miUML Metamodel Class Descriptions
Identifier Subsystem

Leon Starr

Tuesday, May 31, 2011

mint.miUMLmeta.td.9

Version 1.2.0

Table of Contents

Introduction

For Association Class and Generalization Relationships there are certain rules that determine the uniqueness constraints that must be applied to certain groups of Referential Attributes. The bulk of these are nicely defined in the classic [SM1]. In fact, the rules are nothing more than the application of BCNF (3rd normal form) relational principles. It is best to apply these rules automatically rather than relying on the modeler or model editor's knowledge (or lack thereof) of THE RULES. Even if the modeler is well familiar with these rules, it still can be tedious to constantly specify them by hand.

The purpose then, of the Identifier Subsystem, is to model the constraints necessary to automatically construct and verify standard Identifiers from the Referential Attributes generated for each Relationship type.

Great care is taken to ensure that there is no over-specification such that the modeler is hindered from performing any legal, but non-obvious, referential attribute and identification trickery. Only inviolable rules are enforced.

Key to Styles and Conventions

COLOR is used to convey information in this document. If you have the ability to print or display this document in color, it is highly recommended. Here are a few notes on the font, color and other styles used in class model descriptions.

Naming classes, types and subsystems

Modeled elements such as classes, types and subsystems are written with initial caps. The text Air Traffic Controller, for example, would probably be a class. If you see the text Application Domain, you know it is a modeled element, most likely a domain.

Naming attributes

Attributes are named with an initial cap followed by lower case, Time_logged_in, for example.

Instance values

Instance data, such as attribute values, are either in quote marks as in "Gwen" and "Owen" or represented in the following color/font: Gwen and Owen.

Whitespace

When programming, the squashedNamingStyle is optimal for easy typing and is adequately readable for short names. That's fine for programming, but analysis is all about easy readability and descriptive, sometimes verbose names. Programming style is retained for method names, but everywhere else whitespace or underscores are employed. For example: Air Traffic Controller, Time logged in or On_Duty_Controller.Time_logged_in.

hardcoreJavaProgrammersMayDisputeThisPointAndArgueThatWhiteSpacesAreEntirelyUnnecessaryForImprovingReadabilityButIdisagree.

Attribute heading color

Attribute headings are colored according to the role the attribute plays in its class. Referential attributes are brown, **Logged in controller**, for example. Naming attributes are blue, **Number**, for example. Finally, descriptive attributes are red, **Time logged in**, for example. Derived attributes are purple **\Volume**, for example.(One of the nice things about color is that it helps eliminate the need for underscores).

Referential renaming

A referential attribute will often have a name that reflects its role rather than the name of the base reference. Station.Logged_in_controller may refer to On_Duty_Controller.ID, for example. Such renaming will be defined in the referential attribute's description.

MDA (Model Driven Architecture) Layers

The MDA defined layers are referenced from time to time. It is sometimes useful, especially in a metamodel, to distinguish the various meta layer names: M0, M1, M2 and M3. These are M0: data values ('N3295Q', 27L, 490.7 Liters, ...) M1: domain specific model elements (class 'Aircraft', attribute 'Altitude', relationships 'R2 - is piloted by'), M2: metamodel elements to define miUML (class: Class, class: Attribute, class: Relationship, attribute: 'Rnum'...) and M3: meta-metamodel elements to define (ambitiously) any modeling language (class 'Model Node', class 'Structural Connection'). For our purposes, we will seldom refer to the M3 layer and M2 will be relevant only if the subject matter at hand is a metamodel.

Keep in mind that these layers may be interpreted relative to the subject matter at hand. If you are modeling an air traffic control system, 490.7 Liters is likely data in the M0 layer with the class 'Aircraft' at the M1 layer. 'Class' would be at M2 - the metamodel layer.

