

SOA Best Practices & Framework Services in Order to Invoice Enterprise Application Integrations

A White Paper

Oracle Collaborate, April 2008

By

**Raman D. Singh
Consulting Manager, SOA Practice
Protégé Software Services**

Overview

At Monster, application integrations of Oracle's Siebel, Oracle E-Business Suite, and other business applications leverage solutions built on the BPEL components of the Oracle SOA Suite. This white paper shows why and how a framework of services has been developed to provide a common solution for data cross-reference and exception handling. Also we will review enterprise integration patterns including use of canonical data models to enhance your BPEL and ESB solutions.

Business Case

An order for Monster represents the acceptance of the quote and, therefore, a contract between the corporate customer and Monster. Orders contain relevant customer (address, contact, profile) and order data (products, prices, discounts, contract durations and General Ledger accounts). The main business requirement was to integrate Siebel Orders and Oracle Financials for Europe implementation, incorporating order validation and data transformation into the process.

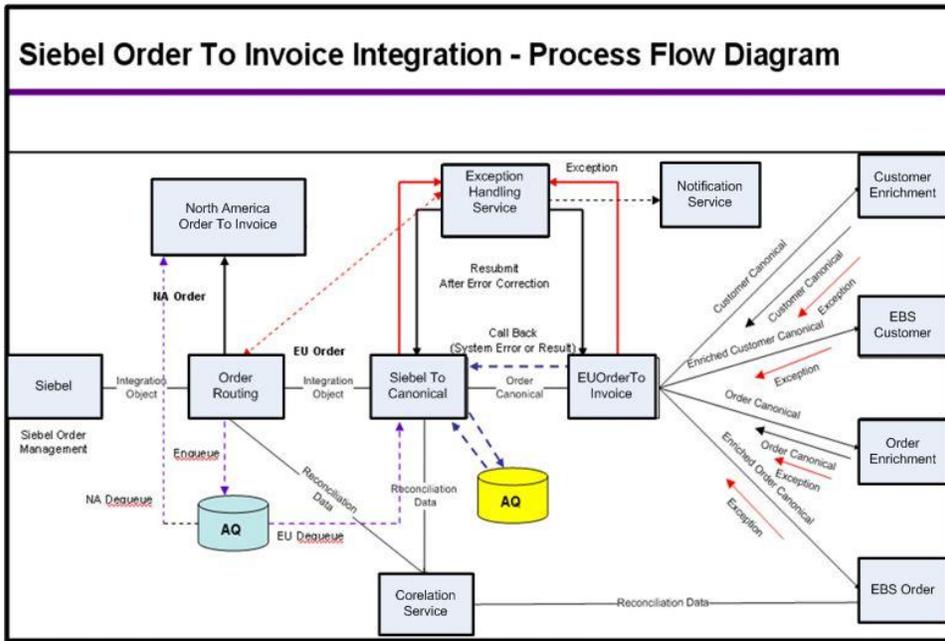
SOA approach met following business requirements,

- Successful enterprise integration to establish communication between multiple computer systems, as well as between business units and IT departments
- Middleware is separate of the participating applications. Needs to have functionality and identity of its own, as it has limited control over participating applications.
- Middleware needs to provide its own services, that support its functionality, that are consistent and comprehensive
- Write services once, run & reuse anywhere.

Solution Details

Monster has a single implementation of Siebel CRM, providing Order Management functionality; and, has two implementations of Oracle Financials, one each for North America and Europe, providing customer invoice processing. Monster chose Oracle's SOA Suite as its integration middleware platform, including the management of the order-to-invoice process through the enterprise.

This solution facilitates the near real time integration of invoices and customers from Siebel to the European Oracle Financials instance through the Oracle SOA Suite in an extensible manner. In the process flow diagram, the BPEL OrderRouting process receives an order integration object from Siebel. The Siebel adapter channel is configured to listen using http on a port where the Siebel integration object is posted and then delivered to the BPEL engine in a JCA inbound interaction pattern. Using content based routing, the order is routed to either the North America BPEL (resident on an North America-based SOA Suite installation) or Europe BPEL (resident on a European-based SOA Suite installation) process. This solution brief describes the flow for European orders that are routed to the European SiebelToCanonical process.



After transforming the Siebel order payload to separate customer and order canonicals, the EUOrderToInvoice process is invoked. The EUOrderToInvoice process uses a series of pick activities and 'onMessage' handlers to process callback 'onResult' or 'on Error' messages from Customer Enrichment, EBSCustomer, OrderEnrichment and EBSOrder processes. It uses enrichment processes for data transformation and validation using a database adapter-based cross reference utility. For example, Oracle E-Business Suite payment terms are cross referenced from the incoming Siebel order payment terms using a DB adapter lookup. After the successful validation and enrichment, the EUOrderToInvoice process invokes EBSCustomer and EBSOrder processes to insert data into the customer and invoice staging tables via database adapters.

