Oracle Application Integration Architecture (AIA) Does It Deliver On It's Integration Promise?

Ahmed Aboulnaga, Apurva Singh
IPN Web, Inc.
USA

Keywords:

Oracle, Application Integration Architecture, Service Oriented Architecture, AIA, SOA

Introduction

A typical Information Technology department in an organization undergoes many transformations in response to the needs of the business which drives it. In a business looking to stay ahead of its competition, change is the only constant. What we see as a result of such an evolution is the existence of disparate enterprise-class IT systems which have been inducted into the organization at various points in time to satisfy the most important business requirements at the time.

These systems could be based on different platforms, procured from different vendors, or custom built. Given the torrid pace of globalization, some of these systems maybe located in different geographies and with outside partners. No matter how good a system may work standalone, it is rarely designed with integration in mind.

Enterprise integration is the task of making disparate systems work together to produce a unified set of functionality.

Integration is important because if applications are to continuously deliver value in the face of ever changing business landscape, they cannot live isolated from each other. We need techniques that allow us to take applications that were never designed to interoperate and break down the stovepipes so we can gain a greater benefit that the individual applications can offer us. It also allows us to deliver substantial value to the business by integrating functionality into more useful services.

Various technologies have been around that promise to solve the integration puzzle. Today we will look at one of such technology, Oracle's Application Integration Architecture (AIA), designed to solve the problem of integrating the various disparate Oracle applications that have mushroomed in the enterprises all around the world.

Integration Patterns

If we could anticipate our future business requirements and somehow designed a system which could incorporate all future requirements as they came up, we wouldn't be talking about integration. The stark reality, however, is that we almost always can't anticipate our future business or integration needs. This has given rise to multiple classes of enterprise applications and dozens of different ways to integrate them.

Each integration pattern learns from the previous one and aims to make it better by solving the problem in a more clever way. We briefly talk about the two common integration patterns followed by an explanation of the integration pattern utilized by Oracle Application Integration Architecture.

Point-to-Point Integrations

Interfaces contain the logic for connectivity with the source applications, message transformation, and connectivity with target applications. In the typical point-to-point integration, the source and target applications are *tightly coupled*. This offers limited scalability, as adding a new application will almost always result in the development of a new interface as shown in Illustration 1.

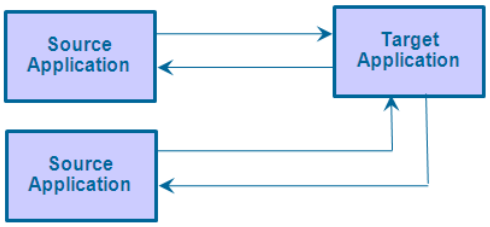

Illustration. 1: Point-to-point integrations lead to tight coupling

SOA Integrations

With the introduction of Service Oriented Architecture, otherwise known as SOA, integrations are developed with *loose coupling* in mind. Source interfaces are built independent of target applications. This allows the ability to scale and add new target interfaces with limited changes to the existing integrations. Illustration 2 demonstrates how adding a new source application does not change the interface to the target application.

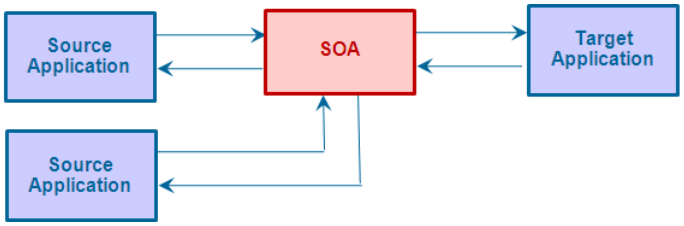

Illustration. 2: SOA developed integrations lead to a higher degree of reuse

Oracle Application Integration Architecture

Through organic growth and multiple acquisitions, Oracle has become an enterprise application powerhouse. Hundreds of thousands of organizations across the globe have multiple Oracle applications running. Almost all of them have a business need to integrate them at the data, functional, process or UI level.

As is to be expected, they were approaching this problem to the best of their abilities. But almost of them were reinventing the wheel, so to speak. Oracle however, noticed a pattern in what was going on and in that, an opportunity to standardize the process of integration amongst various Oracle applications.

It launched Oracle Application Integration Architecture.

The left figure in Illustration 3 shows the standard point-to-point integrations among multiple applications. Each integration is built separately, allows for limited reusability, consists of multiple transformations, and more seriously, as the number of applications grow, the points of connection grow exponentially, making it much more difficult and more expensive to maintain.

