patterns1 nugget

Created by Unknown User (joshs@amdocs.com)

This Nugget is supposed to explain the usage of two of the basic patterns used in the SID

- Specification pattern (EntitySpec/Entity)
- Characteristics pattern (EntityCharacteristicsSpec/EntityChararteristics/EntityCharacteristicsValue)

These patterns are used to enable inherent extensibility to the information model framework. The following text explains the usage of these pattern to extend an instantiated model (resource model) without changing the base model. This is a common functionality in catalog applications where you want to represent new entities without the need to change the model and rebuild the application. The example I use is for modeling equipment (type of resource) but it is correct for every instantiation of entities which use this pattern (Product, Service, Usage and many others throughout the SID)

In order to define entities in the Information model we need to define its structure - in other words what are its attributes (name, type cardinality, possible values and other attribute definition meta data). The structure is composed of:

- The list of attributes. This list is common, invariant and fixed for all instances of a specific type - the example from programming world is the definition of a class in C++ or Java.
- The value of the attributes - some of the values are fixed and shared for all entities of the same type (such as manufacture and model type assuming you create different entity for each equipment type) this is parallel in C++ or Java languages to “static const” attributes
- Attributes which take different value for each instance (such as serial number of equipment)

All these cases can be supported by defining classes for each type of equipment and using UML to define the invariant attributes as static and const but this way the model should be changed each time a new equipment is introduced (or product or service or any other type). On most cases where this pattern appears in the model, SID takes a different approach and makes the structure of the entity dynamic so the model is not changed when you introduce new types of entities; just the instantiation of the model is changed. This is not to say that this approach can't be used, but it is recommended doing so only if the list and structure of entities is pre-known and the chances of adding new entity or changing existing entities is very slim. This is a good approach to define a book - we know what book attributes are (name, author, ISBN # etc.) and the list of attributes changes very infrequently.

For cases where the model requires the flexibility to add new entity types or you to change entity definitions frequently SID uses the EntitySpec/Entity pattern in conjucttion with the EntitySpecCharacteristic/EntityCharacteristic pattern. When using these patterns ALL the attributes are EntitySpecCharacteristics (both variant and invariant), invariant characteristics take EntitySpecCharacteristicValues and variant characteristics take EntityCharacteristicValues.

In GB922 addendum 1R (root entities), SID phase 8 and later, there is an explanation how to create entity specifications by deriving from EntitySpecification (existing class), and how to connect characteristics to them without the need to replicate the entire pattern.

Important note: Do not derive your entity from RootEntity since the SID team tries to make changes in the way we use RootEntity in the SID, instead you should create a new association between your entity and CharacteristicValue (see diagram Ch.02 in addendum Addendum 1R (Common Business Entity Definitions – Root Business Entities)).