
CDIFMETAIDENTIFIER	1037
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is the name of the *ProjectionComponent.*
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF ProjectionComponent
NAME	**SpecificationLanguage**
CDIFMETAIDENTIFIER	1038
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This states what language has been used for the *SpecificationText* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	A value shall be supplied when *SpecificationText* is present.
DATA TYPE	Enumerated
DOMAIN	See Computable Languages topic in Clause 6.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF ProjectionComponent
NAME	**SpecificationText**
CDIFMETAIDENTIFIER	1039
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	Where the *DefinitionObject* is a projection or view of another *ProjectionComponent* or set of *ProjectionComponents*, this defines how the projection is formed.
USAGE	The specification of how the projection is formed, is held in this meta-attribute.
ALIASES	
CONSTRAINTS	When a value is supplied, the language used shall be passed using the meta-attribute *SpecificationLanguage*.
DATA TYPE	Text
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.16 Relationship

META-ENTITY DEFINITION	
NAME	**Relationship**
CDIFMETAIDENTIFIER	1040
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	The *Relationship* meta-entity describes the concept of a relationship in Entity-Relationship-Attribute Modelling. A *Relationship* could be defined as the instance of two or more *DataModelObject* instances participating in some kind of interaction. *Relationships* may also participate in other *Relationships*.
USAGE	No volumetric information is provided for *Relationship*. Any such information shall be passed on the *RolePlayers* of the *Relationship*. When the *Relationship* is binary, the details shall be passed on the *RolePlayer* connected to the *Entity* to which the figures relate, giving the instances et al. for one instance of the other *Entity*. On a one-to-many relationship, the role used shall be that connected to the detail entity. The information shall not be duplicated, unless there are two sets of volumetrics for a many-to-many relationship, when they shall be passed on the appropriate *RolePlayer*.
ALIASES	
CONSTRAINTS	
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	Role.BelongsTo.Relationship
LOCAL METAATTRIBUTES	InverseName

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Relationship
NAME	**InverseName**
CDIFMETAIDENTIFIER	1041
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This contains the inverse or name of the *Relationship*, where bi-directional naming is supported.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

8.3.17 Role

META-ENTITY DEFINITION	
NAME	**Role**
CDIFMETAIDENTIFIER	1042
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	*Role* is a collection of one or more *RolePlayers*. A *Role* describes the participation of a *DataModelObject* in a *Relationship*. Each *Role* instance has direction, and may be attributed.
USAGE	
ALIASES	Partnership
CONSTRAINTS	
TYPE	Associative
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	Role.BelongsTo.Relationship RolePlayer.Plays.Role
LOCAL METAATTRIBUTES	IsMaster IsSource

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Role
NAME	**IsMaster**
CDIFMETAIDENTIFIER	1043
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates that the *Role* can be considered to be a master role in the *Relationship*. In an n-ary relationship, more than one role can be marked as master. The master roles identify the objects taking part in the *Relationship* which define the "context" or "scoping" for the other objects. This is not the same as *IsSource*, which merely indicates the direction of the relationship naming, although in general they will coincide on the same roles on binary relationships.
USAGE	This is used to indicate the master or parent objects in a relationship. For example, in a relationship ***Customer.Places.Orders***, the ***Customer*** would be the master role, since the ***Order*** is dependent on the ***Customer***.
ALIASES	IsParent
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF Role
NAME	**IsSource**
CDIFMETAIDENTIFIER	1044
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates the directionality of the *Relationship* by identifying the source participant(s).
USAGE	This is used to indicate the direction of the naming of the relationship or role. It is mainly required in binary relationships, where the name applies to the relationship as a whole, rather than to the role.
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

8.3.18 RoleConstraint

META-ENTITY DEFINITION	
NAME	**RoleConstraint**
CDIFMETAIDENTIFIER	1045
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is an optionally named or numbered set of groups of *RolePlayers* or *RoleConstraints* that are related to each other in the manner defined in the *Operator* meta-attribute. This enables complex interactions between roles to be modelled, including mutual exclusivity and mutual inclusivity.
USAGE	The *RoleConstraint* is used to "comprise" all the *RolePlayers* or *RoleConstraints* related by the single operator value. For example, *RolePlayers* played by mutually exclusive *DataModelObject* instances would be included in the same *RoleConstraint* with an operator value of ***XOR***. A more specific example would constrain a ***CustomerAccount*** such that it could not act as both the *RolePlayer* in the ***from*** and ***to*** *Roles* in a movement of funds for a bank transfer. Complex conditions can be built up by using *RoleConstraints* with different operators as components of other *RoleConstraints*.
ALIASES	PartnershipSet
CONSTRAINTS	*RoleConstraint* can participate in both *RoleConstraint.Incorporates.RoleConstraint* and *RoleConstraint.Incorporates.RolePlayer*.
TYPE	Kernel
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	RoleConstraint.Incorporates.RoleConstraint RoleConstraint.Incorporates.RolePlayer RoleConstraint.Incorporates.SemanticInformationObject
LOCAL METAATTRIBUTES	Name Operator