But if the subject matter is Executable UML, as is the case in the miUML metamodels, both the class 'Aircraft' and 490.7 Liters could be compressed into the M0 layer. ('Aircraft' is data populating the Class and Attribute subsystem and 490.7 Liters populated into the Population subsystem). At the M1 layer we would possibly have 'Class' and 'Value'. And, since the metamodel will populate itself, as we bootstrap into code generation, the M2 layer is also 'Class' and 'Value'.

In other words, we can create a data base schema from the miUML metamodel and then insert the metamodel itself into that schema. So if we define a table 'Class' based on our metamodel in the data base, we could then insert metamodel classes 'Class', 'Attribute', etc. into that table. Thus we see 'Class' both at the M0 (inserted data) and M1 (modeled) and M2 (metamodel) layers!

Local Data Types

The following data types are used by attributes in this subsystem.

Name

This is a string of text that can be long enough to be readable and descriptive. A handful of words is okay, a full tweet is probably too much. Since the exact value may change to suit a particular platform, it is represented symbolically as "LONG_TEXT".

Domain: String [LONG_TEXT]

Nominal

This is a whole number used purely for naming with no ordinal or computational properties.

Identifiers used only for referential constraints may be stripped out during implementation (assuming the constraints are still enforced). During simulation and debugging, however, these can be nice to have around for easy interpretation of runtime data sets without the need for elaborate debug environments. "File-6:Drawer-2:Cabinet-"Accounting" is easier for a human to read than a bunch of pointer addresses.

Domain: The set of positive integers {1, 2, 3, …}.

Description

This is an arbitrarily long amount of text. The only length limit is determined by the platform.

References

[MB] Executable UML: A Foundation for Model-Driven Architecture, Stephen J. Mellor, Marc J. Balcer, Addison-Wesley, 2002, ISBN 0-201-74804-5

[DATE1] An Introduction to Database Systems, 8th Edition, C.J. Date, Addison-Wesley, 2004, ISBN 0-321-19784-4

[LSART] How to Build Articulate UML Class Models, Leon Starr, Model Integration, LLC, Google Knol, http://knol.google.com/k/how-to-build-articulate-uml-class-models

>> Associative Reference

IMPORTED

From Formalization Subsystem, td.6.

RELATIONSHIPS

R50

Associative Reference IS A To Many in Mx:Mx, To Many in 1x:Mx, To One in 1x:1x OR A To One in 1x:Mx Associative Reference

The role of an Associative Reference in an RR Associative Identifier depends on two things. It depends on the full multiplicity of the abstracted Association, without regard to conditionality (1x:1x, 1x:Mx, Mx:Mx) and it depends on the multiplicity of the Perspective to which the Associative Reference is referring. The four subclasses result from this combination.

In an Mx:Mx Association, each Associative Reference refers to an Mx side yielding two To Many in Mx:Mx Associative References.

In a 1x:1x Association each Associative Reference refers to a 1x side yielding two To One in 1x:1x Associative References.

This only leaves the 1x:Mx multiplicity pair which yields a single To One and a single To Many in 1x:Mx Reference.

Required Referential Identifier RRI

Each type of Relationship results in the systematic placement of References. The type of Relationship and, in the case of an Association, multiplicity, automatically determines the eligibility of each Reference as an Identifier or a component of an Identifier. For example, a Reference from a Subclass to its Superclass always forms a *Required Referential Identifier*. As another example, both T and P References in an Association Class, taken together, form a single complete Required Referential Identifier. Always.

This type of Identifier is 'required' because the modeler may not choose to ignore it or modify it without violating relational normalization rules. During model editing, when a new Relationship is created, the relevant Required Referential Identifiers should be automatically defined without any need for input from the modeler.

The modeler is, of course, free to add any additional Modeled Identifiers which may or may not incorporate Referential Attributes as appropriate. But the modeler may not delete or alter any Required Referential Identifiers without altering the Relationship formalized by the underlying References.

At least one Executable UML modeling tool, at the time of this writing and for many years prior, has routinely generated Required Referential Identifiers incorrectly. This in turn has caused unnecessary confusion for modelers using that tool, not realizing that the tool is consistently wrong. That was part of the motivation for modeling the rules explicitly.

