Using UML service components to represent the SOA architecture pattern

Help stakeholders to better understand service components that comprise an SOA

Prithvi Rao

June 19, 2007

As an architect, you are often challenged -- by client enterprise architects and IT stakeholders -- to articulate Service-Oriented Architecture (SOA) patterns and service components in a nonproprietary, product-agnostic way. In this article, use Unified Modeling Language (UML) models to describe the SOA architecture pattern and its associated service components. You also learn about the service components of the SOA pattern in the context of industry-standard UML formats to help stakeholders to better understand the components that constitute an SOA.

Introduction

*Product-agnostic: This term means that the discussion about using UML to present an SOA solution will be covered in high-level, abstract terms, avoiding product specifics.

The SOA pattern and associated services in this article are discussed in a nonproprietary, vendor-neutral way to clearly demonstrate the benefits of presenting the SOA pattern to stakeholders in an easy-to-understand format.

If you need to present Service-Oriented Architecture (SOA) in a logical format so that stakeholders can better use artifacts to leverage SOA Unified Modeling Language (UML) components in their architecture and design efforts, this article can help. Find out how to leverage service components of the SOA architecture pattern, in the form of UML nodes, to create SOA scenarios. And learn about the SOA pattern and its associated services in a product-agnostic* manner using UML to represent SOA components, connections, and interactions of the SOA architecture pattern.

Logical SOA reference architecture

The SOA pattern is represented in a UML product-agnostic manner in Figure 1. In its simplest form, the SOA pattern consists of a decoupled Enterprise Service Bus (ESB) that connects and provides interactive services among the requestors and providers.
Figure 1. Logical SOA reference architecture

The SOA pattern consists of an ESB infrastructure, with the service interaction points (SIPs), or end points, shown in Table 1.

Table 1. Service interaction points

<table>
<thead>
<tr>
<th>Service</th>
<th>What it provides</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction services</td>
<td>Capabilities and functions to deliver content and data using a portal, or other related Web technologies, to consumers or users</td>
</tr>
<tr>
<td>Process services</td>
<td>Control capabilities to manage the message flow and interactions across multiple services according to business processes and flows.</td>
</tr>
<tr>
<td>Information services</td>
<td>Capabilities to federate, replicate, and transform disparate data sources</td>
</tr>
<tr>
<td>Partner services</td>
<td>Capabilities to integrate partner electronic data interchange (EDI) and legacy systems into the corporate enterprise architecture</td>
</tr>
<tr>
<td>Business application services</td>
<td>Capabilities for service consumers to be called by the business application services</td>
</tr>
<tr>
<td>Application and data access services</td>
<td>Capabilities to integrate core applications with external data repositories and packaged applications</td>
</tr>
</tbody>
</table>

ESB

The ESB serves as the connectivity entry point of the SOA model and provides the following services:

- Request and response services
- Transformations
- Content-based routing
- Customized logging
- Optimization
- Monitoring

ESB also provides the common connectivity and virtualization of services. To address the latest business application demand, the ESB leverages the Service Component Architecture (SCA) programming model.
In Figure 2 you can see an ESB supporting the latest SCA programming model, with a default messaging engine built on Java™ Message Service (JMS) specifications. The ESB uses a mediation component (module) that is based on SCA modules to mediate messages between service requestors and service providers. The mediation services in the ESB can thus be tailored to form complex mediation patterns that implement virtualization in the form of location and identity transparency. They can also improve upon the quality of service (QoS) requirements such as performance, encryption/decryption of messages, and reliable and secure content delivery and transactions.

**Figure 2. Enterprise Service Bus**

![Diagram of Enterprise Service Bus](https://via.placeholder.com/150)

The mediation component is comprised of three components:

**Message models**
- Based on the meta-model of the message under consideration (The ESB should be capable of supporting different types of message models flowing between the service provider and service requestor, thus creating a message-model-agnostic exchange.)

**Mediation flows**
- Contain interfaces to invoke the mediation flows to perform mediation between service requestor and service provider, and to provide references to external services (The mediation flows support several mediation patterns: monitor, modifier, validator, cache, router, discovery clone, and so on.)

**Communication protocols**
- Provide support for different communication protocols, such as MQ, Java Message Service (JMS), HTTP, and Remote Method Invocation (RMI), to connect the service providers with the service consumers (The communication protocols support several interactions patterns, such as request/response, publish/subscribe, and synchronous/asynchronous.)