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RoleConstraint
NAME	**Name**
CDIFMETAIDENTIFIER	1046
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This is a name for the *RoleConstraint*.
USAGE	In some methods, a number is used for the group; this is considered to be the name of the group.
ALIASES	
CONSTRAINTS	
DATA TYPE	String
DOMAIN	
LENGTH	256
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RoleConstraint
NAME	**Operator**
CDIFMETAIDENTIFIER	1047
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This indicates how the contained *RolePlayers* or *RoleConstraints* are related.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Enumerated
DOMAIN	AND, OR, XOR
LENGTH	
ISOPTIONAL	True

8.3.19 RolePlayer

META-ENTITY DEFINITION	
NAME	**RolePlayer**
CDIFMETAIDENTIFIER	1048
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This models the concept that a *Role* can be played by many players (*DataModelObjects*). *RolePlayer* can also be attributed.
USAGE	This concept allows the Modelling of alternate role players as well as heterogeneous collections (or parts lists).
ALIASES	
CONSTRAINTS	If the value of *UpdateEffect* or *DeleteEffect* is "SET NULL" then the *Role* instance for the referenced *CandidateKey* shall have *MinOuterCardinality* of zero. That is, the cardinality of the relationship must agree with the referential integrity rules. In this case if a *CandidateKey* entity instance is being deleted or its key changes and the *ForeignKey* entity instance is to have its *ForeignKey* values set to ***NULL***, then the relationship between the two entities shall be optional.
TYPE	Associative
ISABSTRACT	False
LOCAL SUBTYPES	
LOCAL METARELATIONSHIPS	RolePlayer.IsSupportedBy.Key RolePlayer.Plays.Role RolePlayer.Refines.RolePlayer RolePlayer.RefinesForSubtype.DataModelObject DataModelObject.ActsAs.RolePlayer ForeignKey.Incorporates.RolePlayer RoleConstraint.Incorporates.RolePlayer
LOCAL METAATTRIBUTES	AvgNumberOfOccurrences DeleteEffect DeletionTimePeriod InsertEffect InsertionTimePeriod IsDeleteDeferrable IsInsertDeferrable IsUpdateDeferrable MaxInnerCardinality MaxNumberOfOccurrences MaxOuterCardinality MinInnerCardinality MinNumberOfOccurrences MinOuterCardinality NumberOfDeletions NumberOfInsertions NumberOfReads NumberOfUpdates ReadTimePeriod UpdateEffect UpdateTimePeriod

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**AvgNumberOfOccurrences**
CDIFMETAIDENTIFIER	1049
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the average number of instances of the *Relationship* to which a single instance of the *DataModelObject* acting as the *RolePlayer* is related.
USAGE	This is used to capture volumetric data. It defines the expected average number of instances. For example, on a relationship ***Places*** between ***Customer*** and ***Order***, it may be that there is an average of 30 ***Orders*** on the system for a ***Customer***. On a ternary relationship between ***Customer***, Stock and ***Depot***, the value on the participation of ***Stock*** could be 5, indicating that from each ***Depot*** related to a ***Customer***, an average of five ***Stock*** items are ordered.
ALIASES	
CONSTRAINTS	
DATA TYPE	Float
DOMAIN	This shall be zero or positive if specified.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**DeleteEffect**
CDIFMETAIDENTIFIER	1050
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the effect of a Delete operation on both the *CandidateKey* entity instance and the *ForeignKey* entity instance. If the entity instance acting as the *RolePlayer* is a *CandidateKey* instance then: • RESTRICTS The operation cannot complete if there are matching *ForeignKey* entity instances. This occurs where the *CandidateKey* entity's participation in the relationship is mandatory. For example, ***Customer.Places.Order*** (1:1 to 0:N). This is the equivalent of "NO ACTION" in SQL2. • CASCADES The operation will delete *ForeignKey* entity instances, and associated *ForeignKey* entity instances in a chain reaction, where the candidate key's *DeleteEffect* value is ***CASCADES***, until there are no more *ForeignKey* entity instances or the *ForeignKey* entity has a *DeleteEffect* value of ***RESTRICTS***. For example, an entity ***Order*** contains a foreign key, ***Cust_FKey***, that references the ***Customer*** entity, and the ***LineItem*** entity contains a foreign key, ***Order_FKey***, that references the ***Order*** entity. Deleting a ***Customer*** instance will cause a cascading delete of all referencing ***Order*** and ***LineItem*** instances. • SETNULL The delete operation will cause the values of referencing *ForeignKeys* to be set to ***NULL***. This preserves the *ForeignKey* entity instances. For example, in a model where we have the relationship ***Supplier.Fulfills.Order*** and the ***Supplier*** instance is deleted, the ***Order*** instances have their *ForeignKeys* set to ***NULL***.