Relationships

R51

Required Referential Identifier IS AN RR Subclass, RR Associative OR AN RR To One Unconditional Identifier

By 'required' it is meant that an Identifier is known to be necessary regardless of the modeler's intent. In other words, relational normalization rules dictate that certain Identifiers are automatically created from configurations of References. By abstracting RR Identifiers, it is possible to save the modeler some effort by consistently generating them correctly.

There are three cases where RR Identifiers are necessary. The Superclass Reference in a Subclass always forms an Identifier. A Reference to a one unconditional Perspective on a 1x:1x non-associative Association is always an Identifier. Finally, a Reference in an Association Class participates or does not participate in an RR Identifier depending on the multiplicity pair on the formalized Association and to which Perspective the Reference is pointing.

RR Associative Identifier RRA_ID

An Association that uses an Association Class (associative) is always formalized with two References, T and P, regardless of multiplicity, reflexivity or symmetry. Depending on the multiplicity, either one or two *RR Associative Identifiers* are formed.

The role of a T or P Reference in an RR Associative Identifier is determined systematically from the multiplicities on each Perspective of the Association formalized by the Association Class.

Attributes

Number

Required Referential Identifier.Number

Association class

Required Referential Identifier.Class — Must be a Class playing the role of an Association Class.

Domain

Required Referential Identifier.Domain

Identifiers

1> Number + Association class + Domain

Same as the superclass.

Relationships

R52
RR Associative Identifier IS AN RR Associative MM, M OR 1 Identifier

The relational rules for composing an RR Associative Identifier vary based on the multiplicity of the Perspectives formalized by the Association Class as follows:

1x:1x :: Each Reference constitutes a distinct RR Associative Identifier.

1x:Mx :: The Reference to the Many Perspective constitutes an RR Associative Identifier.

Mx:Mx :: Both References combined form a complete RR Associative Identifier.

These three rules are a product of relational theory that ensure proper third normal form.

RR Associative 1 Identifier RRA1_ID

An *RR Associative 1 Identifier* is formed from either the T or P Reference in an Association Class formalizing a 1x:1x Association.

Attributes

Number

RR Associative Identifier.Number

Association class

RR Associative Identifier.Association class and To One in 1x:1x Associative Reference.Association class — Both the Identifier and Reference must be in the same Class.

Domain

RR Associative Identifier.Domain and To One in 1x:1x Associative Reference.Domain — Both the Identifier and Reference are in the same Class and, hence, Domain.

Type

To One in 1x:1x Associative Reference.Type

Referenced class

To One in 1x:1x Associative Reference.Participating class

Rnum

To One in 1x:1x Associative Reference.Rnum — The Association formalized by the Association Class.

Identifiers

1> Number + Association class + Domain

Same as the superclass.

2> Type + Association class + Referenced class + Rnum + Domain

A reference to the other side of a 1:1 association is always an identifier. The Type (T or P) is required in case the formalized Association is reflexive with T and P referring to the same Class.

RELATIONSHIPS

R56

RR Associative 1 Identifier IS FORMED BY *exactly one* To One in 1x:1x Associative Reference

To One in 1x:1x Associative Reference FORMS *exactly one* RR Associative 1 Identifier

In a 1x:1x Association formalized with an Association Class, the Reference to each side, T and P, forms an Identifier in its own right. Consequently, the formalizing Association Class must have two RR Associative Identifiers. (In a Unary Association, both Symmetric Associative References will point to the same Symmetric Perspective).

By definition, an RR Associative 1 Identifier is created from a To One in 1x:1x Associative Reference.

RR Associative M Identifier RRAM_ID

An *RR Associative M Identifier* is formed from the Reference to the Many Perspective in an Association Class formalizing a 1x:1x Association.

Attributes

Number

RR Associative Identifier.Number

Association class

RR Associative Identifier.Association class and To Many in 1x:Mx Associative Reference.Association class — Both the Identifier and Reference must be in the same Class.