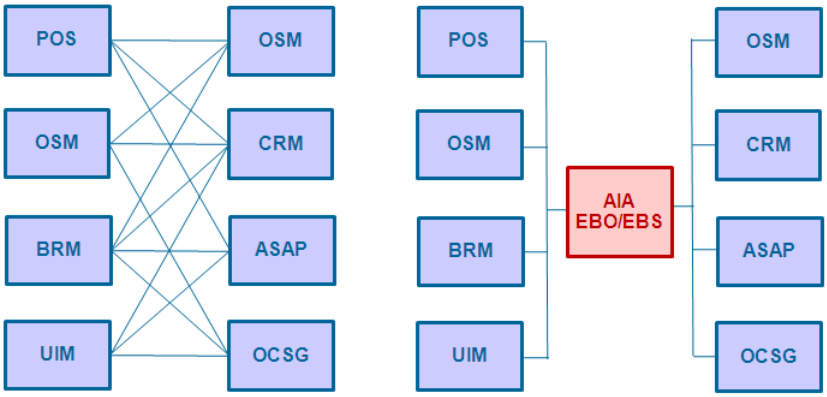

Illustration. 3: Illustration of point-to-point integrations (left) compared to AIA-based integrations (right)

The figure on the right of Illustration 3 demonstrates how, by leveraging AIA, which is based on true Service Oriented Architecture concepts, interfaces can be reused, the number of transformation reduced, and overall maintenance is reduced.

Oracle AIA is not a novel concept. It is merely a formalization of an observed integration pattern and a set of best practices and frameworks, which if adhered to, promise to make integration of Oracle applications easier and more straightforward. The main components of Oracle AIA include:

1. Process Integration Packs (PIPs)
2. Enterprise Business Objects (EBO)
3. Development Methodology
4. Error Handling Framework
5. Composite Application Validation System (CAVS)
6. Other tools and features (e.g., Project Lifecycle Workbench, PIP Auditor, etc.)

These will be discussed in varying detail in the remainder of this paper.

From an architectural standpoint, the underlying J2EE application server is usually Oracle WebLogic Server, over which Oracle SOA Suite is installed. The AIA Foundation Pack is built and installed over Oracle SOA Suite, in which pre-built integrations, otherwise known as PIPs, are based on the AIA Foundation Pack.

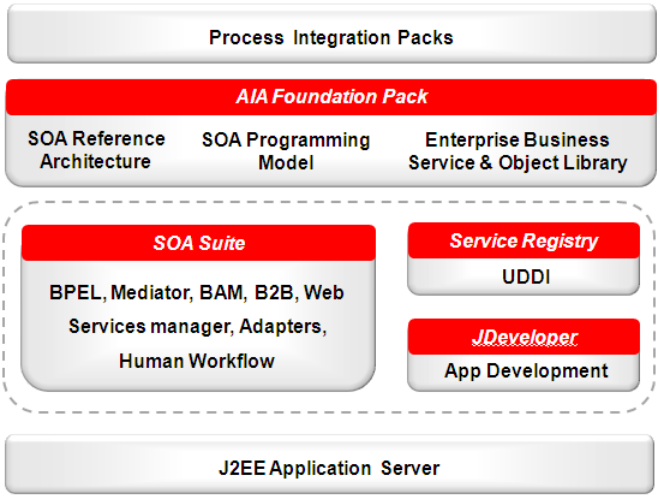

Illustration. 4: High-level AIA architectural framework

Enterprise Business Objects – The Canonical Data Model

One of the more valuable components included with the AIA Foundation Pack are EBOs, or Enterprise Business Objects. EBOs are purely canonical data model representations. Companies often spend many months just trying to agree on a corporate data standard to represent their key business objects. For example, as shown in Illustration 5, applications A, B, and C, may have different internal representations of a *customer*. By leveraging the Customer Party EBO, which is essentially a superset of customer elements, the source application is responsible for publishing as much information to the EBO as possible. The integration will transform this data from the EBO to the format of the target systems.