The ESB uses the service registry and repository component as a dynamic look-up mechanism to provide information about service endpoints. The registry and repository service thus enables optimized access to service metadata and management of service interactions and policies. It also supports the integration and federation of other standard registries and repositories. In its most
elemental state, it is composed of service metadata artifacts documents, such as XML Schema Definition Language (XSD) or Web Services Description Language (WSDL) files. These service metadata files are stored and managed by the service repository.

**Interaction services**

The interaction services that have service integration points with the ESB are shown in Figure 3. The interaction services node serves as the SOA entry point for users. The interaction services provide the presentation layer for SOA that abstracts the interfaces and aggregates the information sources between the end user and the SOA applications.

Interaction services are cataloged into three main services:

**User interface services**
Composed of a portal application that uses dashboards for decision making, as well as visibility into operations

**User interaction services**
Composed of visualization, collaboration, composite applications, alerts and forms

**Deployment services**
Comprised of mobile, browser, and rich clients

Interaction services use the support template components to easily create composite applications. The composite applications:

- Provide the basis for outsourced or in-house service applications
- Support rich clients and mobile end-user clients
- Provide highly customized and dynamic data, which gives real-time visibility that link results to the underlying business process metrics
- Serve as a dashboard, providing users with a real-time view of key performance indicators (KPIs) on the project
Each of the composite applications may contain prebuilt portlets that have a specific function and an associated workflow. Interaction services could also have built-in filtering capabilities, browser-based configuration wizards, interactive Web forms, search, Web 2.0 technologies, and collaboration. For example, the collaboration service component is a completely integrated, portal-based collaborative environment that includes e-mail, calendaring and scheduling, instant messaging, Web conferencing, and document and Web content management.

**Process services**

Figure 4 shows the process services and components it uses to carry out its function within the SOA domain. Process services use the business processes and mediation modules to accomplish its business flow requirements. Process services use the SCA programming model to model the business services that consume and produce business data.
Business processes are defined using Business Process Execution Language (BPEL). A business process is the set of business-related activities, rules, and conditions invoked in a predefined sequence to achieve a business goal. Business rules are a means of implementing and enforcing business policy through externalization of business function. Externalization enables the business rules to be managed independently from other aspects of an application. This independence allows a dynamic business rules update capability, providing for a more agile business.

SCA components consist of interfaces, references, and an implementation. Service components can have interfaces either in Java or WSDL port types. The business process type component consists of a process implementation type that can implement one or more SCA interfaces with either Java or WSDL port types. The processes can be long or short running; the short-running processes are called micro flows. The long-running business processes can interact with multiple partners, with interactions performed through standard, stateless Web service calls.

The interaction with each partner occurs through Web service interfaces. BPEL is built on top of WSDL and XML schemas. The process model is validated using an extended XML schema for syntax and a comprehensive set of rules applied for semantic constraints, as defined by the BPEL specification.

**Information services**

You can see the information services stack for the ESB in Figure 5.
Figure 5. Information services

It has the following components:

**Metadata management**
Provides information about the data (Metadata is information about the data structure and meaning. The metadata management component can manage the metadata and the metamodel and meta-metamodels.)

Metamodels (also known as meta-metadata) define the structure and semantics of the metadata. Examples of standardized metamodels include UML and Common Warehouse Metamodel (CWM). The meta-metamodel layer is comprised of the description of the structure and semantics of meta-metadata. It is an attempt to provide a common language that describes all other models of information. MetaObject Facility (MOF) is a standard used for meta-metamodels. (See Resources for more information.)

**Extract, transform, and load (ETL)**
Extracts, transforms, and loads data from one or more data sources to one or more targets (ETL enables data consolidation, migration, and propagation, and is closely allied with data warehousing and business intelligence functions.)

**Federation**
Uses the data and content classes to federate data across heterogeneous content sources (These decentralized approaches reduce data and content redundancies, bandwidth, storage, ongoing synchronization, and additional administrative cost associated with a centralized approach. Real-time access to distributed information sources also brings new capabilities to business intelligence.)

Federation reduces the need to develop and maintain custom application program interfaces (APIs) for various data sources. Federation also improves performance by caching frequently used data and by using materialized query tables (MQTs) and distributed query optimization and execution. To increase performance, the federated server can also cache and create MQT tables that can be a subset of the rows from the targeted federated data sources to increase performance.
Data placement (replication and caching)
Copies data from one location to another location. The target location can be either a
centralized location, such as data warehouse, or another distributed place on the network.
In a grid environment, replication and cache services are used to create the placement
management service to meet QoS goals. Depending on the access patterns and location of
the consuming applications, a placement management service can improve response time
and information availability by creating caches or replicas. The staged data governance gives
organizations greater control over the life cycle of information flow.