DESCRIPTION (CONT)	• SETDEFAULT The delete operation will cause the values of referencing *ForeignKeys* to be set to default values as specified by the meta-attribute *DefaultValue*. This preserves the *ForeignKey* entity instances. For example, in a model where we have the relationship ***Supplier.Fulfills.Order*** and the ***Supplier*** instance is updated or deleted, the ***Order*** instances have their *ForeignKeys* set to a default value of ***UNKNOWN***. If the entity instance acting as the *RolePlayer* is a *ForeignKey* instance then: • RESTRICTS The delete operation cannot complete if the number of foreign key entity instances falls below the minimum cardinality value. For example, in the relationship ***Employee.WorksFor.Manager*** (1:N to 1:1) where a given ***Manager*** has only one ***Employee*** assigned. • CASCADES The operation will delete referenced *CandidateKey* entity instances when the last *ForeignKey* entity instance has been deleted. For example, for the relationship ***LineItem.IsContainedIn.Order***, when the last ***LineItem*** instance is deleted for a particular ***Order***, instance, the ***Order*** instance is deleted.
USAGE	This ensures referential integrity when a referenced *Entity* instance is deleted.
ALIASES	
CONSTRAINTS.....	
DATA TYPE.....	Enumerated
DOMAIN.....	RESTRICTS, CASCADES, SETNULL, SETDEFAULT
LENGTH.....	
ISOPTIONAL.....	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**DeletionTimePeriod**
CDIFMETAIDENTIFIER	1051
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfDeletions* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall be valued if a value is provided for *NumberOfDeletions*.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**InsertEffect**
CDIFMETAIDENTIFIER	1052
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the effect of an Insert operation on both the *CandidateKey* entity instance and the *ForeignKey* entity instance. If the entity instance acting as the *RolePlayer* is a *CandidateKey* instance then: • RESTRICTS The operation cannot complete if there are no matching *ForeignKey* entity instances. This occurs where the *ForeignKey* entity's participation in the relationship is mandatory. For example, ***Employee.Has.WorkSpace*** (0:1 to 1:1). • CASCADES An instance of the *ForeignKey* entity shall be inserted. This is the equivalent of deferred checking of integrity. This applies to a mandatory relationship. For example, ***ProductManager.IsResponsibleFor.Product*** (0:N to 1:N). If the entity instance acting as the *RolePlayer* is a *ForeignKey* instance then: • RESTRICTS The operation cannot complete if there are no matching *CandidateKey* entity instances. This occurs where the *CandidateKey* entity's participation in the relationship is mandatory. For example, ***Order.Has.LineItem*** (1:1 to 0:N). • CASCADES An instance of the *CandidateKey* entity shall be inserted. This is the equivalent of deferred checking of integrity. This applies to a mandatory relationship. For example, ***Customer.Places.Order*** (1:1 to 0:N).
USAGE	This ensures referential integrity when a referenced *Entity* instance is inserted.
ALIASES	
CONSTRAINTS	
DATA TYPE	Enumerated
DOMAIN	RESTRICTS, CASCADES, SETNULL, SETDEFAULT
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**InsertionTimePeriod**
CDIFMETAIDENTIFIER	1053
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberOfInsertions* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall be valued if a value is provided for *NumberOfInsertions*.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**IsDeleteDeferrable**
CDIFMETAIDENTIFIER	1054
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	If the value of this meta-attribute is ***TRUE***, then referential integrity constraints for an insert operation are deferrable until the end of the transaction.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**IsInsertDeferrable**
CDIFMETAIDENTIFIER	1055
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	If the value of this meta-attribute is ***TRUE***, then referential integrity constraints for an insert operation are deferrable until the end of the transaction.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**IsUpdateDeferrable**
CDIFMETAIDENTIFIER	1056
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	If the value of this meta-attribute is ***TRUE***, then referential integrity constraints for an update operation are deferrable until the end of the transaction.
USAGE	
ALIASES	
CONSTRAINTS	
DATA TYPE	Boolean
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MaxInnerCardinality**
CDIFMETAIDENTIFIER	1057
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the maximum number of instances of the *Relationship* acting as this *RolePlayer* that can be connected to a single instance of the *DataModelObject*. This is the maximum cardinality of the owning *Relationship* seen from the viewpoint of the participating *DataModelObject*.
USAGE	
ALIASES	
CONSTRAINTS	The value, if defined, shall be equal to or greater than the value of *MinInnerCardinality*. It shall not conflict with the value of *MaxNumberOfOccurrences*.
DATA TYPE	String
DOMAIN	Numeric values are held as the appropriate digit string and "N" represents an undefined upper-bound.
LENGTH	10
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MaxNumberOfOccurrences**
CDIFMETAIDENTIFIER	1058
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the maximum expected number of instances of the *Relationship* to which a single instance of the *DataModelObject* playing this *Role* is related.
USAGE	This is used to capture the volumetric data for the Data Model. It defines the expected maximum number of instances, as opposed to the enforced maximum cardinality. The following examples explain this: (a) On a relationship ***Places*** between ***Customer*** and ***Order***, it may be that there is a maximum of 100 ***Orders*** on the system for a ***Customer***. (b) On a ternary relationship between ***Customer***, ***Stock*** and ***Depot***, the value on the participation of ***Stock*** could be 20, indicating that from each ***Depot*** related to a ***Customer*** (i.e. each instance of the relationship), a maximum of twenty ***Stock*** items are ordered.