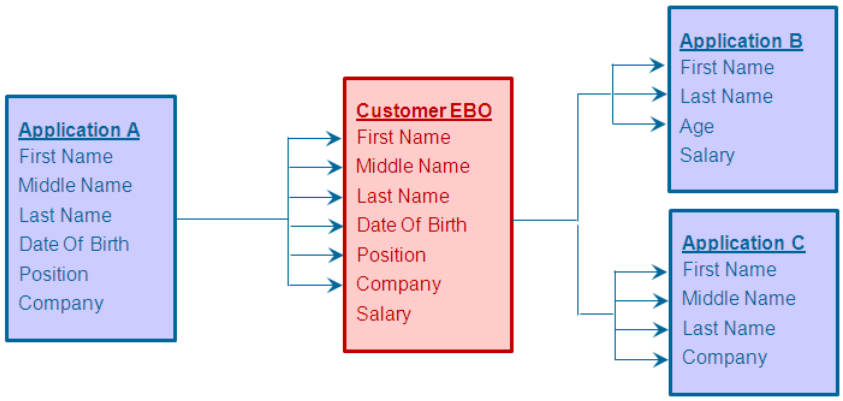

Illustration. 5: Understanding the usage of the canonical data model, or EBO (Enterprise Business Object)

EBOs are based on the Open Applications Group content known as OAGIS and have been extended to incorporate best-of-breed attributes. They are designed with extensibility in mind, and eliminates the need to comprehensively analyze your environment to determine a common message format.

From a technical standpoint, EBOs are merely XML schemas (i.e., xsd files). Example of some Oracle AIA EBOs include:

- BankAccount

- CurrencyExchange
- CustomerParty
- Invoice
- Item
- Location
- PurchaseOrder

The AIA Development Methodology

In order to promote loose coupling between applications and increase the reusability of the integrations, AIA proposes a methodology to follow when developing custom integrations. This methodology is designed to decouple the source and target applications. Thus, if a change is made to the target application, it is virtually unknown to the source. Updates to any application requires little to no changes to your integrations as a result. PIPs, which are essentially pre-built integrations from Oracle, are developed using this methodology.

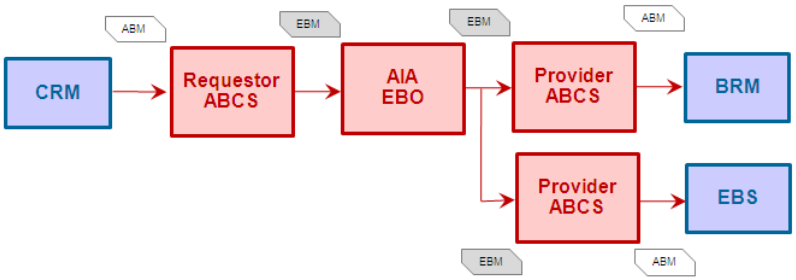

Illustration. 6: Demonstrating the Oracle AIA development methodology

In the illustration above, the end-to-end integration is separated into 3 separated components. The Requestor ABCS, which is tied to the source application, the Provider ABCS, which is tied to the target application, and the EBS (Enterprise Business Service), which is the router in between.

As shown in the illustration, the Requestor ABCS transforms the custom message (ABM) into the common canonical format (EBM). The EBS, shown in the middle, routes to the appropriate target application. The Provider ABCS accepts the message in the common canonical format and transforms it into the target applications custom format.

AIA Error Handling Framework

The AIA Foundation Pack also delivers and error handling framework. This is detailed in various Oracle documentation, but in summary, it can capture errors, notify on these errors, and provide workflow processes surrounding the management and assignment of these errors. The error handling framework is used by PIPs and can be used by custom SOA integrations as well. Via the BPM worklist application shown in Illustration 7, errors can be claimed, escalated, and delegated to users, such as the Tier 2 help desk.

