Shared Information/Data (SID) Model

Addendum U –Using the SID

Release 6.0

TeleManagement FORUM

GB922 –1U

Version 6.4 *November, 2005*

Notice

No recipient of this document shall in any way interpret this document as representing a position or agreement of TM Forum or its members. This document is a draft working document of TM Forum and is provided solely for comments and evaluation. It is not a Forum Approved Document and is solely circulated for the purposes of assisting TM Forum in the preparation of a final document in furtherance of the aims and mission of TM Forum.

Although it is a copyrighted document of TM Forum:

- Members of TM Forum are only granted the limited copyright waiver to distribute this document within their companies and may not make paper or electronic copies for distribution outside of their companies.
- Non-members of the TM Forum are not permitted to make copies (paper or electronic) of this draft document other than for their internal use for the sole purpose of making comments thereon directly to TM Forum.
- If this document forms part of a supply of information in support of an Industry Group Liaison relationship, the document may only be used as part of the work identified in the Liaison and may not be used or further distributed for any other purposes

Any use of this document by the recipient, other than as set forth specifically herein, is at its own risk, and under no circumstances will TM Forum be liable for direct or indirect damages or any costs or losses resulting from the use of this document by the recipient.

This document is governed by all of the terms and conditions of the Agreement on Intellectual Property Rights between TMForum and its members, and may involve a claim of patent rights by one or more TMForum members or by non-members of TMForum.

Direct inquiries to the TM Forum office:
240 Headquarters Plaza,
East Tower – 10th Floor,
Morristown, NJ 07960 USA
Tel No. +1 973 944 5100
Fax No. +1 973 944 5110
TM Forum Web Page: www.tmforum.org

Table of Contents

List of Figures

1. Extending the SID Business View

1.1. Introduction

The SID is being constructed as a framework of frameworks. For example, the core framework consists of a set of root business entities [Addendum 1R], which serve to anchor entities in each domain. This presents a unified class hierarchy for the system and other views in the NGOSS Lifecycle [Life]. The framework of frameworks approach provides inherent extensibility and flexibility. The SID program will be describing and documenting new functionality during each phase, so it is important to enable this new functionality to be easily added and extended without adversely affecting the overall SID model.

This Addendum provides guidelines and examples of how to define extensions to the SID. As a by-product, it also provides guidelines for developing ABEs[1] that have not yet been modeled by the SID team. Use of these guidelines will enable extensions made by different people and organizations to have the same structure, thus enabling these extensions to be compatible with each other as well as with the SID itself. It also enables such extensions to be proposed back to the SID team for official incorporation as part of the SID.

The guidelines presented here include:

- Creating a model or model packages[2] from the SID model
- Patterns and rules for extending existing business entities
- Patterns for new ABE development (which are used to design model packages and elements)
- Association, attribute, and package naming conventions
- General modeling guidelines

The remainder of this document will use “SID model” or “SID framework” to refer to GB922 and its Addenda, which collectively define the NGOSS Business view of the SID.

1.2. Creating a Model or Model Packages From the SID Model

This section describes how to create a new model using the SID model as a foundation and how to add SID model ABEs to an existing model. The Rational Rose import feature is used to

[1] An ABE is an Aggregated Business Entity – a collection of common classes that model a set of related concepts. Please see GB922, Concepts and Principles, for more information.

[2] A “package” is a general-purpose modeling construct that organizes model elements into cohesive groups.

either create a new model or expand an existing model. Of course, the SID model can be used as-is and then extended by following the extending the SID model guidelines in this chapter.

1.2.1. Getting Started

A review of the Control Units (Overview) and Virtual Path Maps (Overview) help topics in Rational Rose[3] will provide you with an understanding of the Rational Rose features that will be used during the modeling process. It is also recommended that you experiment with control units and virtual path maps before beginning the modeling process for real. (Note: Because of a bug in Rational Rose you must create the path before opening a model. How to create a path map is explained in this chapter.)

MAKE A COPY OF YOUR MODEL BEFORE YOU START ANY OF THIS WORK. Mistakes happen, so you need to have a model to which you can fall back in case something goes wrong. SAVE OFTEN. This enables you to simply restart the model[4] if you want to revert to a known good copy of the model.

1.2.2. Model Structure

The SID model framework in Rational Rose should resemble that depicted in the figure below. The SID is constructed using two types of files. The SID itself is organized into a "mdl" file. In the SID, this contains a set of directives of which files to load, plus additional graphical presentation information. The SID model content itself is organized into a set of "cat" files that are used to describe specific knowledge domains in the SID.

[3] Go to the Help menu when Rational Rose is launched , choose "Search for Help On…" and type in these topics.
[4] Go to the File menu, select the name of the model that you were working on, choose "No" to the Save Changes dialog, and then reload the model.

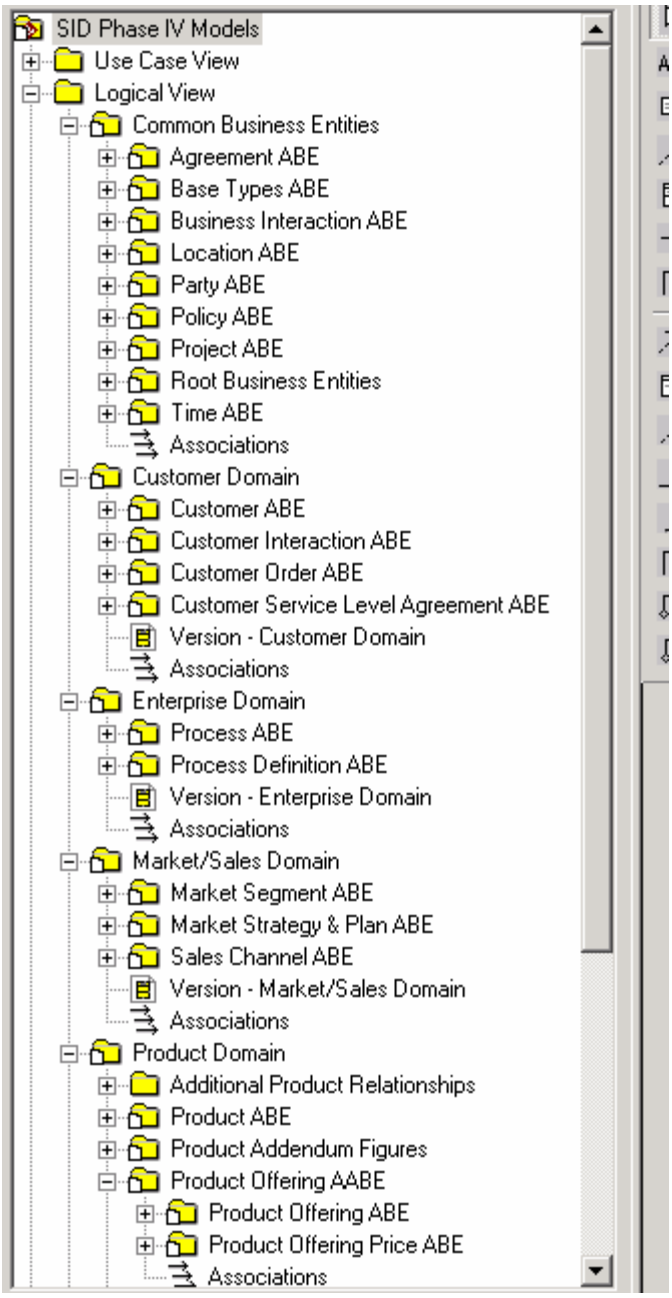

Figure U. 1 - SID Model Framework in Rose

Each ABE package name in the model should have "ABE" as a suffix. Each package should also be stored as a control unit. Note: This convention is not used in all the figures in this document.

It is also convenient, but not mandatory, to have a package (folder) in each ABE that contains the figures that describe the model. Doing this provides for a better organization of the model. For example, in the Business Interaction ABE package there is a package labeled "Business Interaction Addendum Figures" as shown in the figure below. It is not necessary, but is a consistent naming convention, to have the name of the ABE as part of the package name that contain the figures. In Figure U. 2, the Business Interaction Business Entities class diagram is

not shown in the Business Interaction Addendum figures package because it is present as a "working diagram". Other working diagrams that are not contained in the addendum can be kept in the ABE folder, also shown in the figure below.

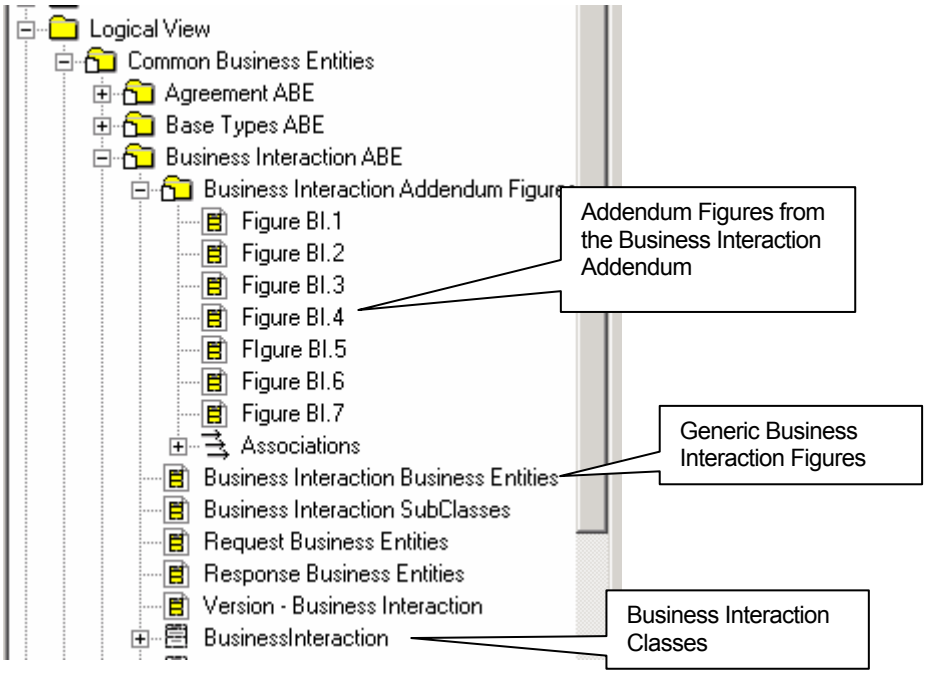

Figure U. 2 - Addendum Figures in Rose

1.2.3. Starting a Model From Scratch

To start a model from scratch, all that has to be done is open up the SID model. Explore the ABEs of interest and their associated addenda. Next, begin extending the SID using the guidelines presented in this addendum.

1.2.4. Organizing an Existing Model Using the SID Framework

The first step is to introduce the SID framework as shown in Figure U. 1 - SID Model Framework in Rose. Then move your application-specific Rose classes, associations, and diagrams into the appropriate ABEs within the SID framework.

Rational Rose control units help organize a model. Each ABE sub-model is defined as a control unit[5]. Addendum figure packages, if used, should also be defined as control units. Control units appear in the Rose model as shown by the circled areas in the figure below. Control units are annotated with a small white or gray package icon within the yellow package icon.

[5] A control unit is a file that contains all or part of a model. They enable a team of people working on the model to place different parts of it under version control, to support shared development.