Domain

RR Associative Identifier.Domain and To Many in 1x:Mx Associative Reference.Domain — Both the Identifier and Reference are in the same Class and, hence, Domain.

Type

To Many in 1x:Mx Associative Reference.Type

Referenced class

To Many in 1x:Mx Associative Reference.Participating class

Rnum

To Many in 1x:Mx Associative Reference.Rnum — The Association formalized by the Association Class.

Identifiers

1> Number + Association class + Domain

Same as the superclass.

2> Type + Rnum + Association Class + Referenced class + Domain

A reference to the other side of a 1:1 association is always an identifier. The Type (T or P) is required in case the formalized Association is reflexive with T and P referring to the same Class.

Relationships

R55

RR Associative M Identifier IS FORMED BY *exactly one* To Many in 1x:Mx Associative Reference

To Many in 1x:Mx Associative Reference FORMS *exactly one* RR Associative M Identifier

In a 1x:Mx Association formalized with an Association Class, the Reference to the Many Perspective creates an Identifier. This will be the one and only RR Associative Identifier in the Association Class. (The same Class may not play more than one Association Class role, so there really cannot be any RR Associative Identifiers in the Class).

By definition, an RR Associative M Identifier is created from a To Many in 1x:Mx Associative Reference.

RR Associative MM Identifier RRAMM_ID

An *RR Associative MM Identifier* is formed from both the T and P Reference in an Association Class formalizing an Mx:Mx Association.

Attributes

Number

RR Associative Identifier.Number

T class

To Many in Mx:Mx Associative Reference.Participating class — This is the Class referred to by the T Reference.

P class

To Many in Mx:Mx Associative Reference.Participating class — This is the Class referred to by the P Reference.

Rnum

To Many in Mx:Mx Associative Reference.Rnum — The Association formalized by the Association Class. Both T and P sides are combined into the same value since they must be on the same Association.

Association Class

RR Associative Identifier.Association class and To Many in Mx:Mx Associative Reference.Association class for both the T and P References — The Identifier and its component T and P References are all in the same Association Class.

Domain

RR Associative Identifier.Domain and To Many in Mx:Mx Associative Reference.Domain

Identifiers

1> Number + Association class + Domain

Same as the superclass.

2> T class + Rnum + Association class + Domain

A reference to a 1 side of a 1:1 association is an identifier.

3> P class + Rnum + Association class + Domain

A reference to a 1 side of a 1:1 association is an identifier,.

Relationships

R54

RR Associative MM Identifier INCLUDES *exactly one* To Many in Mx:Mx Associative T Reference

To Many in Mx:Mx Associative T Reference IS COMPONENT OF *exactly one* RR Associative MM Identifier

In an Mx:Mx Association formalized with an Association Class, the Reference to each side, T and P, are both combined to form a single RR Associative Identifier. This will be the one and only RR Associative Identifier in the Association Class.

By definition, an RR Associative MM Identifier created from a pair of T and P Many in Mx:Mx Associative References.

R57

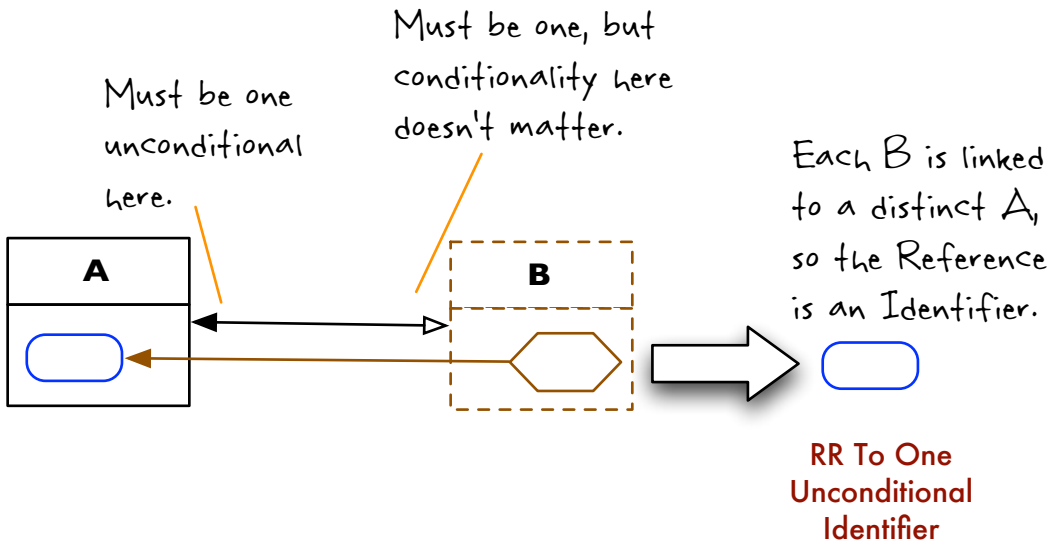
RR Associative MM Identifier INCLUDES *exactly one* To Many in Mx:Mx Associative P Reference