Data modeling
Has an aggregation of the logical, physical, and metadata models for storing the respective
models of the enterprise. These data models have a direct impact on the business and
business applications in the form of data structure and data usage. Careful planning of the data models also improves the overall transaction performance.
There’s a dependency on the type of transaction: online transaction processing (OLTP)
transactions use E/R models; data warehousing transactions use dimensional modeling

Search
Primarily uses its own search mechanisms. The most generic search function is through a
query language, such as SQL and XQuery. Database search is great to retrieve structured
and exactly matched data, but it requires familiarity with the data model structure to construct
the queries accordingly.
Database search is limited in scope; it is not designed for relevance ranking, fuzzy search,
or multiple keywords. For search engines to achieve higher performance, flexibility, or
relevance ranking, the engine sometimes connects to the databases directly to extract data
and generate indexes from databases.

Analytics
Assists in better decision making, data mining, and cross-departmental reporting. The
traditional analytics include reporting, data mining, dashboards, scorecards, and business
performance management. Organizations want to access heterogeneous data sources in real
time to ensure regulation compliance, respond to customers better, predict market trends, and
improve on operational efficiency.
The analytics component is closely tied to the function and features of composite applications
of the interactions services and provides a real-time view of KPIs for business applications.
In the future, analytics will build increased intelligence to access and correlate information
from heterogeneous information sources to provide new insights to make better business
decisions.

Partner services
The partner services that act as the reuse entry point for SOA are shown in Figure 6. This
implies that legacy systems and electronic data interchange (EDI) systems can connect to the
SOA enterprise architecture with the help of custom adapters and hook up to the ESB, thereby
increasing operational efficiency and QoS. Adapters provide communication between the EIS and
the integration broker. Each back-end system or business application requires a specific adapter.
Business integration adapters consist of a collection of software APIs, providing native communication with the back-end enterprise information system (EIS), and tools that let you configure business objects and adapters. Partner services using custom adapters depend on various software vendors, including SAP, Siebel, and IBM® WebSphere® (which offers the WebSphere Adapter Toolkit, as well as dozens of adapters, including the WebSphere Adapter for SAP Software and the WebSphere Adapter for Siebel Business Applications).

**Figure 6. Partner services**

Figure 7 shows a common pattern used in business integration that involves the need to synchronize semantically similar data among various back-end business application systems. The scenario shows two different back-end systems, each integrated with the business integration application running on the process server, through the use of business integration adapters. The integration of the adapters is achieved by leveraging the same SCA programming models and components used by other integration applications.
The center of Figure 7 shows the process server with a business integration application. The business integration application is made available for invocation to other services, outside of the SCA module, through a JMS export. The business integration application is able to invoke other services outside of the SCA module through the use of a JMS import. The adapters communicate with the back-end systems using the application-specific data structure or business object and are configured using the connector configuration file.

When a business object is passed inbound to the process server through the export, it is converted to a format understood by the process server by a data binding that is part of the export. When a business object is passed outbound to the adapter, it is converted to a format understood by the adapter by a data binding that is part of the import. This data synchronization pattern can also incorporate mapping of the business object from an application-specific format to a generic format.

Some of the vendors who partner in supplying application adapter components include Ariba Buyer, Clarify, MatrixOne (eMatrix), JD Edwards, mySAP.com, Oracle applications, PeopleSoft Portal intranet, QAD MFG/PRO, Retek, SAP Exchange Infrastructure, Siebel, WebSphere Process Server, and WebSphere Enterprise Service Bus (ESB).

Some of the technology adapter components are: ACORD XML, Microsoft® COM, CORBA, e-mail, EJB, Microsoft Exchange, FIX Protocol, IBM iSeries®, WebSphere Business Integration iSoft Peer-to-Peer Agent, Java Database Connectivity (JDBC) (SQL and stored procedure access), JMS, JText, IBM Lotus® Domino®, Society for Worldwide Interbank Financial Telecommunication (SWIFT), WebSphere MQ, WebSphere Business Integration Message Broker, WebSphere MQ Workflow, Web services, and XML.

Some of the technology adapters can use data handlers, including data handlers for EDI, SOAP, XML, and various text formats.

The adapters are built using a common adapter framework and are:

- Bidirectional, in that they are capable of event processing and request processing
- Configurable through metadata and capable of processing multithreaded business objects.

The business integration collaboration component has collaborations with components such as customer relationship management (CRM), Health Insurance Portability and Accountability Act...
Act (HIPAA), health care, order management, procurement, telecommunication, life insurance, and so on. Business integration collaboration is done with prebuilt templates that streamline and synchronize information and data related to the respective components.