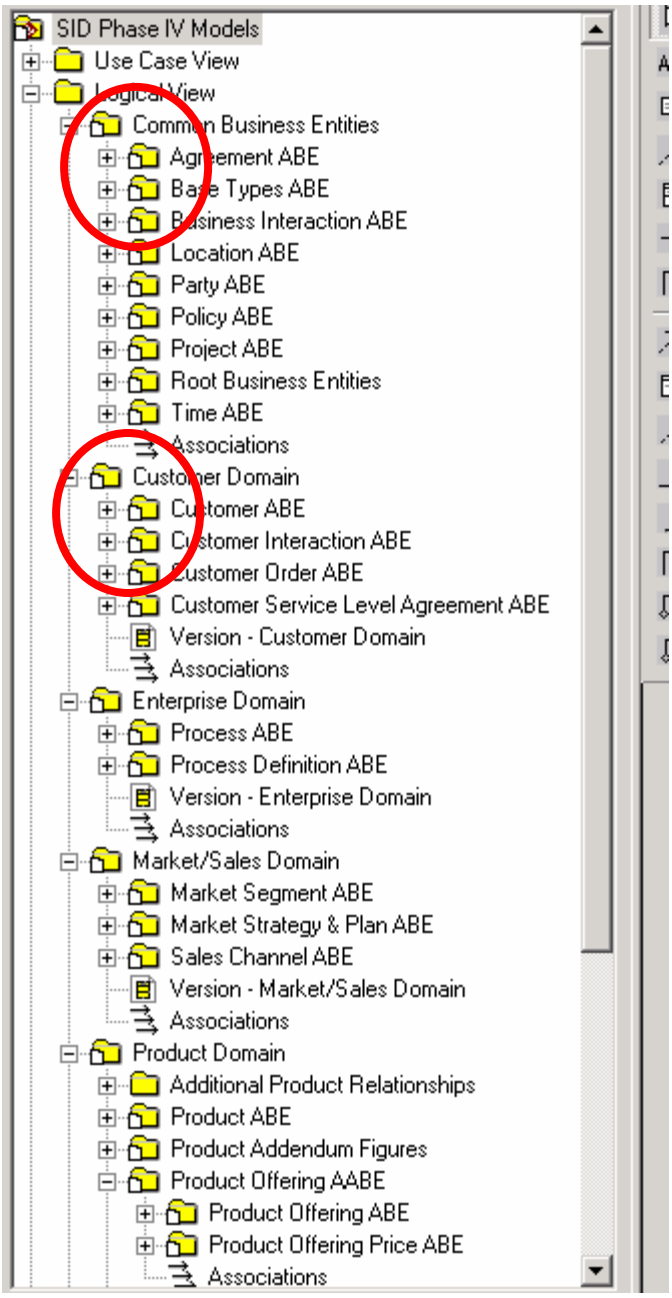

Figure U. 3 - Control Units in the Rose Model

Before defining control units for ABEs or using the SID model for the first time, it is wise to set up a virtual path map. (Because of a bug in Rose, you must create the path map before opening a model.) The virtual path specifies the relative address that contains the ABE control units. An example of virtual path map is shown in the figure below.

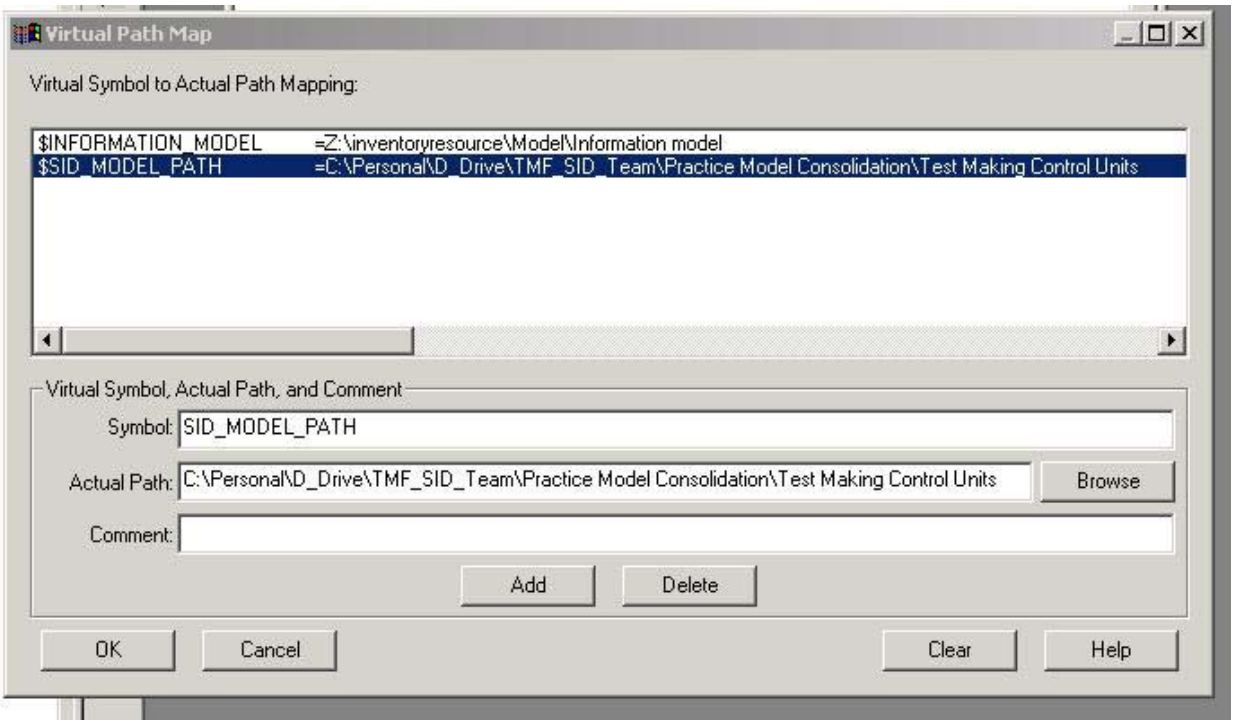

Figure U. 4 - Virtual Path Map Example

The actual path is the location in the user's machine where the various parts of the SID model are stored. Set up a folder that will contain the SID .mdl file, with a name such as SID_R4 Model. Then set up subfolders within it for each domain. Each subfolder will hold the respective domain's ABEs (please refer to GB922, Concepts and Principles, for a definition of the SID Business Domains and the top-level ABEs that they contain). Each subfolder's name is the domain name followed by the word "domain". For example, the common business entities sub-folder is used to contain all of the different Common Business Entities Domain ABEs. These domains names must be used consistently, as they will become part of the actual address of the ABE. The figure below shows the folder structure for the SID Model folder and all domains.

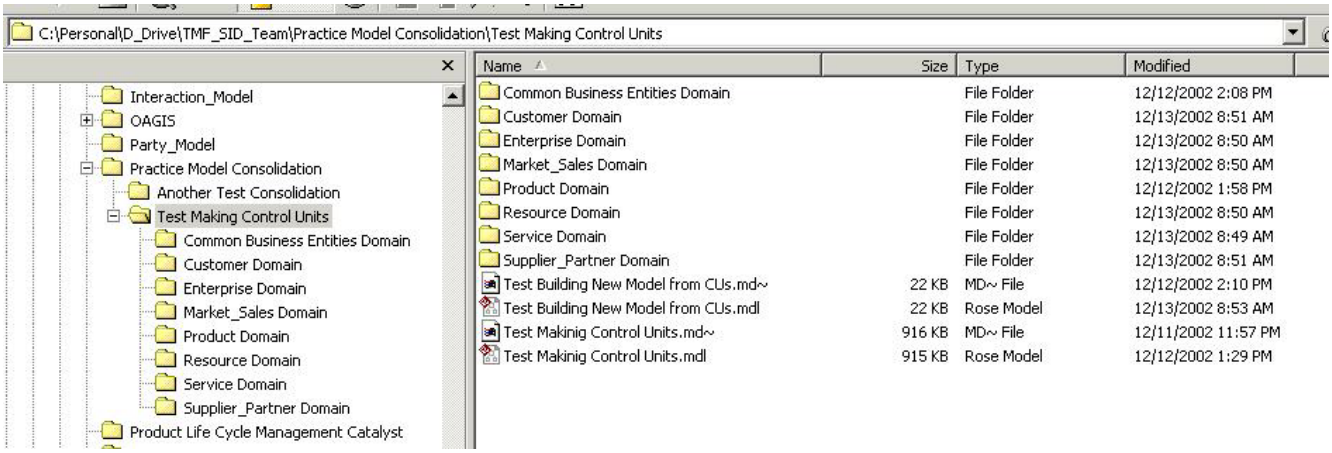

Figure U. 5 - Domain Folder Structure

The figure below contains an example of the way the address of an ABE is stored within the SID model.

```
          file_name     "$SID_MODEL_PATH\\Common Business Entities Domain\\Root
Business Entities ABE.cat"
          quid          "3CF68FE80295")
      (object Class_Category "Business Interaction ABE"
          is_unit       TRUE
          is_loaded     FALSE
          file_name     "$SID_MODEL_PATH\\Common Business Entities
Domain\\Business Interaction ABE.cat"
          quid          "3C18FFFD032B")
      (object Class_Category "Agreement ABE"
          is_unit       TRUE
          is_loaded     FALSE
          file_name     "$SID_MODEL_PATH\\Common Business Entities
Domain\\Agreement ABE.cat"
          quid          "3D4015D4013F")
      (object Class_Category "Base Types ABE"
          is_unit       TRUE
          is_loaded     FALSE
          file_name     "$SID_MODEL_PATH\\Common Business Entities Domain\\Base
Types ABE.cat"
          quid          "3D801BAB03B0")
      (object Class_Category "Location ABE"
          is_unit       TRUE
          is_loaded     FALSE
          file_name     "$SID_MODEL_PATH\\Common Business Entities
Domain\\Location ABE.cat"
```

Figure U. 6 - Virtual Paths in an .MDL File

Using virtual paths allows the SID models to be stored in a folder of choice, as long as the domain folders are properly set up as sub-folders within the chosen folder. Rational Rose substitutes the folder name assigned to SID_MODEL_PATH when loading control units.

1.2.5. Introducing SID ABEs Into An Existing Model

To introduce SID ABEs into an existing model, open a blank diagram, such as Main. Click on File, click on Import, go to the ABEs domain folder, and select the ABE to import as shown in the figure below (note that the ABEs are all .cat files, and the default is for .mdl files, so initially you won't see any ABEs – simply select Category Files in the Files of type combo box, as shown in the figure below).

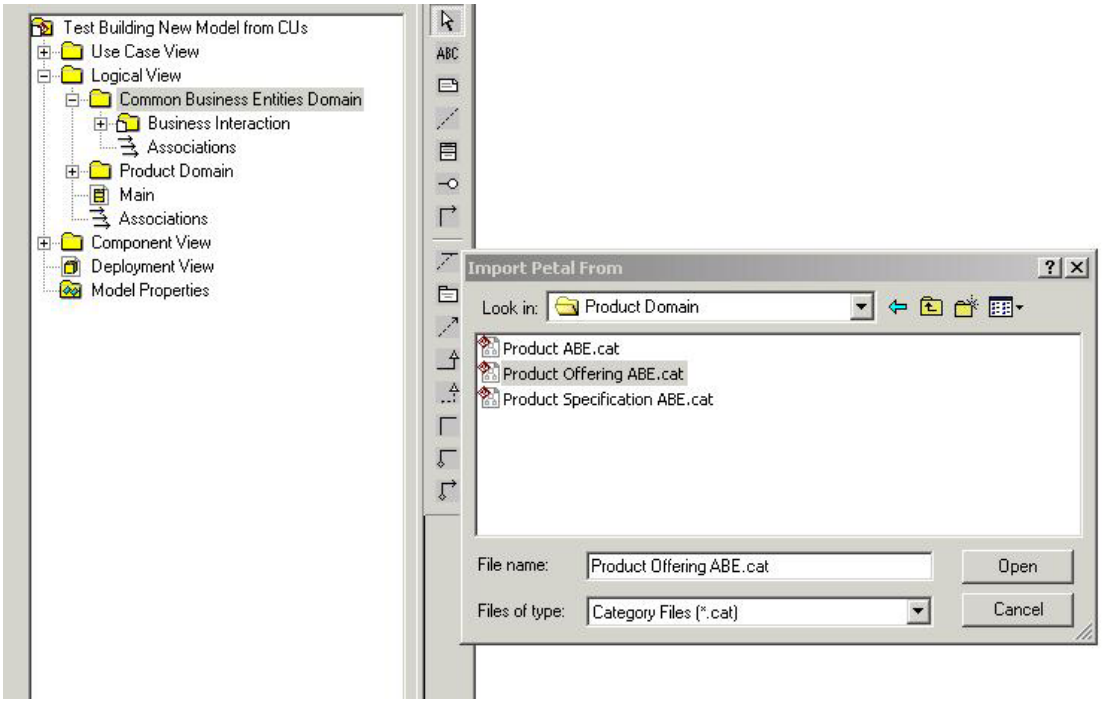

Figure U. 7 - Importing an ABE

The ABE is added to the model and the diagram as shown in the figure below.