To Many in Mx:Mx Associative P Reference IS COMPONENT OF *exactly one* RR Associative MM Identifier

See description for R54.

RR To One Unconditional Identifier RR1U_ID

An *RR To One Unconditional Identifier* is formed from the Reference in one Class to the opposite Perspective in an Association that is not formalized using an Association Class.



As shown above, the referring Perspective is one conditional (0..1) or unconditional (1) while the referenced Perspective must be one unconditional (1). A To One Unconditional Reference could participate in either a 1:1x or 1:Mx Association. But it will only form a Required Referential Identifier in the 1:1x case.

Number

Required Referential Identifier.Number

From class

Required Referential Identifier.Class and To One Reference.From class — Both the Identifier and Reference must be in the same Class.

To class

To One Reference.To class — The referenced Class.

Rnum - constrained

To One Reference.Rnum — This value is constrained such that the referring Perspective is 1x and the referenced Perspective is 1 (1x:1).

Domain

Required Referential Identifier.Domain and To One Unconditional Reference.Domain — Both the Identifier and Reference are in the same Class and, hence, Domain.

Identifiers

1> Number + From class + Domain

Same as the superclass.

2> From class + To Class + Rnum + Domain

A reference to the other side of a 1:1 association is always an identifier.

R54
RR To One Unconditional Identifier IS FORMED BY *exactly one* To One Reference
To One Reference FORMS *zero or one* RR To One Unconditional Identifier

A Reference to an unconditional One Perspective always refers to an Identifier value (as opposed to the value none) on that Perspective. If the Association is 1:1x, the referring Instance is linked to one Instance and no other referring Instances may link to that same Instance, so this value will be unique.

On the other hand, if the Association is 1:Mx, multiple referring Instances may link to the same Instance on the unconditional One Perspective disqualifying the Reference from constituting an Identifier.

A Reference to a conditional One Perspective (0..1) will hold the none value when there there is no Link present across the Association. Since multiple Instances on the referring side could also be unlinked, multiple none values could be present disqualifying the Reference as an Identifier.

By definition, an RR To One Unconditional Identifier is formed from an eligible To One Unconditional Reference in the same Class.

RR Subclass Identifier RRSUB_ID

An *RR Subclass Identifier* is formed from the Superclass Reference in a Subclass.

Attributes

Number

Required Referential Identifier.Number

Subclass

Required Referential Identifier.Class — This must be a Specialized Class playing the role of a Subclass.

Superclass

Superclass Reference.Superclass

Rnum

Superclass Reference.Rnum

Domain

Required Referential Identifier.Domain

Identifiers

1> Name + Class + Domain

Same as the superclass.

2> Subclass + Superclass + Rnum + Domain

A reference to a 1 side of a 1:1 association is an identifier.