ALIASES	
CONSTRAINTS	The value shall not conflict with the value of *MaxInnerCardinality*.
DATA TYPE	Integer
DOMAIN	This shall be positive if specified.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MaxOuterCardinality**
CDIFMETAIDENTIFIER	1059
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the maximum number of instances of the *DataModelObject* acting as this *RolePlayer* that can be connected to a single instance of the *Relationship*.
USAGE	
ALIASES	
CONSTRAINTS	The value, if defined, shall be equal or greater than *MinOuterCardinality*.
DATA TYPE	String
DOMAIN	Numeric values are held as the appropriate digit string and "N" represents an undefined upper-bound.
LENGTH	10
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MinInnerCardinality**
CDIFMETAIDENTIFIER	1060
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the minimum number of instances of the *Relationship* related through this *RolePlayer* to a single instance of the *DataModelObject*. If the value is 0, it indicates that the *RolePlayer* is optional, and that therefore there can be instances of the *DataModelObject* that do not take part (acting as this *RolePlayer*) in any instances of this *Relationship*. If the value is non-zero, it indicates that each instance of the *DataModelObject* shall act as this *RolePlayer* in at least as many instances of this *Relationship* as the number specified. This defines the minimum cardinality of the *Relationship* from the viewpoint of the *DataModelObject*.
USAGE	This usually has a value of zero, where an object can exist without being required to take part in any *Relationships*. This is equivalent to the MetaAttribute called *IsOptional* having a value of ***TRUE***. It is non-zero (usually 1) where an object shall participate acting as the specified *RolePlayer* of the *Role* in the *Relationship*. For example, the detail *Entity* of a mandatory binary *Relationship*.
ALIASES	
CONSTRAINTS	Any specified value shall not conflict with the value of *IsOptional* or the value of *MinNumberOfOccurrences*.
DATA TYPE	String
DOMAIN	Numeric values are held as the appropriate digit string and "N" represents an undefined upper-bound.
LENGTH	10
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MinNumberOfOccurrences**
CDIFMETAIDENTIFIER	1061
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the minimum expected number of instances of the *Relationship* to which a single instance of the *DataModelObject* is attached.
USAGE	This is used to capture the volumetric data for the Data Model. It defines the expected minimum number of instances, as opposed to the enforced minimum cardinality. For example, on a binary relationship ***Places*** between ***Customer*** and ***Order***, it may be that there is a minimum of 3 instances of ***Places*** on the system for each ***Customer***.
ALIASES	
CONSTRAINTS	This shall not conflict with the value of *MinInnerCardinality*.
DATA TYPE	Integer
DOMAIN	This shall be zero or positive if specified.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**MinOuterCardinality**
CDIFMETAIDENTIFIER	1062
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the minimum number of instances of the *DataModelObject* related through this *RolePlayer* to a single occurrence of the *Relationship*. If the value is 0, it indicates that the *RolePlayer* is optional, and that therefore there can be instances of the *Relationship* that do not have any object acting as this *RolePlayer*. If the value is non-zero, it indicates that each instance of the *Relationship* shall have at least the number specified of related object instances in this *RolePlayer*.
USAGE	
ALIASES	
CONSTRAINTS	When a binary relationship is being represented, the *MinOuterCardinality* shall be non-zero, since a value of zero would indicate that relationship instances connected only to a single object could exist.
DATA TYPE	String
DOMAIN	Numeric values are held as the appropriate digit string and "N" represents an undefined upper-bound.
LENGTH	10
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**NumberOfDeletions**
CDIFMETAIDENTIFIER	1063
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of deletions for an *Entity* or *Relationship* instance from the related *Relationship* in the time period specified in the meta-attribute *DeletionTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued unless a value is provided for *DeletionTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**NumberOfInsertions**
CDIFMETAIDENTIFIER	1064
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of insertions for an *Entity* or *Relationship* instance into the related *Relationship* in the time period specified in the meta-attribute *InsertionTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued unless a value is provided for *InsertionTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**NumberOfReads**
CDIFMETAIDENTIFIER	1065
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of reads for an *Entity* or *Relationship* instance from the related *Relationship* in the time period specified in the meta-attribute *ReadTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued unless a value is provided for *ReadTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**NumberOfUpdates**
CDIFMETAIDENTIFIER	1066
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the expected number of updates for the *Entity* or *Relationship* instance from the related *Relationship* in the time period specified in the meta-attribute *UpdateTimePeriod*.
USAGE	
ALIASES	
CONSTRAINTS	This shall not be valued unless a value is provided for *UpdateTimePeriod*.
DATA TYPE	Float
DOMAIN	
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**ReadTimePeriod**
CDIFMETAIDENTIFIER	1067
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines a time period for the *NumberofReads* meta-attribute.
USAGE	
ALIASES	
CONSTRAINTS	This shall be valued if a value is provided for *NumberOfReads*.
DATA TYPE	Enumerated
DOMAIN	Millisecond, Second, Minute, Hour, Day, Week, Month, Year.
LENGTH	
ISOPTIONAL	True