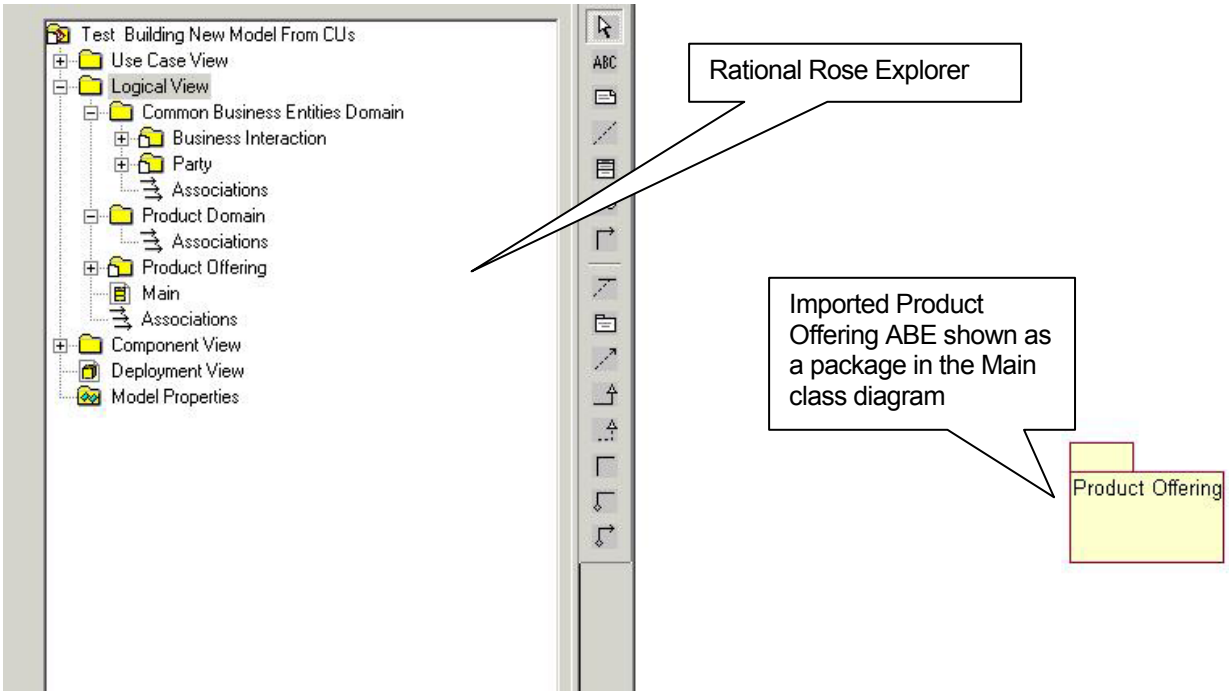

Figure U. 8 - Imported ABE

Next, move the imported ABE to the appropriate domain. In the example, the Product Offering ABE is being moved to the Product domain. The imported ABE package (folder) will appear in this diagram as shown in Figure U. 8 - Imported ABE (the imported package can be deleted from the diagram later.). Alternatively, create a temporary diagram in the domain package to which the imported ABEs belong and use it during the import process. This eliminates the need to move the imported ABEs to the appropriate domain. But, don't forget to delete the temporary diagram when finished.

Both inter- and intra-ABE associations should be kept in the package of the ABE that "owns" the association. The owning ABE is the ABE that is used as the subject in the name of the association. Rose will add associations to the ABE package that is being worked on. This means that if you switch your focus to another diagram in another ABE, the association will be added to that ABE. You will need to find that association and move it to the correct ABE. This can be easily done by selecting the association in the other ABE, dragging it over the desired ABE, and releasing the mouse.

Finally, save the model and control units.

1.3. Patterns for Extending Existing Business Entities

These guidelines should be used when extending an existing ABE. By following them, the existing structure and content of an ABE will not be compromised. Additionally, if the SID team makes changes to an ABE, adherence to these guidelines should minimize the impact to the extensions. The guidelines include:

- creating packages to hold SID extensions
- adding attributes
- adding new entities
- adding associations

Note that the above guidelines apply for extending business entities – a longer and stricter set of rules is being developed for extending system entities and will appear as a future Addendum to GB926.

1.3.1. Creating Packages to Hold Extensions

Rational Rose packages should be used to hold extensions to the SID model. There are a number of reasons for this. One reason is so that future versions of the SID can be imported into a model without impacting extensions. Another reason is that it is easy to show the extensions made to the SID model. Packages added should also be defined as control units so that they can be easily moved from model to model. The figure below shows an example of a packaged added to hold extensions to the SID model Customer ABE.

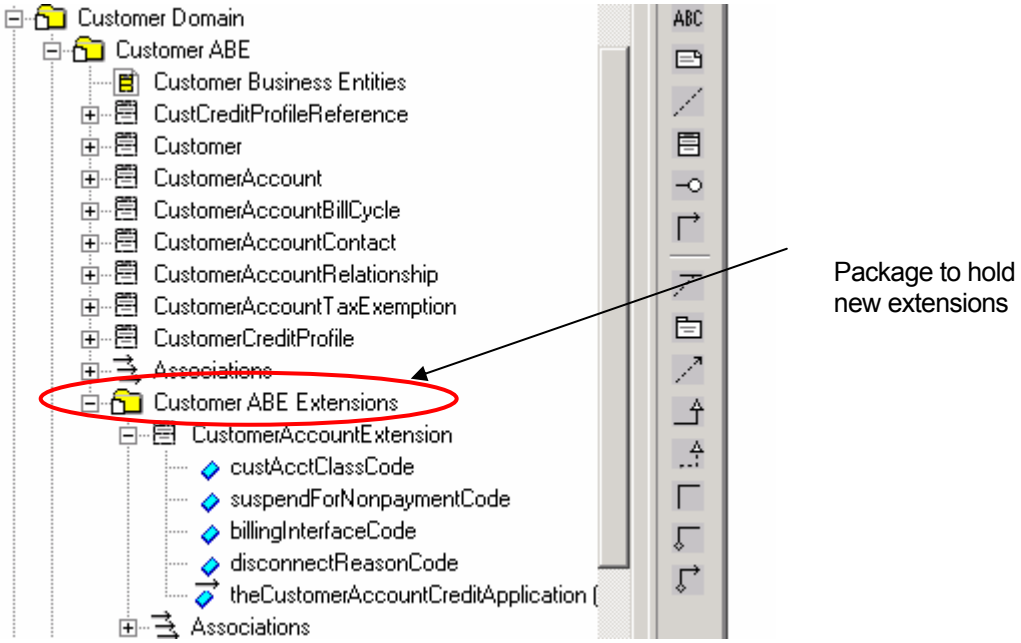

Figure U. 9 – Package Structure For SID Extensions

The package contains all new entities, attributes, and associations that are added to extend the SID model.

1.3.2. Adding attributes

Attributes should not be added directly to SID model business entities or business entity subclasses. Rather, create a subclass of the SID model business entity to which the attributes will be added. The subclass will inherit all of the attributes and associations from the SID business entity thus maintaining the integrity of the SID model. The new subclass holds the attributes as shown in the figure below.

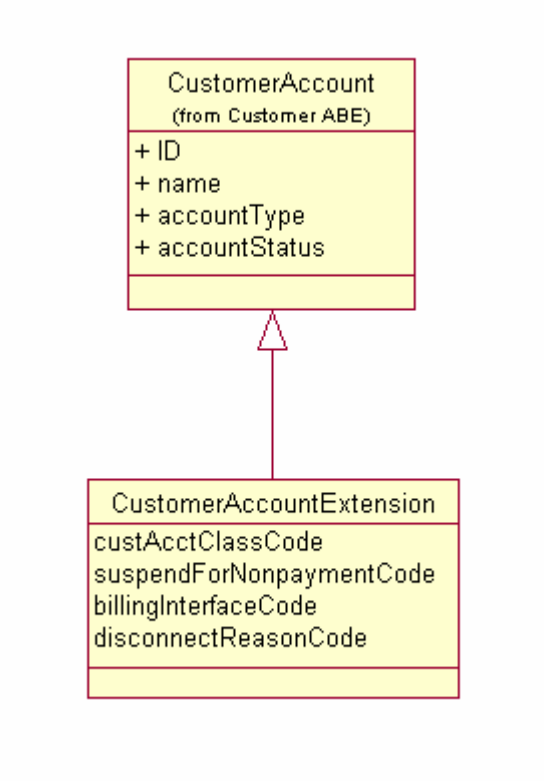

Figure U. 10 – Adding Attributes to an Existing SID Model Business Entity

The name of the business entity that holds attribute extensions is the name recommended by the SID Model team. An alternate name is CustomerAccountSpecialization, since a subclass is used to hold the attributes. The actual name is at the discretion of the individual extending the SID model. At a minimum, a consistent naming convention should be used.

1.3.3. Adding entities

As in adding attributes, new entities should not be added directly to an existing SID ABE package, but rather added to the package that holds the SID model ABE extensions. Two techniques can be used to add entities to a SID model ABE.

The first technique should be used when attributes as well as a new class need to be added to the SID to extend an existing SID class. In the example below, the new entity

(CustomerAccountCreditApplication) needs to be related to an extension of the existing SID CustomerAccout entity (CustomerAccountExtension). This is done by first subclassing the existing SID entity (creating CustomerAccountExtension) and then defining an association (CreditApplicationSubmittedBy) between the subclass of the existing SID entity and the new entity. The figure below shows an example of using this technique.

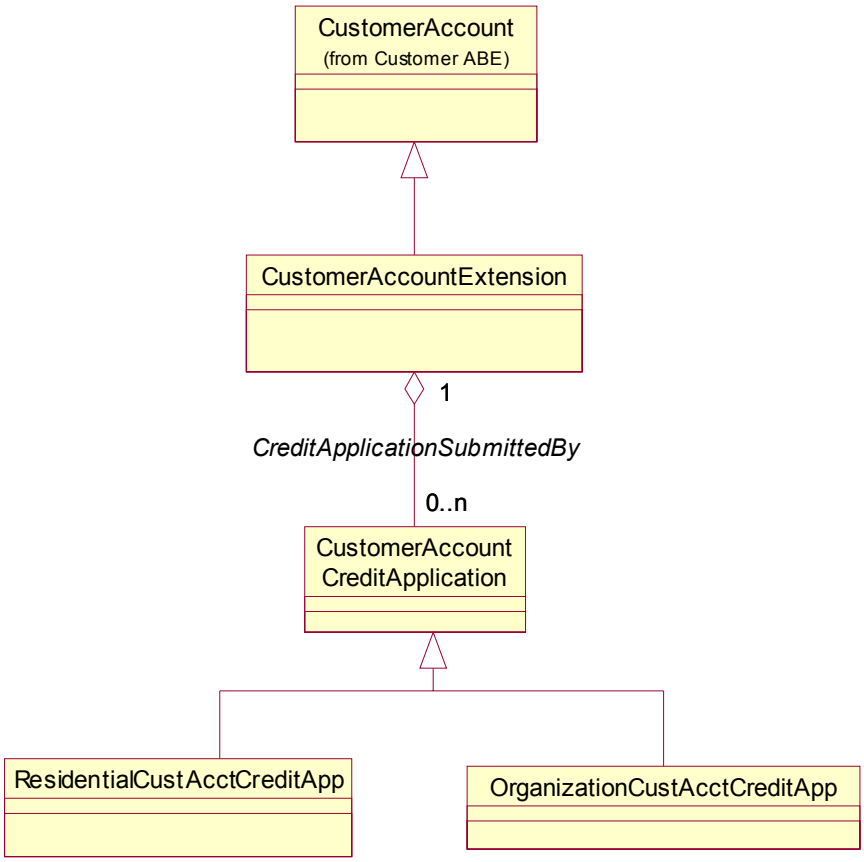

Figure U. 11 - Adding an Entity Related to an Existing Entity – Technique 1

The second technique can be used to add a related entity, such as CustomerAccountCreditApplication, when no attributes have been added to the existing related SID model entity, such as CustomerAccount (there is no CustomerAccountExtension class). The figure below shows an example of using this technique.