Relationships

R53
RR Subclass Identifier IS FORMED BY *exactly one* Superclass Reference
Superclass Reference FORMS *exactly one* RR Subclass Identifier

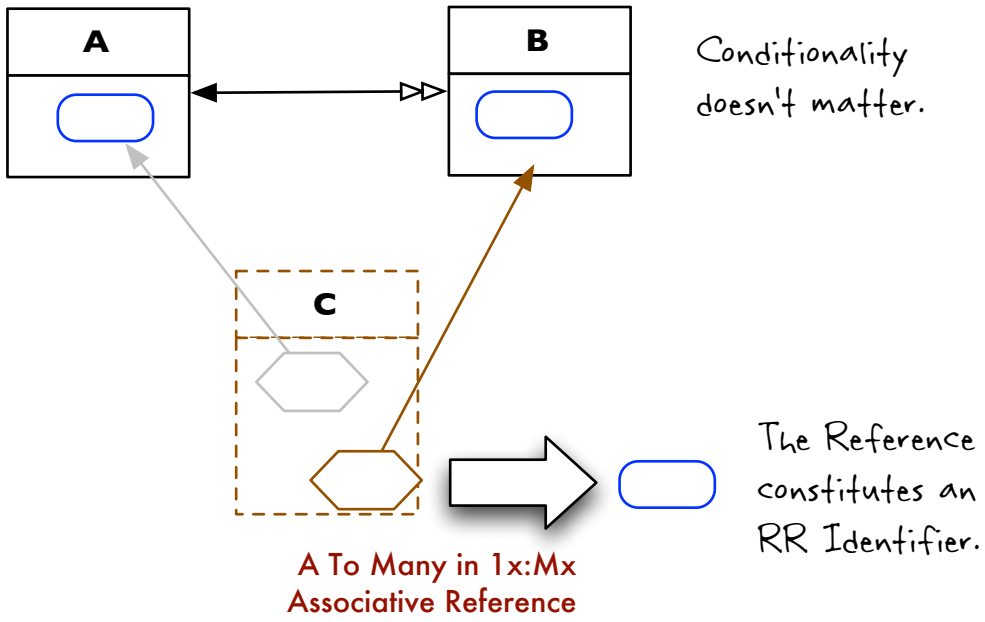
A Generalization is partially formalized by the Reference from a Subclass to its Superclass. This Reference always forms a Required Referential Identifier in the Subclass.

By definition, an RR Subclass Identifier is formed from the Superclass Reference inside its Subclass.

To Many in 1x:Mx Associative Reference

M-1xMx_AR

The Reference inside an Association Class to a many Perspective of a 1x:Mx Association formalized by that Association Class is a *To Many in 1x:Mx Association Reference*.



An RR Identifier is fully created by the Reference to the many Perspective.

Attributes

Type

Associative Reference.Type

Association class

Associative Reference.Association class

Participating class

Associative Reference.Participating class

Rnum

Associative Reference.Rnum

Domain

Associative Reference.Domain

Identifiers

1> Type + Association class + Participating class + Rnum + Domain

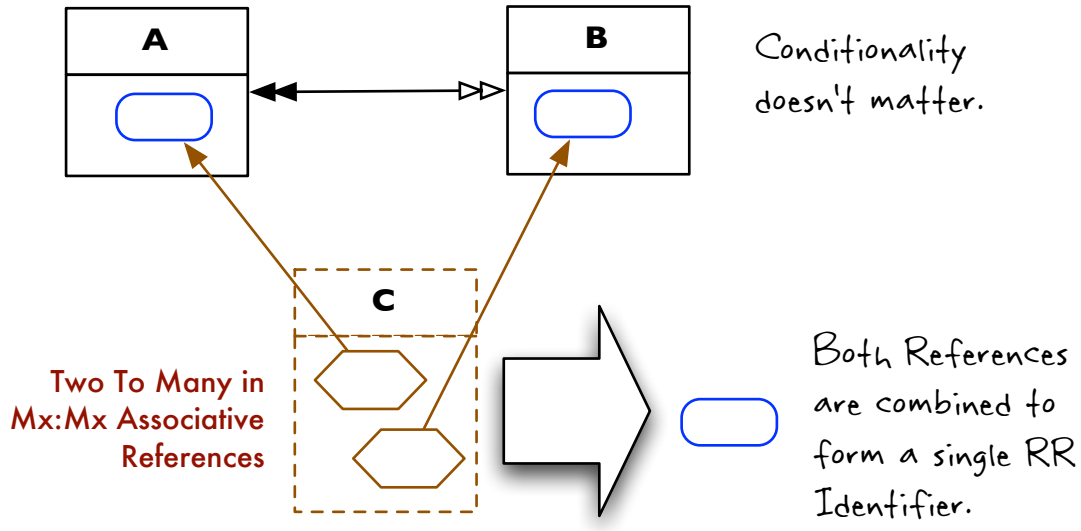
Same as superclass.

