Pipeline
Knowledge Is Power

www.pipelinepub.com Volume 3, Issue 12

Achieving Data & System Integration Nirvana with SID

by Kenneth W. Rugg

Over the past several decades, information technology (IT) professionals at service provider organizations have struggled with integrating the diverse operational and business systems present in their environments. While they have in some cases found technologies such as enterprise service busses (ESBs) to connect systems together, they have continued to hit challenges in getting these systems to understand data that is exchanged between them, or put more simply, to "speak the same language." This difficulty led to a desire to develop some form of common model for information exchange.

With the advent of service oriented architectures (SOAs) and new systems interacting with legacy and homegrown systems, the need for a common data model that can be used as a common language in the exchange of data between applications has never been so critical.

Some service providers have developed internal exchange models that are used in limited ways to simplify cross-system integrations but few have achieved a common model across their entire enterprise, let alone one based on an industry standard that could be used across enterprises.

With the Shared Information/Data (SID) model, the TeleManagement Forum (TM Forum) has developed a common language for enterprise operations in the telecommunications industry. It provides a vocabulary for communications across the entire business and operational systems of a service provider as well as a standard format for exchanging information with partners and vendors.

So how are service producers taking full advantage of the SID as a common data model in integration projects for operational and business support systems (OSS/BSS)?

This article will take a fresh look at the SID model and its relevance in addressing the thorny challenges of data interoperability in the integration of OSS/BSS and examine the requirements for software tools to effectively support the SID in application integration.

Why use the SID model?
Think about, for example, a service provider's retail division and its wholesale division; two divisions with two *drastically* different definitions of what constitutes a customer. A retail division has a simple definition of customer; it represents a person with a name, address, and perhaps a credit rating; but nothing too complicated. The wholesale customer, on the other hand, might include many other attributes such as multiple contact points, custom service level agreements, and VAT identifiers among other things. In addition to this, a wholesale customer may also be a vendor or even a competitor in other contexts.

For this and other reasons, it's useful to use the SID model to provide the party abstraction. A party represents any participant in a business interaction. This could be a retail customer, a wholesale customer, or even a vendor or competitor. This allows both individual customers and organizational customers to be represented, enabling both divisions to share the same information and information model.

A Triple Play Case Study
The value of a comprehensive information model is demonstrated in this case study based on a project to implement a Triple Play bundle consisting of:

·Voice, using Voice Over Internet Protocol (VOIP)
·Video, using the Internet Protocol Television (IPTV) protocol
·Data, using high-speed Internet (broadband)

In this project, over a dozen applications supporting nearly 100 operations needed to be integrated. These systems provided functionality that spanned CRM, inventory, activation, and service assurance; designed to provide a "zero-touch" fulfillment and assurance solution. The use of ESB and business process management (BPM) technology enabled integration in an SOA. However, the lack of a common data model was going to necessitate an enormous amount of custom code in order to implement the required data transformations and ensure the semantic consistency of data exchanged between systems.

Using the SID as an exchange data model, to which each system interface needed to be mapped just once, it was possible to reduce integration costs and timeframes

by over 50%. Further, as systems are upgraded, added, or replaced, the use of the SID as a common data model for integration "future proofs" the integration by ensuring that systems can change without impacting other systems with which they are exchanging data.

In complex OSS/BSS environments such as this one, which supports a large number of systems, it is clear that a SID-based data abstraction is necessary in order to support rapid and flexible integration. Although the SID model may at first seem daunting, its adoption as a tool for integration can be dramatically simplified with tools that make it easy to navigate, display, and interact with the SID. And perhaps most important, to map application interfaces to the SID model and generate the runtime services that support true any-to-any interoperability in a loosely coupled architecture.

Is the SID Model Enough?
The short answer is "not always."

For one thing, not everyone will adopt the SID model in all places at the same time. Many legacy systems, whether internally developed or vendor supplied, will remain in place for the foreseeable future and retain their underlying data models. Therefore there is a significant need for tooling that will support mapping between application- or service-specific data structures and the more abstract SID model details.

In addition to this, the SID model is organized generically. It provides an abstract view that is intended to model the entire business operation of a telecommunications service provider. Unfortunately, a model that is rich enough to model a business as complex as this, and to support the degree of interoperability that is required to define the relationships between all the systems and services in this business, must, by necessity be quite complex. The model is complex because it must be large enough to support the sheer scope of the domain and abstract enough to promote the reuse of concepts used in the integration. Because of this, sophisticated tools are necessary to visualize the SID model and to map application-specific views to it.

On the other hand, experience has taught us that no generic data model can anticipate all existing and future use cases that may eventually need to be modeled. While the SID model does a great job of providing a very comprehensive model of a standard service provider's business, there still may be enterprise-specific extensions required to model a particular company's business.

When a service provider adopts the SID model, they will likely find that it does not have all the attributes required to model its business, or that the way that it represents these attributes are not well aligned with the way the business thinks about them. It is tempting to change and "flatten" the SID model, either in UML tools or in XSD representation of the SID. However, this results in a deviation from the standard that will make it difficult to adopt future enhancements of the standard. It is therefore preferable to have tooling that will allow the SID model to be enriched with mapping shortcuts and extensions without deviating from the standard implementation of the SID.

This is where the concept of computed or virtual attributes can be valuable. These are attributes that do not exist in the SID model but are useful to use to map concrete data elements that are represented in abstract ways in the SID's deep class hierarchy, such as "mobile phone number." These computed attributes do not alter the underlying model but can make it much easier for systems analysts and integration developers to work with the SID. In addition, because this approach does not actually alter the underlying model, new versions of the SID can more easily be adopted over time.

It is important for implementation tools to provide this customization without changing the common model itself. In this way the model can remain a standard and be easily upgraded when a new version is published. For example, ensure your technology provider's tools store all that customized information as metadata separate from the SID model so the SID is unchanged.

Since the SID model will evolve over time as the telecommunication business evolves, the tooling that supports the business analysts that interact with it must be designed to accommodate this change. The analyst needs to be able to quickly identify any conflicts resulting in the use of a new version of the SID model in their environment and to assess the impact of these changes on their environment.

Effective implementation tools should be able to map any data item to a SID data item, without requiring any custom coding. Whenever possible it should take advantage of maps that can be reused. Because the mapping of a given interaction may involve more than one legacy system, the implementation tools should also be able to do content-based transformations that analyze the data at runtime and automatically determine the correct format into which the message must be translated. Content-based transformation is distinct from, but analogous to the content based routing done by a message bus. The latter ensures that, based on the content of the message, it is delivered to the right application or service, while the former ensures that the message is in the right format for the service that receives it.

Conclusion

Service providers face increasing pressure to be able to flexibly integrate IT systems in order to rapidly deliver new revenue-generating services and provide an enhanced customer experience. Traditional approaches to integration, even those leveraging newer integration technology, available through ESB and BPM solutions, do not adequately address the challenges of data interoperability in the integration of OSS/BSS. A common data model, implemented to support the real-time exchange of data between systems, is critical to simplify integration and ensure the ability to rapidly reconfigure the integration of systems. The TM Forum's SID model provides an ideal way for service providers to leverage a comprehensive, industry-standard data model in the integration of OSS/BSS, in support of a true loose coupling between systems and enabling that next significant step in data and systems integration Nirvana.

If you have news you'd like to share with Pipeline, contact us at editor@pipelinepub.com.