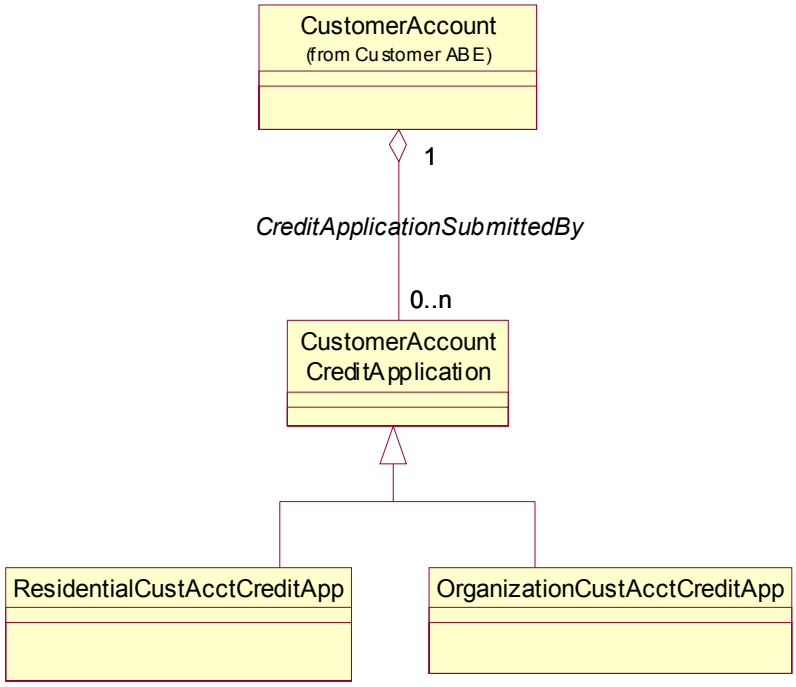

Figure U. 12 – Adding a Related Business Entity

1.3.4. Adding associations

The associations that exist among SID model entities should not be changed or deleted. If a new association is required between existing entities, then they can be added. The association naming guidelines described in this addendum should be used.

1.4. Patterns For Defining New Aggregate Business Entities

These guidelines should be used when developing a new ABE. Occasionally they may be used when adding more detail to an existing ABE. The guidelines include:

- Business entity patterns
- Association, attribute, and package naming conventions
- Guidelines for naming entities
- Guidelines for defining association classes vs. simple associations

1.4.1. Business Entity Patterns

There are established sets of business entity patterns that are used in the SID. These include:

- Entity Specification/Entity
- Entity/Entity Role
- Composite/Component
- Entity Characteristic Spec/Entity Characteristic

1.4.2. Entity Specification/Entity Pattern

The Entity Specification/Entity pattern is used throughout the SID model. Typically, most core business entities (that is, an entity within an ABE that is not dependent upon any other entity within the ABE, such as Customer, Product, Service) have their invariant attributes, methods, relationships and constraints defined by a specification, such as Product Specification and Service Specification. Customer does not have a related specification entity at this time.

This pattern should not be applied to existing ABEs, but should receive high consideration when adding a new ABE or detailing an existing ABE that has not been developed. The figure below shows the use of this pattern in the Root ABE.

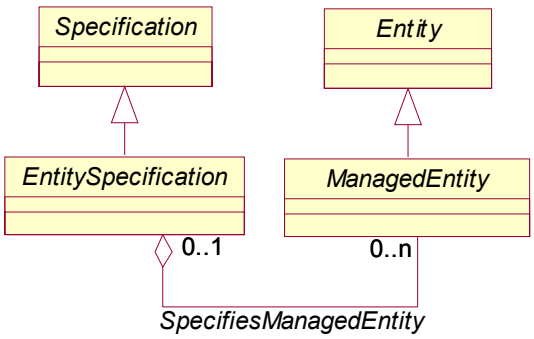

Figure U. 13 – Entity Specification/Entity Pattern

The figure below shows the use of the Entity Specification/Entity pattern in the Service domain.

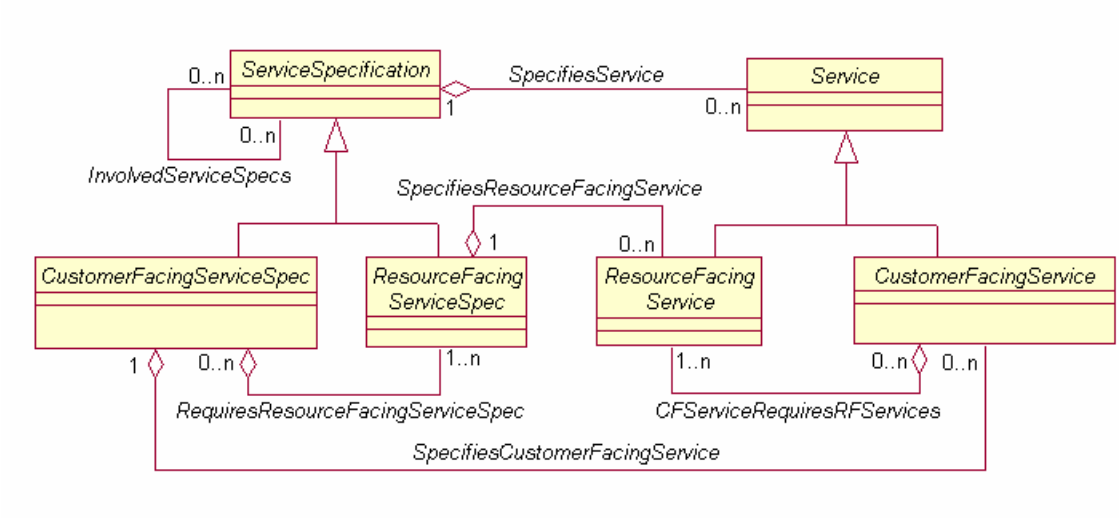

Figure U. 14 – Service Specification/Service Use Of the Entity Spec/Entity Pattern

Typically, there will be an ABE for the specification and an ABE for the entity characterized by the specification within each SID domain or complex ABE. A complex ABE is one that decomposes into a number of other ABEs or into a number of nested ABEs. The reason for this is that each of these business concepts is complex enough to contain a number of related and dependent business entities. The figure below shows the Service Specification ABE to illustrate this point.

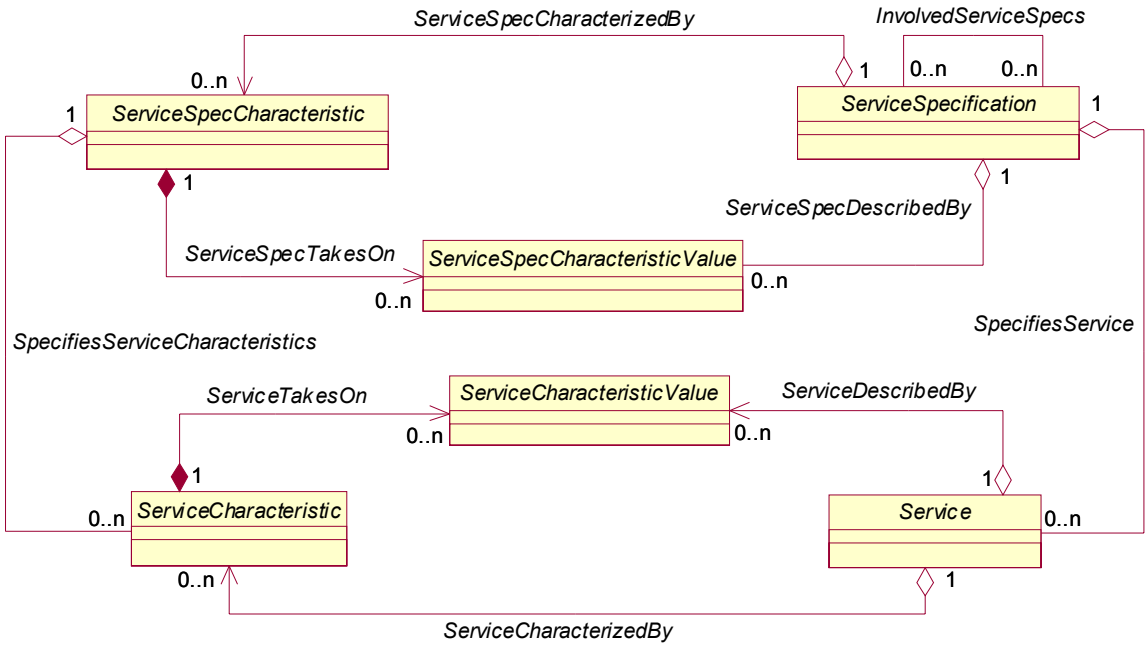

Figure U. 15 – Service Specification ABE

1.4.3. Entity/Entity Role

Many business entities take on a variety of roles during their life of interest to a business. For example, an individual may be a customer and an employee of a service provider. The Entity/EntityRole pattern, shown in the figure below, is used throughout the SID model to represent this concept.

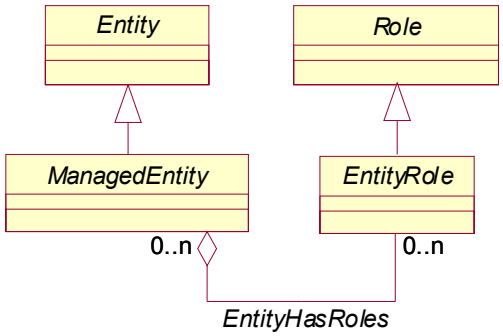

Figure U. 16 – Entity/Entity Role Pattern

The figure below shows the use of this pattern in the Party ABE. Typically, each role that the entity plays is modeled as a subclass of the role entity. Many of these roles are also modeled as ABEs in their own right because of their importance to the business.

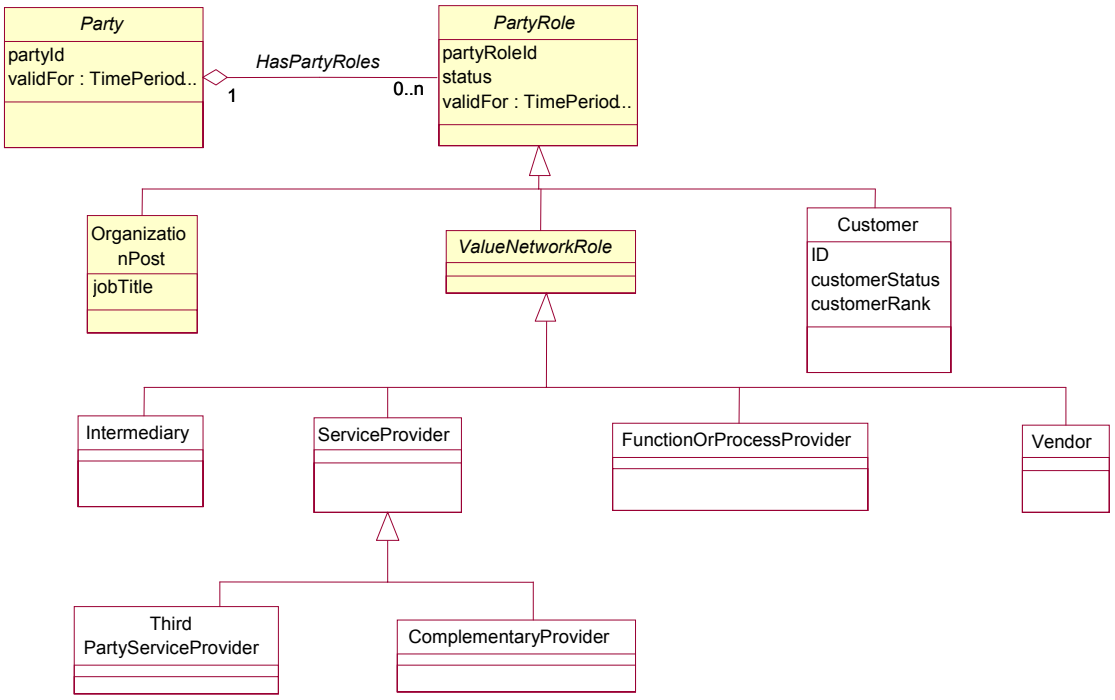

Figure U. 17 – Party/Party Role

1.4.4. Composite/Component

Often, instances of a business entity are composed of other instances of the same business entity. For example, the price for cellular phone service may include a fixed monthly charge and a charge for excess minutes used. To model this, the SID model employs the Composite/Component pattern, shown in the figure below.