For example, collaborations for HIPAA transactions enable compliance with required specifications and standards, and ensure that all transactions and interactions interconnect across multiple applications and across enterprise boundaries. The broker plug-in component provides the necessary classes required for creating a user plug-in node. The microbroker plug-in component provides the necessary access-related information, such as broker name, queue name, data source, and so on.

**Business application services**

Business application services constitute the reuse entry point for SOA. Business applications are loosely coupled to bring business value to the enterprise by using Web services.

Web services reduce the cost of building expensive business applications and enable the deployment of new business models in the enterprise structure. For most organizations, the primary challenge with rapid deployments into the mainstream is security. Business application services incorporate some of the business security features to ensure security during business transactions.

**Figure 8** shows the business application services that use the business process and policy management component to provide business security services to the business applications of the enterprise.

**Figure 8. Business application services**

The business process and policy management component uses the following security components to fulfill its security obligations to the business applications.
Security governance framework
Addresses the need for effective governance structure and decision-making authority within an enterprise (This framework is used to establish the chain of command, responsibilities, and authority to ensure that enterprise business applications are controlled effectively from a security perspective.)

Trust management
Establishes trust between any two enterprises or organizations coming together to conduct secure business transactions (Trust is established with the two entities agreeing to abide by a set of relationship and liability management rules to conduct business.)
Trust is also established from a technology perspective using cryptographic methods, including encryption keys, private or public keys, digital signatures, or protocols.

Identity and access management
Provides the necessary ID management and access privileges across enterprises. This component uses the following additional components to fulfill its services:
• HR identity feed, to notify the administrative authorities about ID status changes and to trigger appropriate workflow initiations
• Approval component, to get the necessary management approval for modifications or updates to identity or information access
• User self-care component, to enable end users to carry out certain security administrative tasks without administrative supervision, such as periodically changing a password
• Delegation component, to provide the ability to delegate the IT security management functions to another individual
• Revalidation component, to provide access to systems that need to be approved at regular intervals

Secure systems and network
Provides security services such as firewalls, intrusion detection systems, virus detection, patch management, and operating and network security (The Business application services component uses a federated component to manage many different service environments of the enterprise.)

Data protection
Responsible for the security of the business information content during its transportation or at its destination (The data protection component establishes trust by managing privacy policies, obtaining detailed reports on sensitive information, publishing policies for review by users, and capturing and enforcing user preferences.)

Risk management
Determines the level of risk associated within the enterprise IT system and the need to take effective action to assess its impact on overall security operations based on a cost benefit analysis and its impact on the system

Compliance management
Ensures compliance with external federal or state regulations and internal compliance with IT organization business security policies
Audit trails and records
Reconciles and assesses how the different IT security policies introduced in the IT system are practically applied in day-to-day operations to ensure compliance that is set against internal and external policies. (This helps management and technical teams take quick corrective actions in case of policy divergence.)

Application and data access services
The application and data access services component serves as the information and reuse entry points of SOA, as shown in Figure 9. Connectivity of services across heterogeneous technologies is the foundation of SOA. Figure 9 shows a scenario of an enterprise application supporting various interaction protocols and QoS with the application and data access services component. The business applications of most organizations must be able to handle a variety of data representations as they start moving toward exposing applications as services in an SOA environment.

Handling the myriad forms of data turns out to be a challenge; there is an imperative need for a common universal API capable of handling the various data sources. The SCA programming model can expose services that interact with the underlying data layer. A robust data access utility called the relational database data access service (RDB DAS) provides a tight integration with Service Data Objects (SDO) to access data within an SCA-based application.

Figure 9. Application and data access services

Summary
Architects and stakeholders often find it challenging to articulate the SOA architecture pattern and to determine which entry point to choose. As a result, they may want to select several SOA entry points that can address the most pressing and challenging issues that face the enterprise. After
reading this article, architects and other key players can help their organization better meet its requirements by drilling down on the various nodes identified at each of the entry points to begin working toward their SOA architecture journey.

In this article, you learned about SOA from a logical perspective and how to create nodes and associated UML components that can represent the SOA reference architecture in a nonproprietary, product-agnostic manner.

**Acknowledgments**

The author thanks Dr. Pavlick, IBM DoD CTO, for providing the Federal market need for logical SOA models and a repeatable process that can help sales and architecture teams quickly respond to customers with a repository of SOA nodes.

© Copyright IBM Corporation 2007
**Trademarks**
(www.ibm.com/developerworks/ibm/trademarks/)