Exploring the Vector behavior of relationships in common information model (CIM) based service-oriented architecture (SOA) environments

Gandhi Sivakumar  
Senior IT Architect  
IBM

In the process of creating SOA based services from Common information model, we know that there is a need to extend CIM due to its abstract nature. An important aspect to understand is that the driving context of the constellation of CIM objects pertaining to an SOA based service is its "Business intention". By "business intention" I mean the expected functionality of the Service component which in turn drives the anchor object(s) specific to the Service operation request/response. Thus the business intention of a service determines the traversal direction to retrieve the elements within the constellation and this in turn largely impacts the relationships of the objects. Eventually relationships tend to behave like a vector which is explored in this article.

Introduction

Industry based CIM model like TMF (TeleManagement Forum) published SID (Shared Information Data Model) manifests as a big blob of entities including objects, attributes and relationships while CIM model like the IBM published IFW (IBM Financial Services Industry Model) additionally includes object interfaces and enables the user to generate service specifications.

Business intention, CIM anchor object(s) and relationship directionality

Any service component has its intended business functionality termed as "Business Intention". Business intention determines the CIM anchor object(s) (specific to the service operation) which drifts the object relationship to behave like a vector.

The CIM itself is a big blob and as stated earlier it is the "Business intention" (BI) of each Service which determines the anchor object(s). Anchor object is the CIM object where every service operation request or response hits the bedrock of the CIM. Once the anchor object(s) is defined, all the other CIM elements can be retrieved and this has significant impact in the relationship behavior of the objects within a service. For example when there is a relationship defined by CIM between Object A and Object B, BI may determine the anchor object in such a manner that the traversal may need to happen from B to A (In this case in the Service interface, B will contain A) or
otherwise. Thus relationships tend to acquire a "vector" behavior equating the relationship name to the magnitude in addition to possessing directionality.

In this article, I intend to explore such a characteristic of the relationships of CIM objects in SOA based implementations.

**Business process and CIM layers**

Figure 1 below shows the business process and the CIM layers.

**Figure 1. Business Process and CIM Layers**

- The upper layer is the Business process layer encompassing the process elements shown as bubbles which eventually turn into Services. *(Note: The process layer in reality may manifest at various levels of abstraction ex: e-TOM (enhanced Telecom operations map) in Telecom industry and APM (Analysis process model) in banking industry. Depending upon the re-use, an SOA Architect may decide to realize the functionality as one service or multiple services.)*
- The lower layer is the CIM layer encompassing the CIM objects. Also named are the CIM objects (ProductOrder, Resource, TelephoneNumber) which we will be discussing in the example for understanding the "vector behavior".

**Example: Service for OrderManagement**

Consider an example of creating a service for managing an order termed as "OrderManagement". The next step is to analyze the Business Intention of this service.

**BusinessIntention**

Business intention is nothing but the Business functionality of the service. For simplicity, let us assume that the SOA/Integration architect in this case decides to fulfill this functionality through a single service component only; while in reality the functionality may be fulfilled via multiple service components. An important aspect to understand is that what ever be the number of services
through which an abstract service fulfills the given functionality, its intention remains the same. In our example say though order management process had to be internally realized through services such as "AddressManagement", "ProductValidation", etc, its intention remains to manage the order only. Once this aspect is understood clearly, the SOA/Integration Architect needs to identify the Anchor object(s).

**Anchor object characteristics**

Anchor object is the CIM object(s) where the service operation links to the CIM as stated earlier. Selection of anchor object(s) has to be completely aligned with the business intention coupled with the Service operation request and response.

**Single anchor object**

For example let us say that the BI of OrderManagement is to perform the fulfillment of the order captured from the front end CRM (Customer Relationship & Management) application. The service request say is expected to receive the complete order structure and respond back with another OrderId created by it.