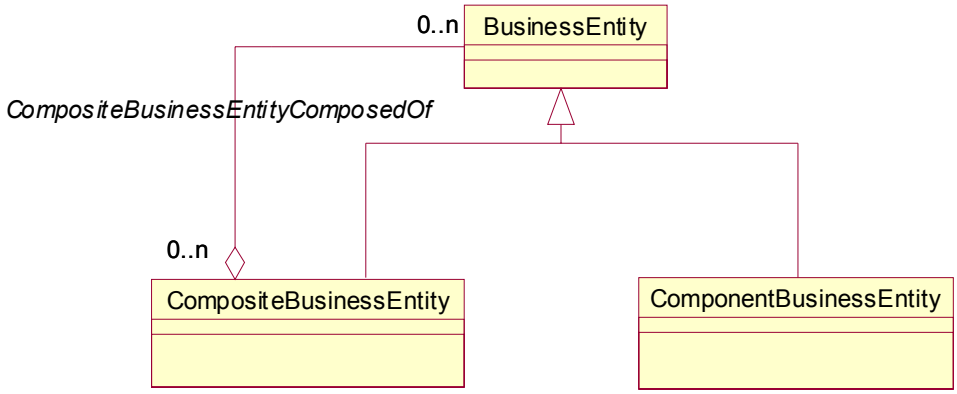

Figure U. 18 – Composite/Component Pattern

The figure below shows the use of this pattern within the Product Offering Price ABE.

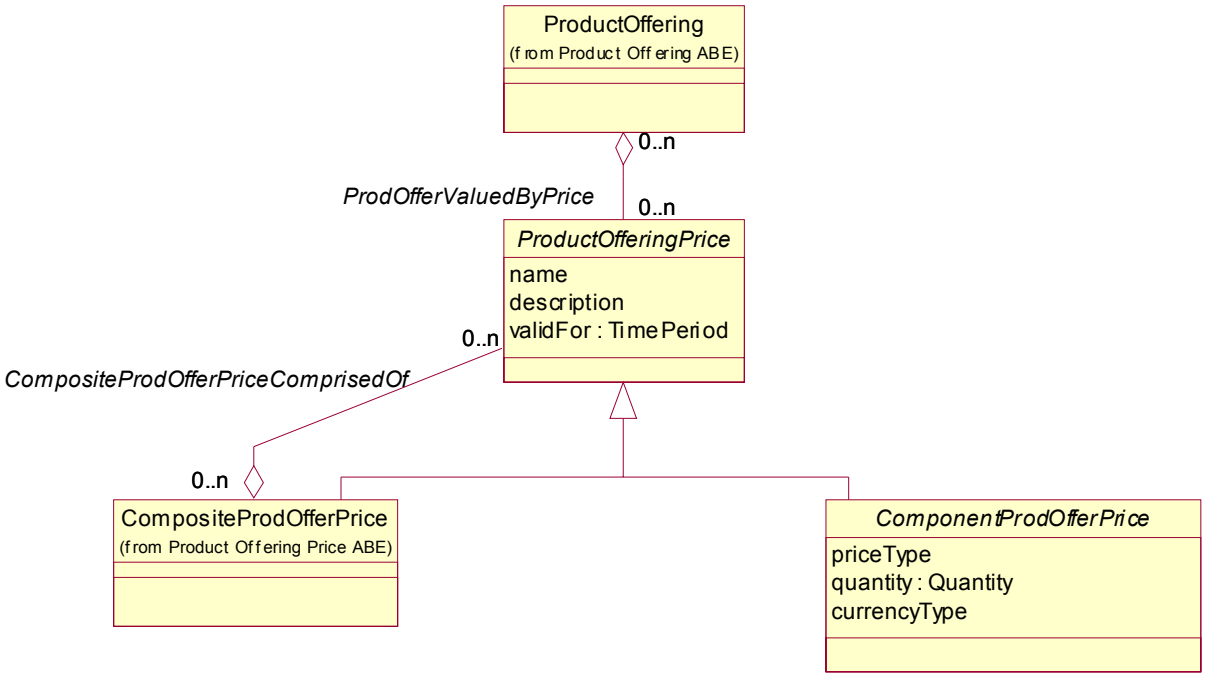

Figure U. 19 – Product Offering Price Composite/Component

1.4.5. Entity Specification Characteristic/Entity Characteristic

When constructing any model, it is almost impossible to discover all the possible attributes that characterize a business entity. Even if all the attributes can be found when the model is constructed, additional attributes will be found as a model is extended. Additionally, certain attributes characterize different types (represented by entity specifications) of business entities. For example, one product specification may be characterized by color and size, while another is characterized by bandwidth. The EntitySpecificationCharacteristic/EntityCharacteristic pattern provides for this type of extensibility and characterization of different entities. The pattern is shown in the figure below.

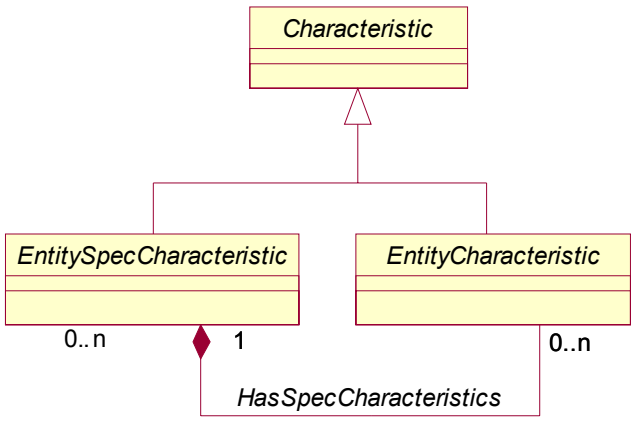

Figure U. 20 – Entity Specification Characteristic/Entity Characteristic Pattern

The next two figures below shows the use of this pattern within the Product Specification ABE.

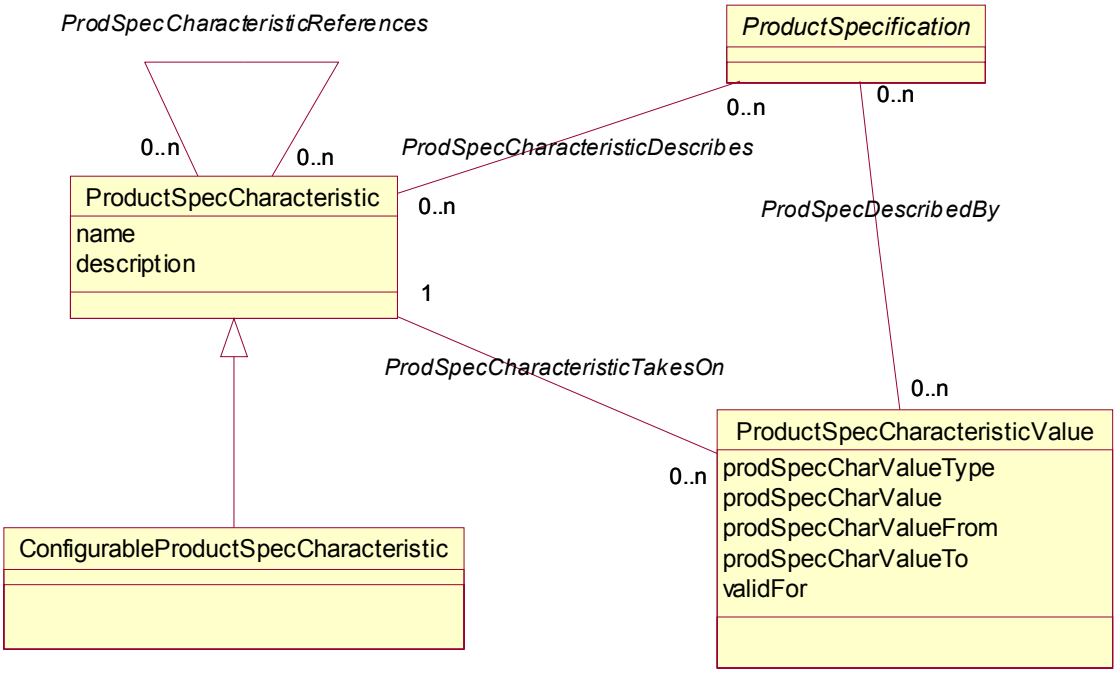

Figure U. 21 – Product Spec Characteristic

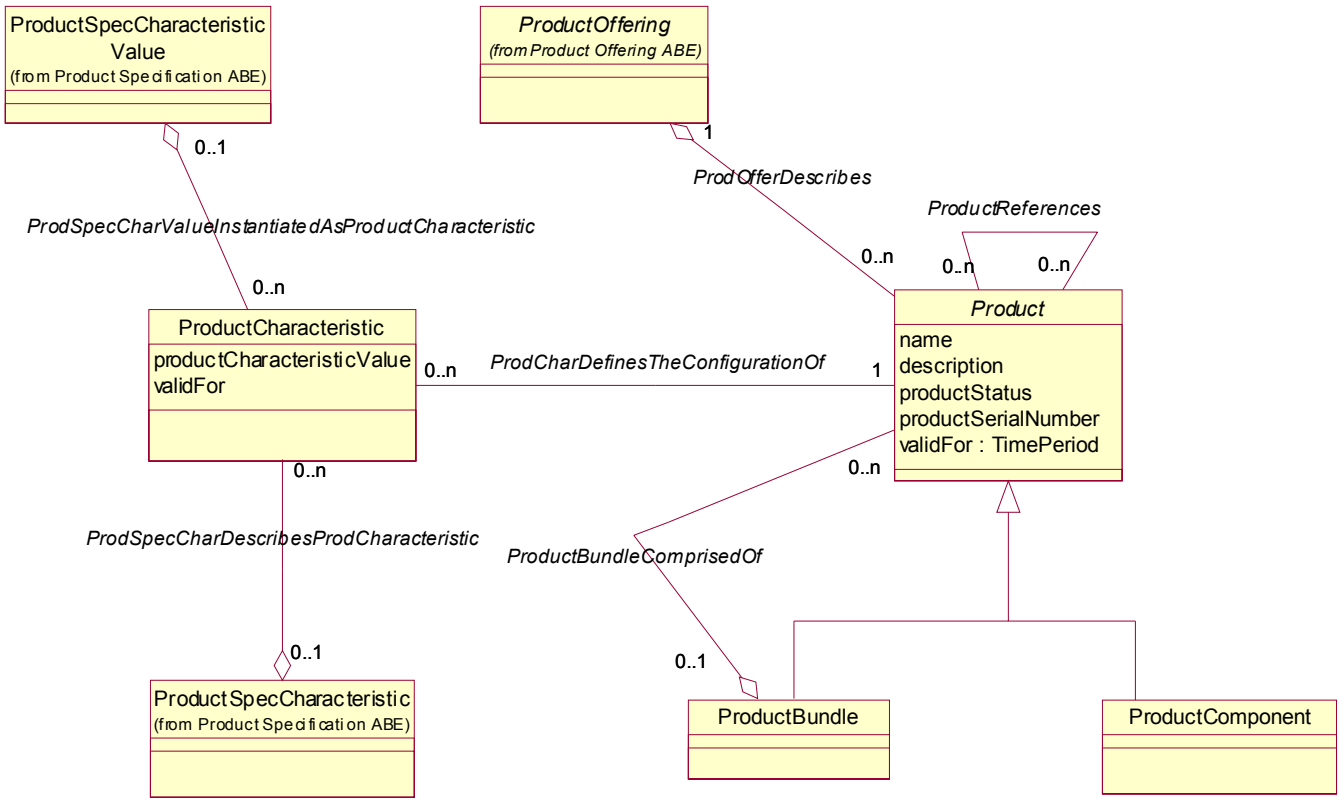

Figure U. 22 – Product Spec Characteristic and Product Characteristic

1.4.6. Entity Classification Group

A Classification Group supports the grouping, organization and analysis of related business management and operational activities according to broad categories, subject areas, and so forth. The use of Classification groups, which are standardized and agreed across the organization allows for the identification and comparison of like activities, as well as the reduction of uncertainty, ambiguity and duplication.