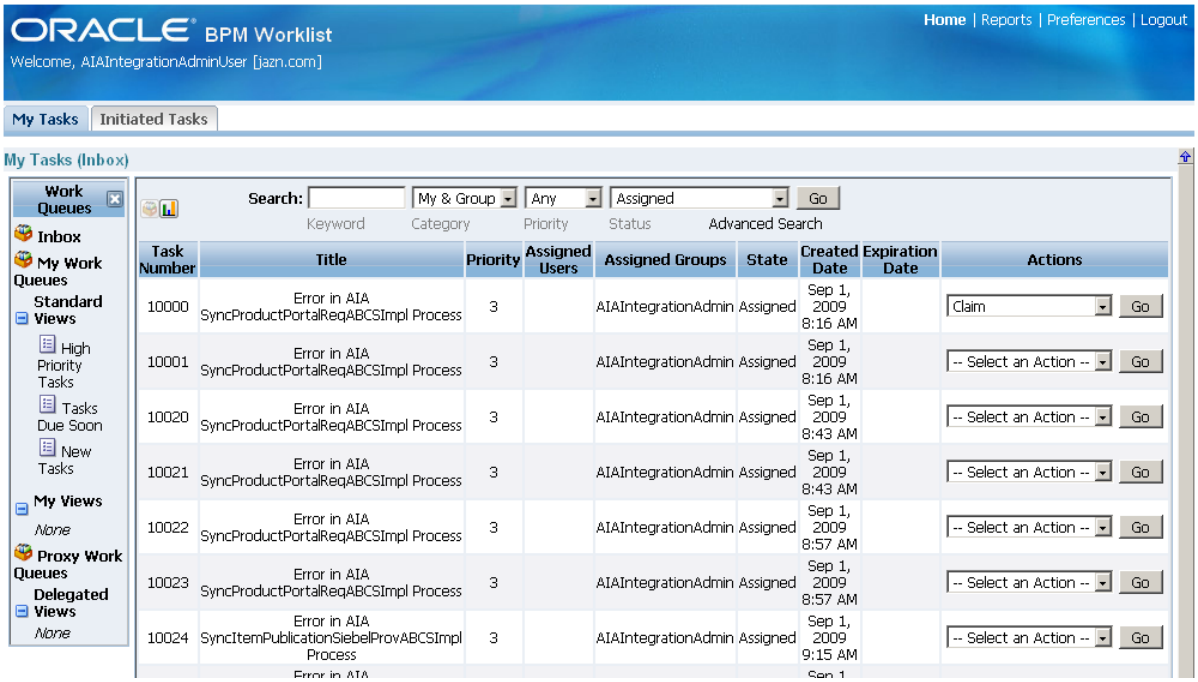

Illustration. 7: Screenshot of the BPM worklist to claim errors captured by the AIA Error Handling Framework

Composite Application Validation System (CAVS)

The Composite Application Validation System, or CAVS, is a testing tool delivered with the Oracle AIA Foundation Pack. It is a framework to test integration of AIA services. CAVS provides test initiators that simulate web service invocations and simulators that simulate service endpoints.

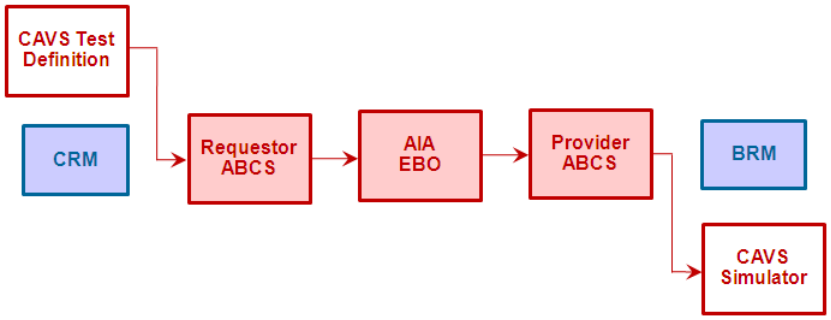

Illustration. 8: CAVS (Composite Validation Application System) usage

This becomes valuable when the target system may be unavailable. By leveraging the CAVS simulator, developers can continue testing without being hindered by the unavailability of the target application. However, this is somewhat limited as responses are hardcoded. Thus, the CAVS simulator is good for binding tests, but not necessarily data validation tests.

The CAVS test definition can simulate web service invocations, similar to tools such as SoapUI.

Other AIA Foundation Pack Features and Capabilities

The Oracle AIA Foundation Pack delivers additional features such as the Project Lifecycle Workbench, the PIP Auditor, open APIs, and much more.

For example, functional designs are created to specify requirements that need to be implemented for an integration project. The Project Lifecycle Workbench is used to perform functional decompositions to break down overall projects into business tasks.