Relationships

None.

To Many in Mx:Mx Associative Reference
M-MxMx_AR

Each Reference inside an Association Class to a Many Perspective of an Mx:Mx Association formalized by that Association Class is a *To Many in Mx:Mx Association Reference*.



The two References (T and P) inside the Association Class, taken together, form a complete RR Identifier of the Association Class.

Attributes

Type

Associative Reference.Type

Association class

Associative Reference.Association class

Participating class

Associative Reference.Participating class

Rnum

Associative Reference.Rnum

Domain

Associative Reference.Domain

Identifiers

1> Type + Association class + Participating class + Rnum + Domain

Same as superclass.

Relationships

R58

To Many in Mx:Mx Associative Reference IS A To Many in Mx:Mx Associative T or P Reference

Since each Associative Reference is either a T or P Reference, and a To Many in Mx:Mx Associative Reference is an Associative Reference, it follows that each To Many in Mx:Mx Associative Reference is either T or P.

An RR Identifier is created from the pair of T and P References inside an Association Class that formalizes an Mx:Mx Association. To specify the necessary pairing, it is helpful to break them out into separate subclasses.

To Many in Mx:Mx Associative P Reference
M-MxMx_PAR

This is a To Many in Mx:Mx Associative Reference that is also a P Reference.

Attributes

Association class

To Many in Mx:Mx Associative Reference.Association class

Participating class

To Many in Mx:Mx Associative Reference.Participating class

Rnum

To Many in Mx:Mx Associative Reference.Rnum

Domain

To Many in Mx:Mx Associative Reference.Domain

Type

To Many in Mx:Mx Associative Reference.Type

Identifiers

1> Association class + Participating class + Rnum + Domain

Same as superclass with the Type attribute omitted since the Type value is always P.

Relationships

None.

To Many in Mx:Mx Associative T Reference
M-MxMx_TAR

This is a To Many in Mx:Mx Associative Reference that is also a T Reference.

Attributes

Association class

To Many in Mx:Mx Associative Reference.Association class

Participating class

To Many in Mx:Mx Associative Reference.Participating class

Rnum

To Many in Mx:Mx Associative Reference.Rnum

Domain

To Many in Mx:Mx Associative Reference.Domain

Type

To Many in Mx:Mx Associative Reference.Type

Identifiers

1> Association class + Participating class + Rnum + Domain

Same as superclass with the Type attribute omitted since the Type value is always T.