Classification of information into agreed classification groups provides organizations a mechanism to aggregate, interrogate and manage business information on broad categories rather than on an instance or individual basis.

An additional rationale for grouping apart from the above is that it also allows identification of patterns and common behaviors across groups. It allows for "management on the large".

Examples of ClassificationGroups used in the telecommunications industry include: Market Segments, Product Portfolios, Distribution Channels, Consumer Revenue, Technology Platform, and so forth.

The figure below shows the ClassificationGroup pattern.

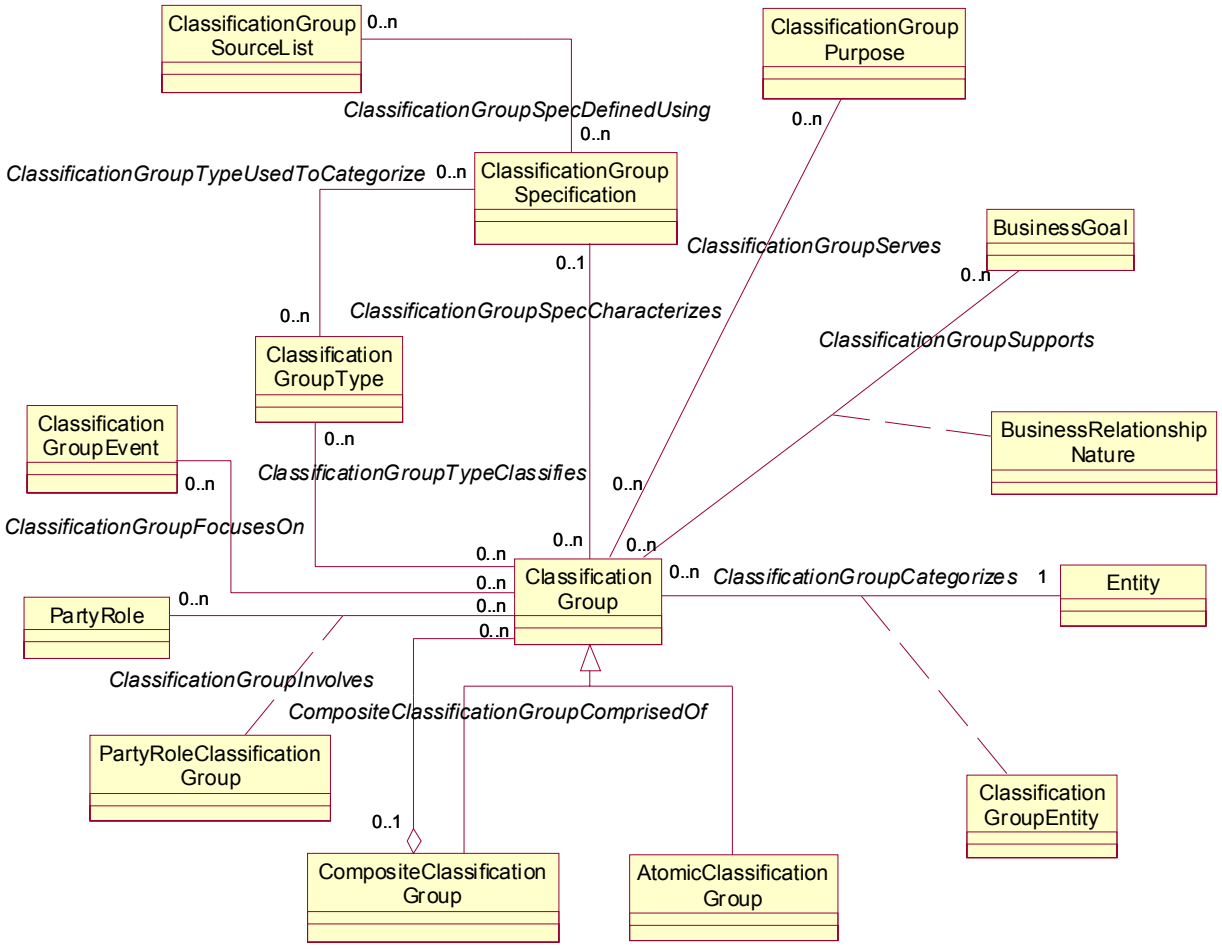

Figure U. 23 – Classification Group Pattern

The figure below shows the application of this pattern for the Market Statistic ABE.

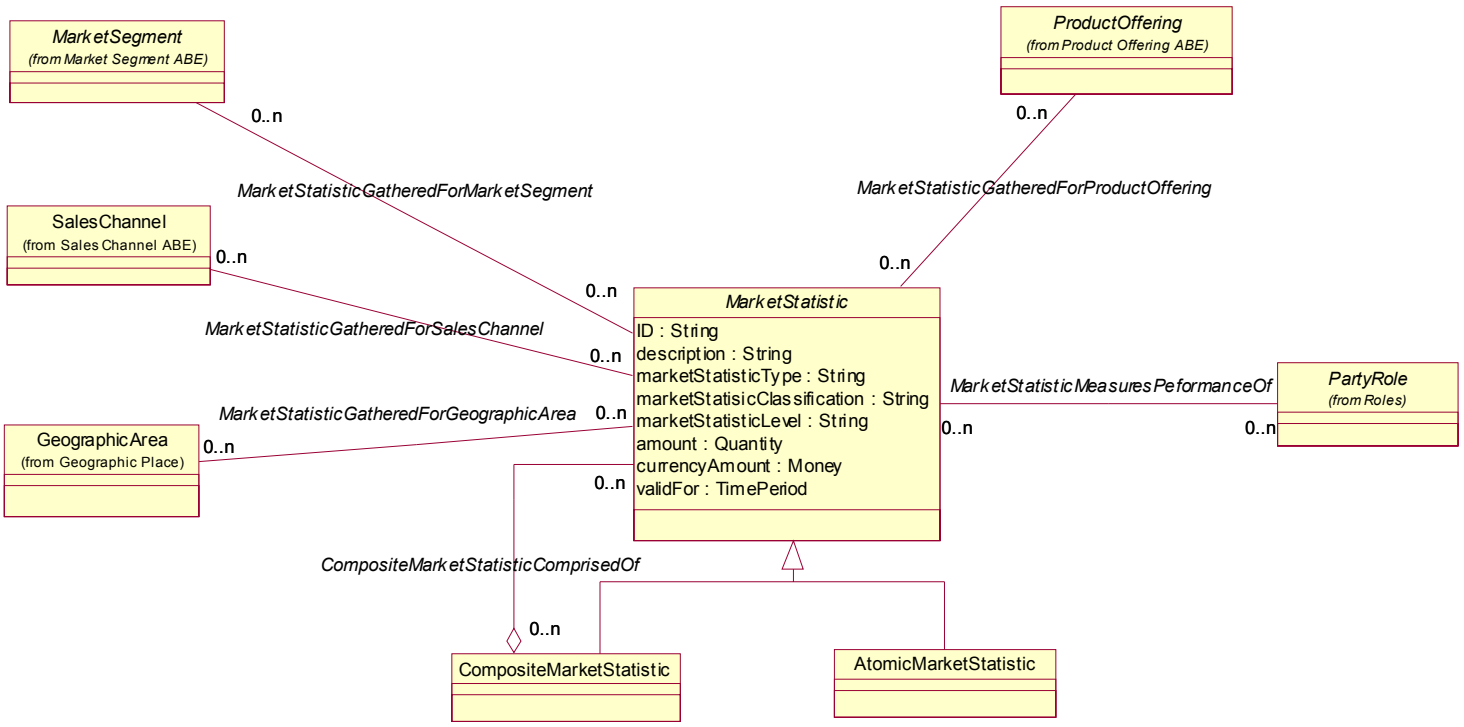

Figure U. 24 – Market Statistic Use of Classification Group Pattern

1.5. Association, Attribute, and Package Naming Conventions

1.5.1. Association Naming Conventions

One method of naming associations is to simply use a verb. To interpret the meaning of the name, the two related business entities are used as subject and object. For example, in the figure below the association between ProductSpecification and BusinessInteractionItem is interpreted as ProductSpecification Involvedin BusinessInteractionItem; the meaning is interpreted by using the upper business entity as the subject. Similarly, the association between Service and BusinessInteractionItem is interpreted as Service InvolvedIn BusinessInteractionItem; the meaning is interpreted from left to right.

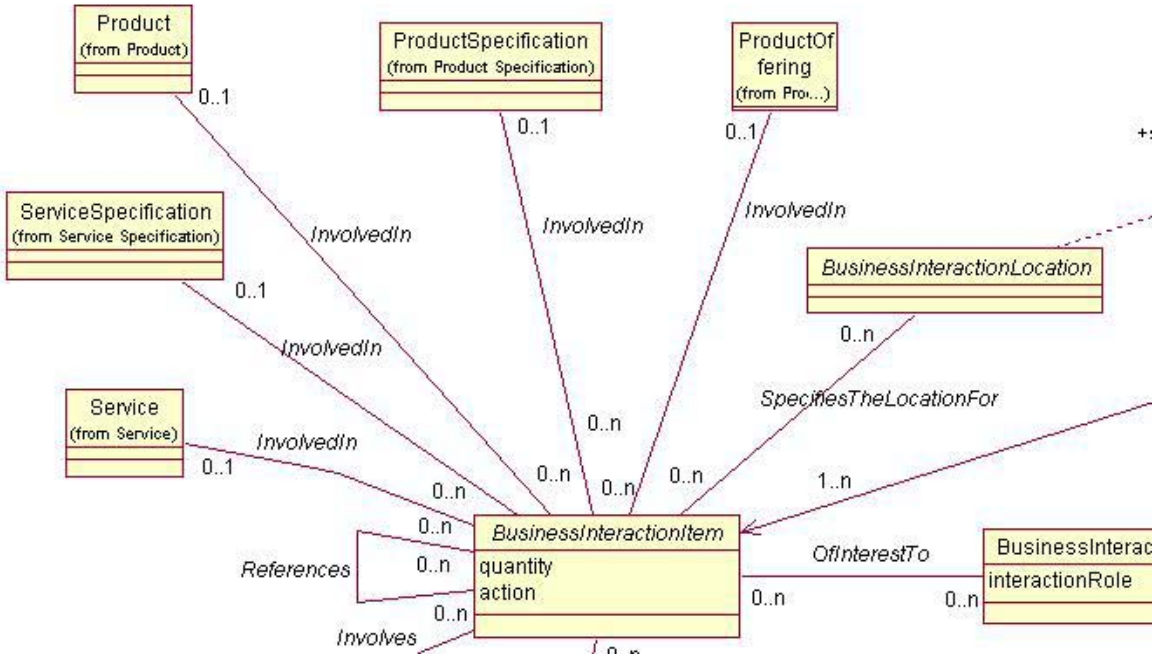

Figure U. 25 – Simple Association Naming

This method of interpretation works well when the business entities retain their relative positions in every diagram in which they appear. However, if in another diagram the positions of business entities are reversed, then the interpretation of the associations is incorrect. For example, if the positions of BusinessInteractionItem and ProductSpecification are reversed, the interpretation of the association is BusinessInteractionItem InvolvedIn ProductSpecification, which is an incorrect interpretation.

This problem with interpreting the meaning of an association can be eliminated by introducing the name of either or both of the related business entities into the name of the association. If one business entity name is used, the name should include the subject as shown in Figure U. 26 - Using One Business Entity in the Name of an Association below. For example, BusinessInteractionTypeCategorizes (BusinessInteraction, the implied object). This naming convention is always used unless the association name results in a duplicate name.