Any data or payload related errors are routed to Exception Handling processes that take order and exception canonicals as input payloads. The Exception Handling processes use BPEL human workflow task services for human intervention and exception notification by e-mail. Oracle's human workflow services and out-of-box worklist application are leveraged to provide a user interface to failed orders. The user interface provides functionality to modify data and acknowledge / resubmit orders. The user interface is customized to present the payload to the user as a set of form field elements. The users and roles are defined in the JAZN repository of Oracle Application server. Notifications and reminders on unattended tasks are sent out via email.

The solution makes use of Oracle Advanced Queues at various integration points manage resiliency of the process in the event that the European SOA environment or the European E-Business Suite (EBS) environment is down due to a failure or for maintenance. It enqueues messages in the North America SOA middleware platform until the messages are dequeued or consumed by turning on the BPEL dequeue process from the BPEL console when the environment is ready to process the orders. The correlation service is a process to reconcile order data across the BPEL processes to ensure that there is no loss of orders due to BPEL errors such as transaction timeouts, remote faults, failure to recover messages from delivery or callback queues etc.

Framework Services

Exception handling

Common service that handles all exceptions generated during the execution of the BPEL and ESB processes.

Exceptions are normal and can be expected throughout architecture.

- Includes logical errors such as coding errors, mapping failures, validation failures, business rule violation, stored procedure failure, or table constraint violations. In addition, technical failures can cause exceptions such as database down, memory limitations, endpoint system down, and network problems.

Exceptions have potentially significant affect,

- Exceptions are real-time events that might impact not only a single transaction, but could also disrupt a critical business process.

Exception Implementation framework architected to,

- Allow exceptions themselves to be an actionable entity.
- Exceptions are very descriptive and contain contextual information about the data and service that failed.
- Framework dispatches these exceptions to the appropriate process.
- Exception is propagated to the appropriate user based on context and defined business flow.

Exception handling Details

- Framework Service that handles application or business faults thrown anywhere within BPEL Processes.
- Leverages human workflow and task services to provide exception notification and integration with Oracle Worklist application.
- Worklist Application User Interface to present , fix and resubmit orders that failed within middleware.
- Uses Jazn repository to manage user credentials.

Cross-reference handling

Common service that performs table-based look-ups for data transformation.

Data transformation is regularly required between applications,

- Data values will need to be transformed as data moves from one application to another to accommodate the data validation requirements of the consuming applications.

Requires flexibility,

- Cross reference source/target values will be volatile over the life of the applications.
- Data ownership will be defined for the source/target values.
- A user-interface for data modification must be provided.

Cross Reference Implementation - framework architected to,

- Pass a cross-reference type and source value to a service
- A target value is returned from the service.
- If the cross-reference is not successful, an exception is returned, to be handled by the exception handling service.

Cross-reference handling Details

- Utility to perform data enrichment and validation on customer and order data.
- Leverages database adapter as interface for lookups.
- Cross-reference tables for product, sales-channels, payments terms, error codes, VAT codes, country, and language.
- Custom JSP Web Application user interface to maintain tables.

Integration patterns

Canonical data models

Common data model used by each participating application to produce and consume messages in a common format for the data object.

- Enterprise level information model representing business entities that can be exchanged across systems
- Structure for any given entity remains consistent regardless of its intended use - whether it is produced by an ERP application or consumed by a CRM application.

Key Benefits

- Endpoint solutions remain decoupled from all other applications. Minimize dependencies when integrating applications that use different data formats with a canonical data model that is independent from any specific application.
- A high level of abstraction can be maintained. The technology solution can be aligned with the business.
- Require each application to produce and consume messages in the common format.

Implementation

- Canonical entities are defined by schemas that define the syntax rules for the entity. The schema defines attributes, data types, and constraints.
- When designing canonical documents in the analysis phase of a project, there are a number of approaches for deriving the enterprise information model.
 - Adopt industry standard schema
 - Adopt prominent endpoint schema
 - Model ground-up from business requirements.

SOA BPEL Best Practices

- Use framework services for use by the BPEL process for example exception handling, cross reference, etc.
- Design decoupled BPEL processes.
Makes it flexible, reusable, minimizes dependencies.
- Maintain Common Schema Repository.
URL based XSD references, no local copies.
- Use Canonical Schema Models.
Standard based business definition, common vocabulary.
- Oracle Advanced Queues for process resubmission, error handling.
Better control, DB persistence, improved performance.
- Managing dehydration space is critical.
Space allocation, Archive strategy, Purge strategy.
- Manage BPEL Domains.
Partition BPEL processes as per business functionality, configurations, control.
- BPEL Process Manager tuning to optimize performance.
Worker bean threads, dspMaxThreads, JTA Transaction timeout, SyncMaxWaitTime, Idempotent.
- BPEL Dehydration Database performance tuning.
Processes, sessions, SGA memory, Redo log file size, separate BPEL tablespace.

Conclusion

In conclusion, this BPEL process to integrate orders from Siebel with Oracle Financials has been designed using best practices such as canonicals, common schema repository and loosely coupled individual processes that make most of the code reusable and scalable.

The use of BPEL human workflow service for exception handling and order resubmission ensures that data is validated and fixed within middleware by business, thus speeding the process and resulting in near real time revenue recognition for orders placed in European countries.