In this case the anchor object for the request is the ProductOrder (using the TMF SID as CIM) i.e the service operation SubmitOrderRequest will be coupled or linked to ProductOrder (the CIM) and then all the other elements would be retrieved using this. On the other hand the response termed as SubmitOrderResponse say will again be linked to the ProductOrder (CIM) as shown in figure 2 below:

**Figure 2. ProductOrder as anchor**
In this case both the request and response have only one anchor object and they are the same for the request and response operations.

**Multiple anchor objects**

In reality there may be more than one anchor object immaterial of the operation request or response. Let us consider a different example a service “AddressManagement”. The BI of AddressManagement is to manage the address details. If the AddressManagement needs to be designed in way that it can receive either the AddressReferenceIdentifier or the MobileTelephone number of the Customer and return back an address structure, then the request say "RetrieveAddressDetailsRequest" may link to Address (CIM) as well as TelephoneNumber (CIM) i.e more than one anchor object while the response "RetrieveAddressDetailsResponse" may link to Address (CIM) only.

**The vector behavior**

Once the anchor object(s) are finalized, traversal to the other objects become sensitive to direction and the relationships tend to behave like a vector which is discussed subsequently.

**Traversal from ProductOrder to TelephoneNumber**

Consider our first example OrderManagement service. In this case let us assume that the order also includes the Telephone number. In order to retrieve "TelephoneNumber" using ProductOrder as the anchor object, the traversal needs to be done from ProductOrder.

**Figure 3. ProductOrder to TelephoneNumber**

![Diagram](image)

Figure 3 shows the traversal from ProductOrder to TelephoneNumber which is through the path a1, b1, c1, d1. a1, b1, c1, d1 represent the relationship names. In the service interface generated, ProductOrder will contain TelephoneNumber which is traversed through a1, b1, c1, d1 i.e traversal becomes sensitive to direction with the direction being similar to the direction of vector and relationship names being similar to the magnitude of the vector.
Traversing from TelephoneNumber to ProductOrder

Consider another service say TelephoneNumberManagement. This service say has a request operation ReserveTelephoneNumber with inputs being TelephoneNumber and ProductIdentifier and the response being success or failure. In this case the anchor object for request is TelephoneNumber because the BI is to manage telephone number. This means that in order to retrieve ProductIdentifier, the traversal needs to be done in the opposite direction as shown in figure 4 below. The path is different and needs to have a different name; in this case it is d2, c2, b2, a2. In the service interface generated, TelephoneNumber will contain ProductOrder with different relationship names.

Figure 4. Reserve Telephone Number Request

UML, XSD and tools

The CIM is UML based most of the times; which means traversal is possible in either direction among the objects. However when processing for SOA, the BI needs to be identified, thus directions get specified as we discussed. For every bi-directional relationship (excluding parent child relationship) ideally another relationship with a different name needs to be defined i.e. different name based upon the direction. Tools should be intelligent enough to handle the cyclic dependency issues. IBM RSA (Rational software architect) tool is intelligent enough to handle the “Vector behavior”. In addition Progress DXSI data extend, a third party tool is able to handle such scenarios and has proven to integrate with IBM blue stack tools like WID (WebSphere Integration Developer), WPS (WebSphere Process Server). Depending upon the project needs, the SOA/Integration Architect needs to select the appropriate tool.

Conclusion

In CIM based SOA environments, relationships among the objects behave like a vector becoming sensitive to direction. Direction gets defined by the anchor object(s) which in turn is driven by the
Business Intention. Appropriate tools are required to handle the design which needs to be carefully chosen for a project.
Resources

- Read further on Information Framework (SID).
- Download the IBM Redbook publication "Implementing Technology to Support SOA Governance and Management".
- Learn more about how to achieve data interoperability with common-model-based data services inside your service-oriented architecture.
About the author

Gandhi Sivakumar

Gandhi Sivakumar is IBM Certified Senior IT Architect and works on complex SOA based integration projects. Gandhi is a member of IBM Academy and leads many communities in IBM. Gandhi has a deep liking towards innovations and has filed a number of patents.

© Copyright IBM Corporation 2010
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)