META-ATTRIBUTE DEFINITION	META-ATTRIBUTE OF RolePlayer
NAME	**UpdateEffect**
CDIFMETAIDENTIFIER	1068
SUBJECTAREANAME	DataModels
SUBJECTAREAVERSION	15476-4:2005
DESCRIPTION	This defines the effect of an Update operation on both the *CandidateKey* entity instance and the *ForeignKey* entity instance. If the entity instance acting as the *RolePlayer* is a *CandidateKey* instance then: • RESTRICTS The operation cannot complete if there are matching *ForeignKey* entity instances. • CASCADES The operation will update *ForeignKey* entity instances. For example, for the relationship ***Customer.Places.Order***, if the ***Customer's*** candidate key is updated then all ***Order*** instances will have their *ForeignKey* values correspondingly updated. • SETNULL The update operation will cause the values of referencing *ForeignKeys* to be set to NULL. This effectively breaks the relationship between the candidate key entity instance and the *ForeignKey* entity instances. • SETDEFAULT The update operation will cause the values of referencing *ForeignKeys* to be set to default values as specified by the meta-attribute *DefaultValue*. If the entity instance acting as the *RolePlayer* is a *ForeignKey* instance then: • RESTRICTS The operation cannot complete if there are matching *CandidateKey* entity instances. • CASCADES The operation will update the attribute components in the *CandidateKey* entity instances.
USAGE	This ensures referential integrity when a referenced *Entity* instance is updated.
ALIASES	
CONSTRAINTS	
DATA TYPE	Enumerated
DOMAIN	RESTRICTS, CASCADES, SETNULL, SETDEFAULT
LENGTH	
ISOPTIONAL	True