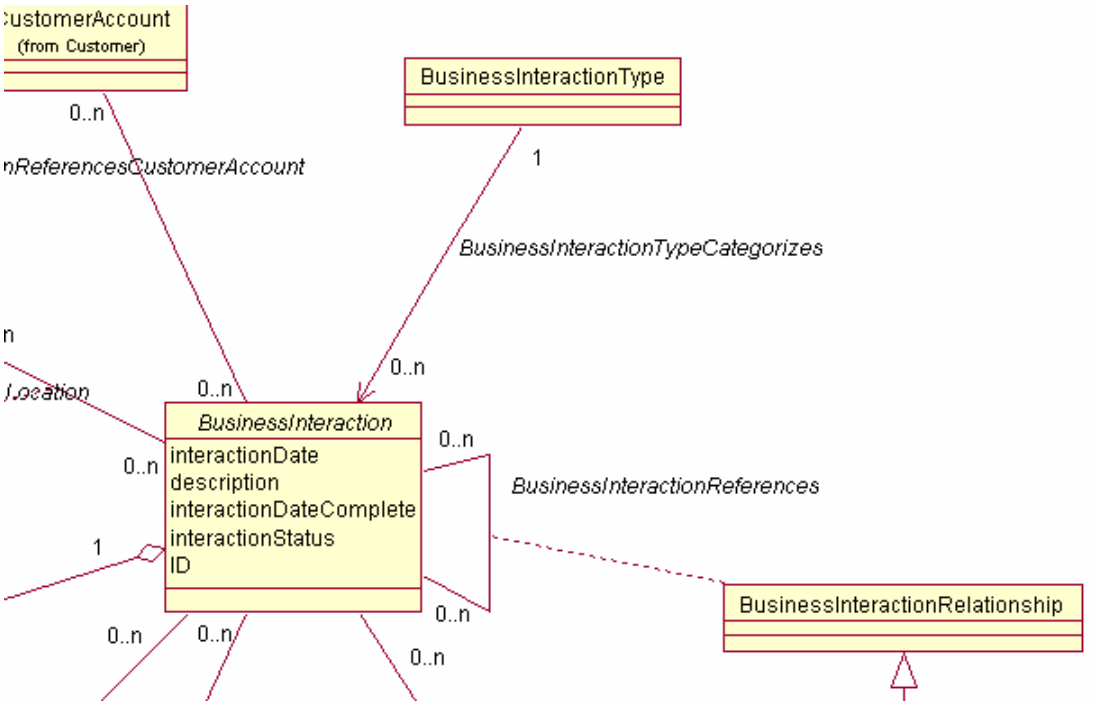

Figure U. 26 - Using One Business Entity in the Name of an Association

To resolve duplicate association names both related business entities are used to form the name of the association as shown in Figure U. 27 below.

For example, BusinessInteractionItemInvolvesProductSpecification. The subject of the association name should be the ABE that owns the association. This means that in the Rational Rose model the association appears in the package containing the business entity that appears as the subject in the association name. In this example, the association appears in the BusinessInteraction package, in which BusinessInteractionItem resides.

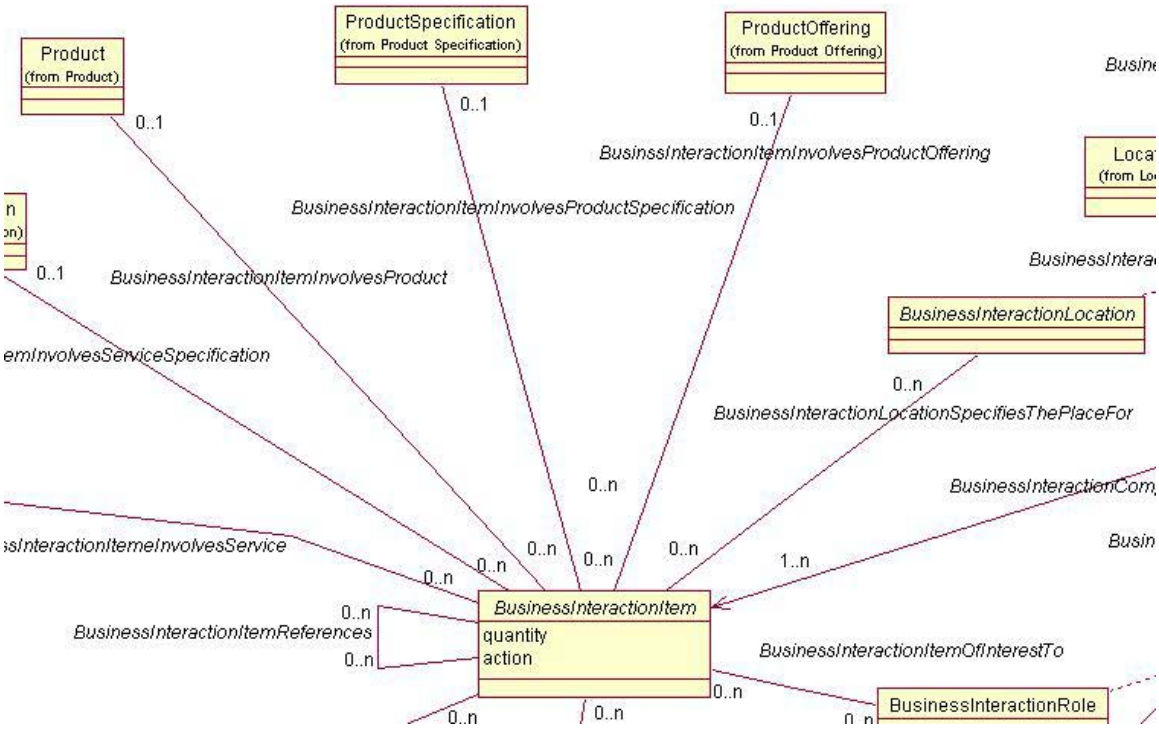

Figure U. 27 - Using Both Related Business Entities in the Name of an Association

It may appear that always using both related business entities in the name of the association would be more prudent. One convention is much simpler to use and understand than two. However, there are a number of reasons for employing two naming conventions. They are:

1. Using both related business entities results in long association names that tend to clutter up diagrams and are difficult to use, and

2. Using the subject/verb/object naming convention will most likely only be needed on a minimum number of occasions.

1.5.2. Attribute Naming Conventions

Naming attributes, like naming associations, can include the name of the attribute's business entity in the name. This is exemplified the figures in this section below.

When an attribute is unique across all business entities or when an attribute is shared among business entities, but maintains the same semantics (characteristics) among the business entities, the name does not include the name of the business entity. An example is shown in the figure below.

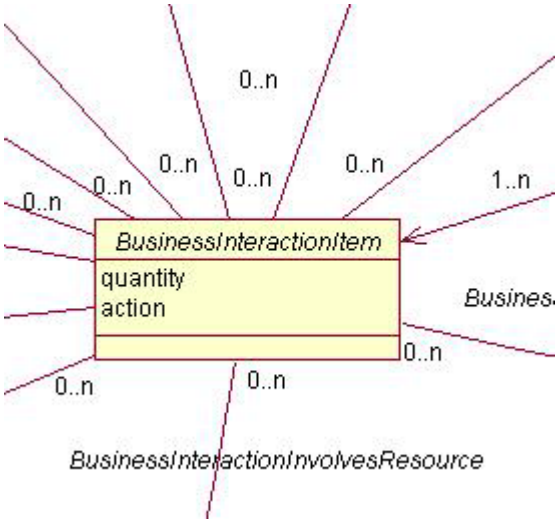

Figure U. 28 - Attribute Names Without Including Business Entity Name

When an attribute, such as interactionStatus, in the figure below does possess different semantics across business entities of which the attribute is a characteristic, then the attribute name should be qualified with the name of the business entity. In the figure below, interactionStatus can take on a number of different values dependent upon the business entity that it characterizes.

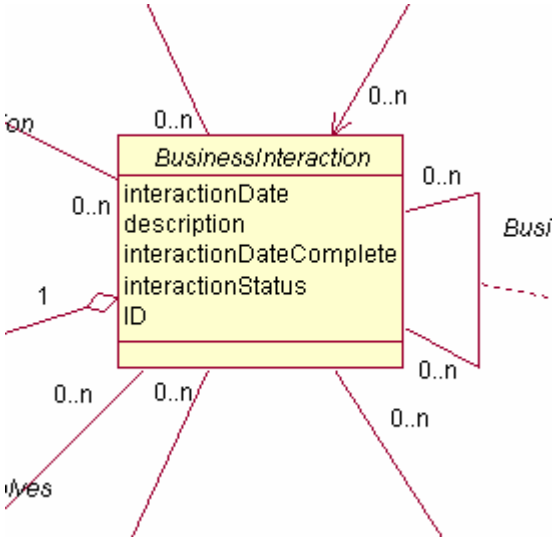

Figure U. 29 - Qualifying Attribute Names With the Business Entity Name

In some cases, a modeler may choose to qualify all attributes that characterize a business entity. This is at the discretion of the modeler.

In the case of Boolean attributes, the recommended convention is to use “is” or “has” as a prefix. For example, the attribute “active” which can take on the values of “yes” or “no”, should be named isActive.

1.5.3. Package Naming Conventions

Package names in the SID Rational Rose model correspond to the names of ABEs in the SID framework.

While no naming conventions are recommended other than using “ABE” as the suffix in the package name of ABEs, the business view names may require adjustment with the use of the model to support other NGOSS views. Package names employing the ABE suffix are shown in the figure below.

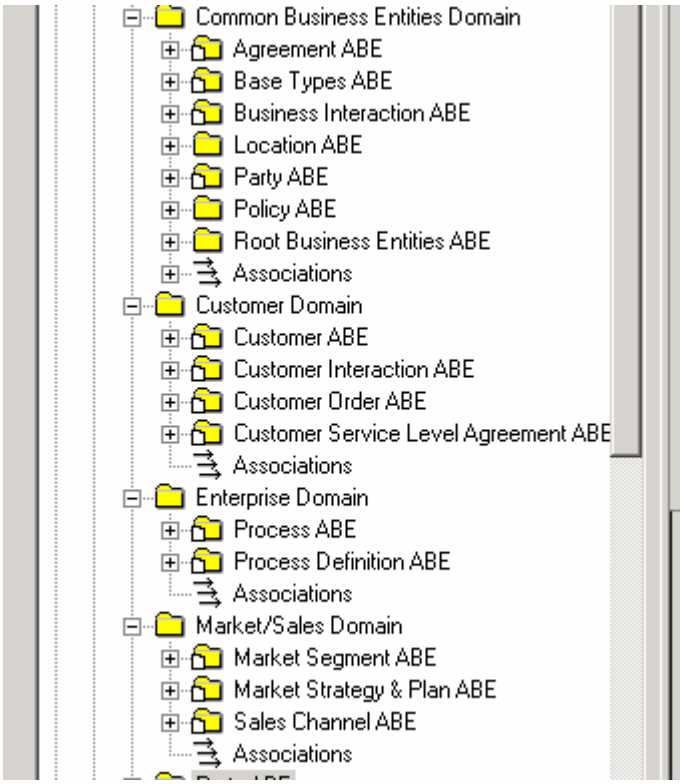

Figure U. 30 - Using "ABE" in the Name of an ABE Package

1.5.4. Guidelines for Naming Entities

Business entities should take on a name that will be familiar to a business individual. The name should not be expressed using technical jargon.

1.5.5. Guidelines for Defining Associations Between ABEs

In general, inter-ABE associations should have a multiplicity of 0…n - 0...n. This is done in order to minimize coupling (dependencies) between ABEs that reside in different domains.

1.6. General Modeling Guidelines

Any work performed on the SID model should follow best practice modeling guidelines, such as:

- Avoiding the use of multiple inheritance
- Minimal use of association classes in the business view; these classes typically appear in the system view
- Describing each artifact is required
- Following the GB922 series of documents with regard to format and content

2. Using the SID as an Integration Framework

One challenge faced when integrating applications is reconciling different terms used for the same concepts in the applications being integrated. A second challenge is reconciling the structure of each application's information model. Being a common vocabulary and an information model framework makes the SID an ideal candidate to be used to meet these integration challenges.

2.1. A SID-Based XML Integration Framework

Defining the SID in terms of XML schemas, or XSDs, is an approach that can be used to create an integration framework. Messages that are interchanged between applications are defined using the SID model components (business entities, attributes, and their associations) along with application specific extensions to the SID.