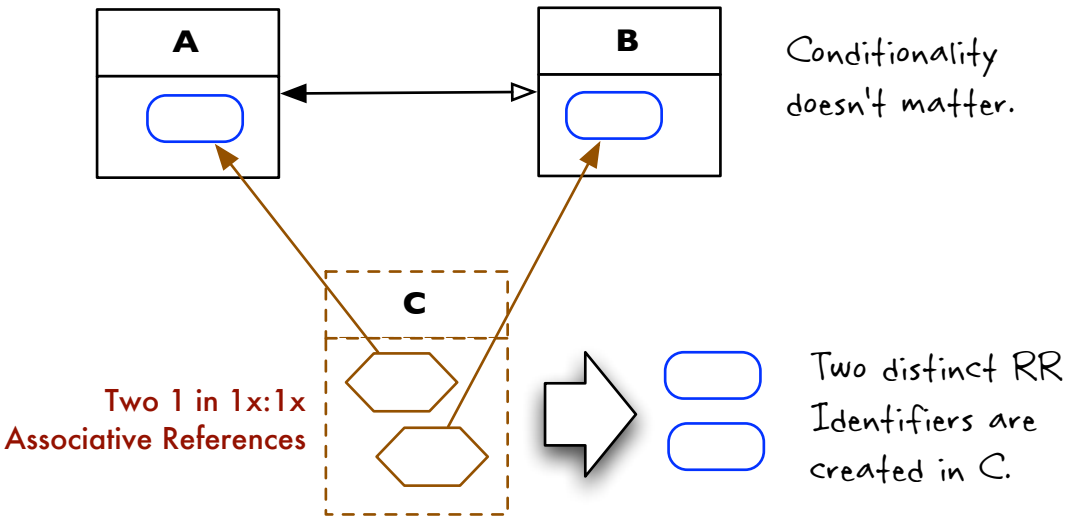
Relationships

None.

To One in 1x:1x Associative Reference

1-1x1x_AR

Each Reference (T and P) inside an Association Class formalizing a 1x:1x Association is a *To One in 1x:1x Associative Reference*.



A distinct RR Identifier is created for each of these References as shown above.

Attributes

Type

Associative Reference.Type

Association class

Associative Reference.Association class

Participating class

Associative Reference.Participating class

Rnum

Associative Reference.Rnum

Domain

Associative Reference.Domain

Identifiers

1> Type + Association class + Participating class + Rnum + Domain

Same as superclass.

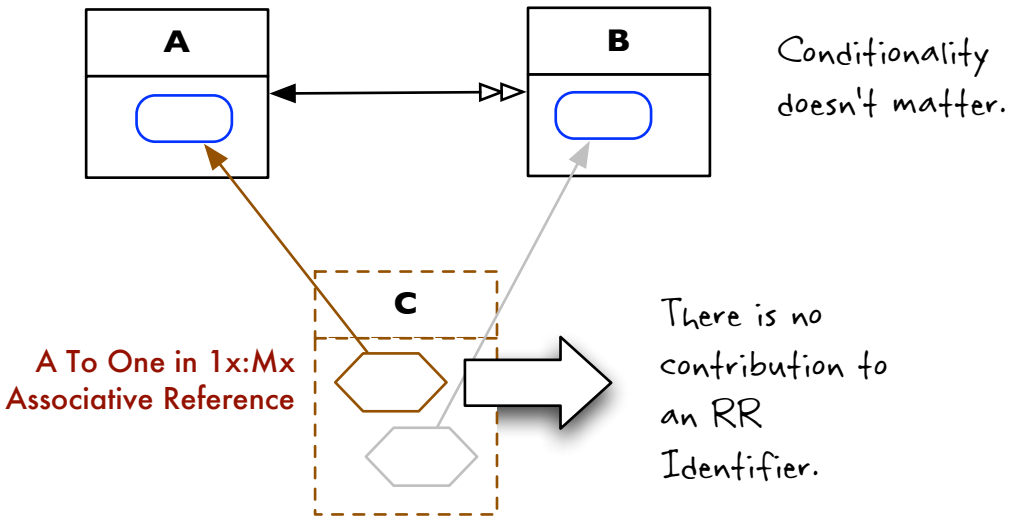
Relationships

None.

To One in 1x:Mx Associative Reference

1-1xMx_AR

The Reference inside an Association Class referring to the one multiplicity on a 1x:Mx Association is a *To One in 1x:Mx Association Reference*.



This Reference does not participate in any RR Associative Identifier. That is because multiple Instances of the Association Class will hold the same value for the same referenced Instance on the One Perspective.

Attributes

Type

Associative Reference.Type

Association class

Associative Reference.Association class

Participating class

Associative Reference.Participating class

Rnum

Associative Reference.Rnum

Domain

Associative Reference.Domain

Identifiers

1> Type + Association class + Participating class + Rnum + Domain

Same as superclass.

Relationships

None.