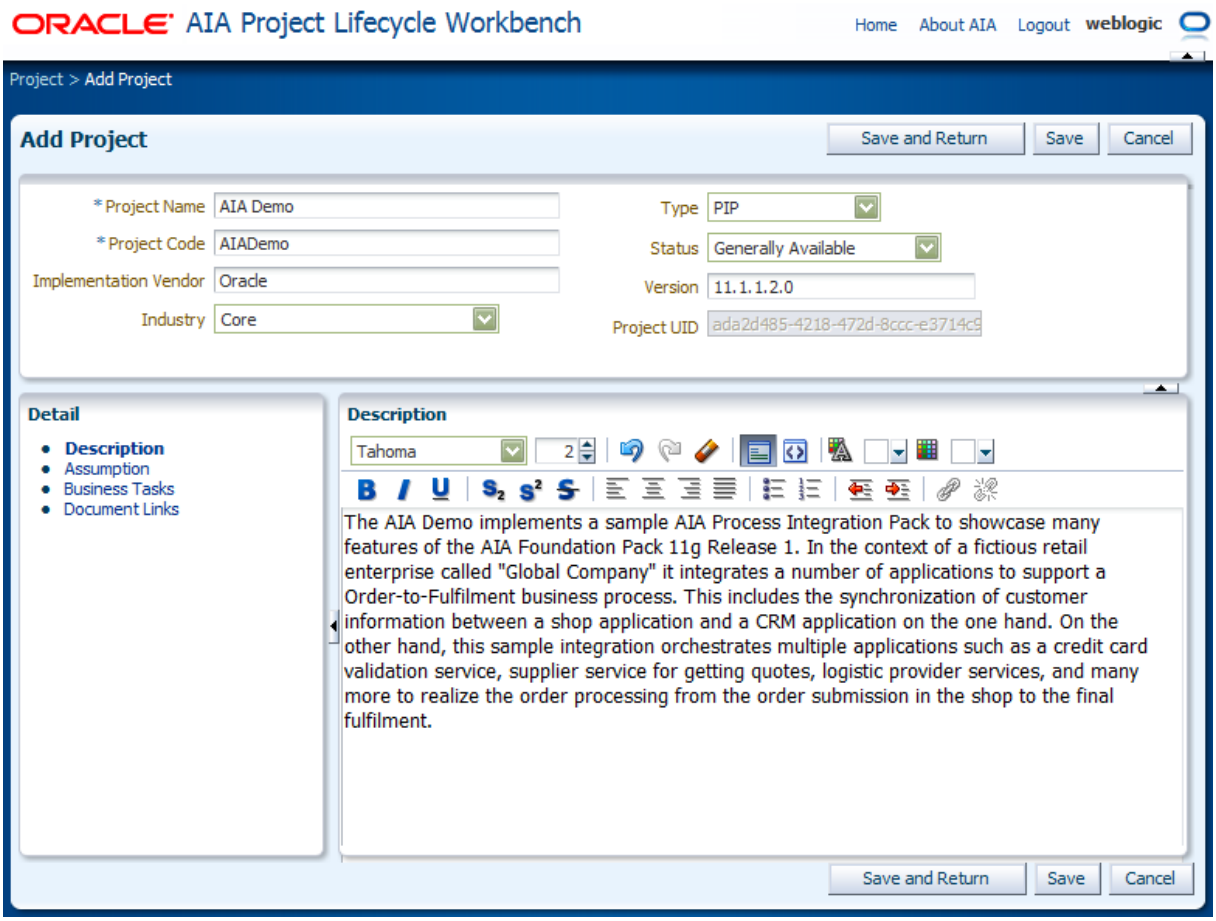

Illustration. 9: Screenshot of the AIA Project Lifecycle Workbench 'Add Project' page

Truth Meter

At DOAG Applications 2011, IPN Web presented use cases for two large customers who used various features of Oracle Application Integration Architecture, and the successes and challenges faced by each in their massive integration implementations. In addition, IPN Web conducted a survey with four leading Oracle AIA Solution Architects, one a leading architect for a global SOA implementation, two architects are leading and well known Oracle SOA and AIA consultants with experience supporting numerous enterprise integration implementations, and the last a senior integration architect from Oracle Consulting Services. The survey was conducted to determine the truth in statements made by Oracle in regards to what AIA can deliver, based on real world implementation experience.

The results are summarized in the Illustration 10 below.

Oracle's Claim		Truth Meter
Reduces complexity, accelerates delivery		35%
Promotes reuse		75%
Leverages industry best practices		73%
Speeds up design with pre-built, extensible product data model		75%
Allocates less work on maintenance		43%
Lowers integration costs	(using PIPs)	56%
Lowers integration costs	(not using PIPs)	65%
Reduces integration risks	(using PIPs)	63%
Reduces integration risks	(not using PIPs)	43%

Illustration. 10: The Oracle AIA Truth Meter – based on results from 4 leading Oracle AIA Architects

Contact address:

Name
IPN Web, Inc.
2275 Research Blvd., Suite 500
Rockville, MD 20850
USA

Phone: +1-301-296-4234
Email info@ipnweb.com
Internet: www.ipnweb.com