Since the SID model is a framework, applications are expected to extend the SID model. These extensions complete the definition of business entities that are to be used by an application. Extensions can include objects such as additional business entities, attributes, and/or associations. The SID model and extensions to the SID form the basis of the XML information schema as shown in the figure below.

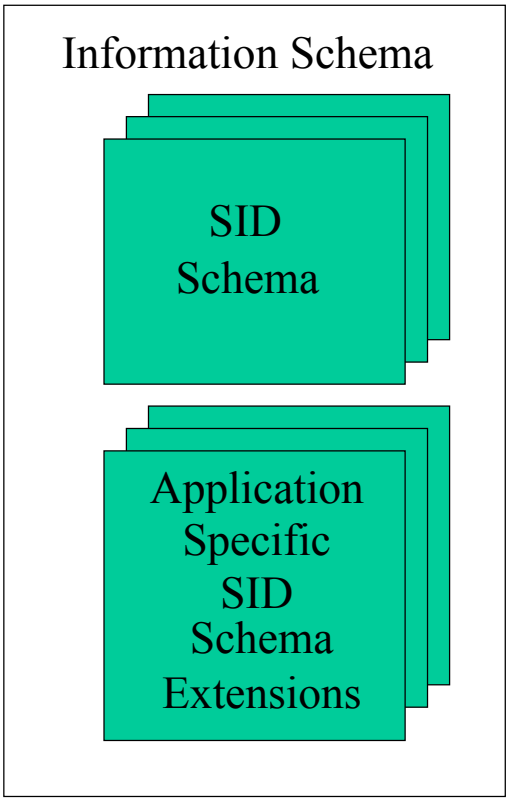

Figure U. 31 – Integration Framework Information Schema

The figure below shows the SID model's Customer ABE as an XML schema.

```
<xs:complexType name="CustomerAccount">
    <xs:sequence>
        <xs:element name="ID" minOccurs="0"/>
        <xs:element name="name" minOccurs="0"/>
        <xs:element name="accountType" minOccurs="0"/>
        <xs:element name="accountStatus" minOccurs="0"/>
        <xs:element name="CustomerAccountTaxExemption" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountBillCycle" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="AppliedCustomerBillingRate" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountChargeSum" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountChargeSumSpec" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerBill" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerBillSpec" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountContact" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountRelationshipHas" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountInteractionRole" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="Customer" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountProductInvovlement" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="CustomerAccountRelationshipFor" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
```

Figure U. 32 – XSD Representation of the SID Customer ABE

The next figure shows an example of an application specific extension to the SID Customer Account business entity within the Customer ABE.

```
<xs:complexType name="CustomerAccount">
    <xs:complexContent>
        <xs:extension base="SIDBusCu:CustomerAccount">
            <xs:sequence>
                <xs:element name="address1" nillable="true" minOccurs="0"/>
                <xs:element name="address2" nillable="true" minOccurs="0"/>
                <xs:element name="address3" nillable="true" minOccurs="0"/>
                <xs:element name="county" nillable="true" minOccurs="0"/>
                <xs:element name="creditScore" nillable="true" minOccurs="0"/>
                <xs:element name="entryDate" type="xs:date" nillable="true" minOccurs="0"/>
                <xs:element name="lastChangedDate" type="xs:date" nillable="true" minOccurs="0"/>
                <xs:element name="resellerReference" nillable="true" minOccurs="0"/>
                <xs:element name="snNumber" nillable="true" minOccurs="0"/>
                <xs:element name="createSessionDate" nillable="true" minOccurs="0"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

Figure U. 33 – Application Specific Extension to Customer Account Business Entity

This type of application specific extension is used if the application contains all the SID attributes and associations. If the application contains a subset of the SID attributes and associations, then a restriction base is used for the extension. Use of this method is described in GB922 Addendum X – SID XML Schema (XSD) Overview.

These two sets of information schema form the basis from which inter-application communication is constructed. The figure below shows how an application-specific schema can be constructed using the SID as a foundation.

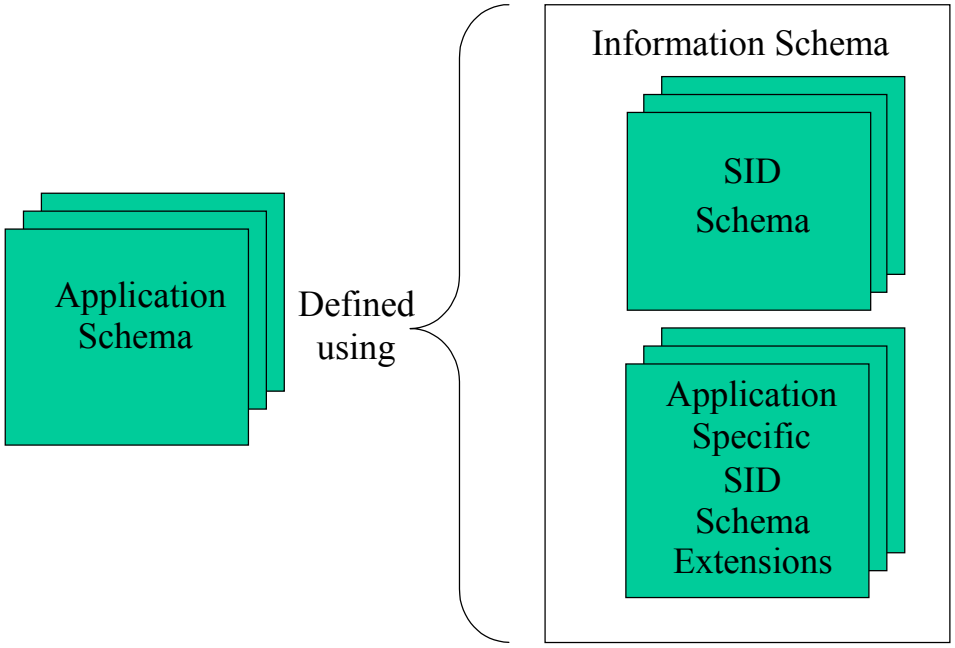

Figure U. 34 – SID-Based Integration Framework

The next figure shows the structure of an XSD, defined in terms of the SID-based integration framework, in graphical form.

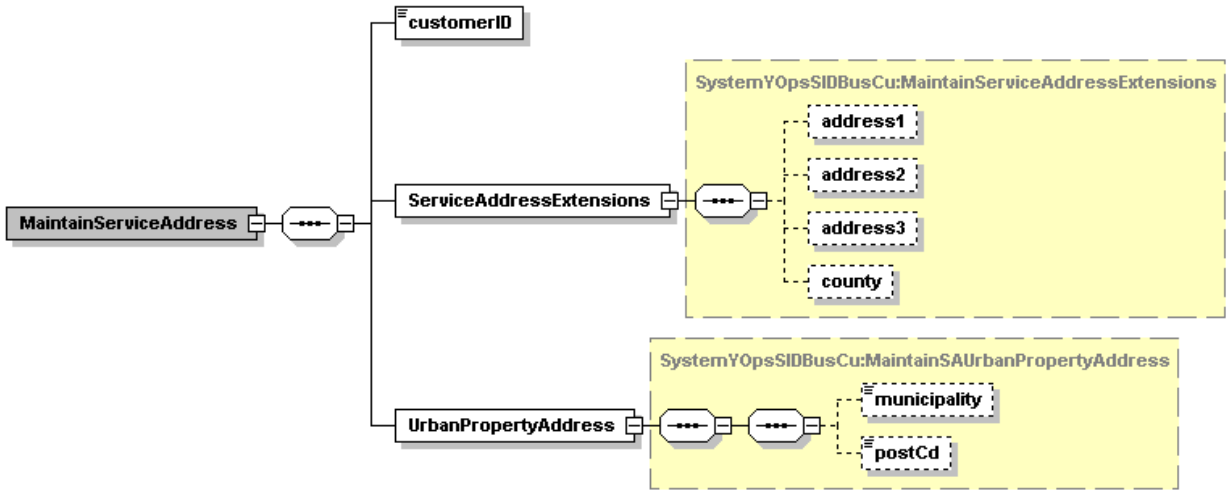

Figure U. 35 – Inter-Application Communication Content Defined in Terms of SID and Application Specific Extensions to the SID

SID domains can be used to organize the schema for each component of the integration framework as shown in the figure below. The figure shows the SID Customer domain XSD, an application operation message payload XSD for the Customer domain, and a application specific extension XSD for the Customer domain.

Figure U. 36 – Integration Framework XSD File Structure

Administrative Appendix

This Appendix provides additional background material about the TeleManagement Forum and this document. In general. sections may be included or omitted as desired, however a Document History must always be included..

About this document

This is a TM Forum Guidebook. The guidebook format is used when:

- The document lays out a 'core' part of TM Forum's approach to automating business processes. Such guidebooks would include the Telecom Operations Map and the Technology Integration Map, but not the detailed specifications that are developed in support of the approach.
- Information about TM Forum policy, or goals or programs is provided, such as the Strategic Plan or Operating Plan.
- Information about the marketplace is provided, as in the report on the size of the OSS market.

Document Life Cycle

This document is being issued for Member Evaluation. The purpose of an Evaluation Version is to encourage input based on experience of members and the public as they begin to use the document. Following the Evaluation Period, documents that are seen to deliver value are candidates for formal approval by the TM Forum. All documents approved by the TM Forum undergo a formal review and approval process.

This document will continue under formal change control. Supporting work will be issued as companions to this document. A document of this type is a "living document," capturing and communicating current knowledge and practices. Further inputs will be made because of detailed work ongoing in the TM Forum and the industry.

Document History

Version History

<This section records the changes between this and the previous document version as it is edited by the team concerned. Note: this is an incremental number which does not have to match the release number>

Version Number	Date Modified	Modified by:	Description of changes
0.1	Sept 2003		First Draft
0.2	Sept 2003		Second Draft based on comments from sub-team (John Strassner and Josh Salomon)
0.3	Oct 2003		Additional comments from sub-team
0.4	Dec 2003		Final review from sub-team, based on other Phase IV deliverable
0.5	Dec 2003		Updates based on review comments by Cliff Faurer and Wayne Sigley
.6	Feb 2004	John Reilly	Updated based on SMT review
6.0	July 2005	John Reilly	Updated based on member contributions.
6.1	November 2005	Tina O'Sullivan	Converted to new template and corrected various administrative items.
6.2	November 2005	Tina O'Sullivan	Figure label
6.3	November 2005	Tina O'Sullivan	Moved fig 5.
6.4		Tina O'Sullivan	Updated notice statement & document status

Release History

<This section records the changes between this and the previous Official document release>

Release Number	Date Modified	Modified by:	Description of changes
Release 6.0	31-Oct-2005	J. Reilly	

Acknowledgments

This document was prepared by the members of the TeleManagement Forum SID team:

The Shared Information/Data Model is a genuinely collaborative effort. The TM Forum would like to thank the following people for contributing their time and expertise to the production of this document. It is just not possible to recognize all the organizations and individuals that have contributed or influenced the introduction. We apologize to any person or organization we inadvertently missed in these acknowledgments.

Key individuals that reviewed, provided input, managed, and determined how to utilize inputs coming from all over the world, and really made this document happen were:

Name	Affiliation
Ian Best	TeleManagement Forum
John Reilly	MetaSolv Software
Wayne Sigley	Telstra
John Strassner	Motorola
Josh Salomon	Amdocs
Greg Fidler	Practical Enterprise Architecture P/L

About TeleManagement Forum

TeleManagement Forum is an international consortium of communications service providers and their suppliers. Its mission is to help service providers and network operators automate their business processes in a cost- and time-effective way. Specifically, the work of the TM Forum includes:

- Establishing operational guidance on the shape of business processes.
- Agreeing on information that needs to flow from one process activity to another.
- Identifying a realistic systems environment to support the interconnection of operational support systems.
- Enabling the development of a market and real products for integrating and automating telecom operations processes.

The members of TM Forum include service providers, network operators and suppliers of equipment and software to the communications industry. With that combination of buyers and suppliers of operational support systems, TM Forum is able to achieve results in a pragmatic way that leads to product offerings (from member companies) as well as paper specifications.

