DOAG
Deutsche ORACLE-Anwendergruppe e.V.

ipnweb

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

Ahmed Aboulnaga
Technical Director
IPN Web, Inc.

Apurva Singh
Technical Director
IPN Web, Inc.

Agenda

1. Introduction
2. The Integration Problem
3. Introducing Oracle AIA
4. AIA Usage in the Real World
5. Summary

IPN Web at a Glance

- Founded in 2004
- Headquartered in the Washington DC area
- Specializes in Oracle Fusion Middleware implementations
- Certified staff in multiple disciplines (OCE, OCP, OCA)
- ORACLE Gold Partner

www.ipnweb.com

The Integration Problem

Why do we need Application Integration?

- Existence of multiple disparate systems in an enterprise and their role in performing larger business functions
- These systems are disparate in their:
  - Technical Architectures
  - Programming Model
  - Connectivity
- Integrations between the systems done at multiple levels:
  - User Interface
  - Functional Level
  - Data Level
  - Process Level

Competitive Edge

- Need to stay ahead of competition

  Introduce new products and services, launch new business models

- Get closer to the customer

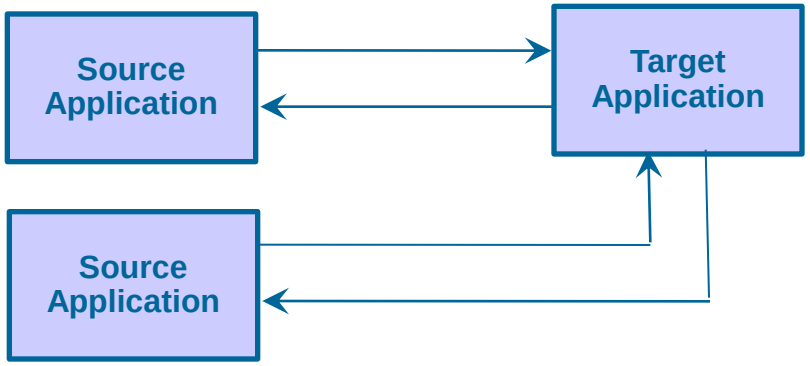
  Accurate and timely customer information on demand

- Streamline operations

  Lower costs and reduced cycle times

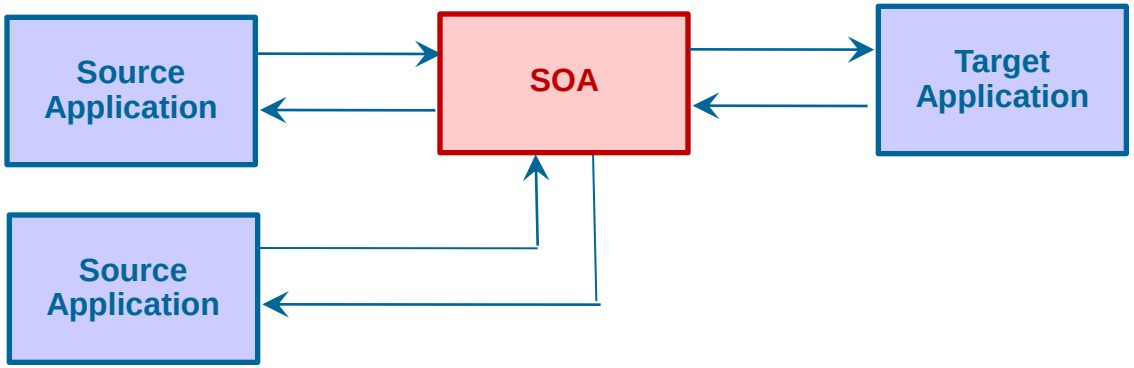
Integration Pattern: Point-to-Point

- Interface contains the logic for:
  - Connectivity with Source
  - Message transformation and
  - Connectivity with target applications
- Source and target applications are *tightly coupled*
- No Scalability – Adding a new application will result in a new interface



Integration Pattern: SOA Integrations

- Source interfaces are built independent of target Applications
- Adopting a Common Data Model is one variant
- Scalable to add new target interfaces

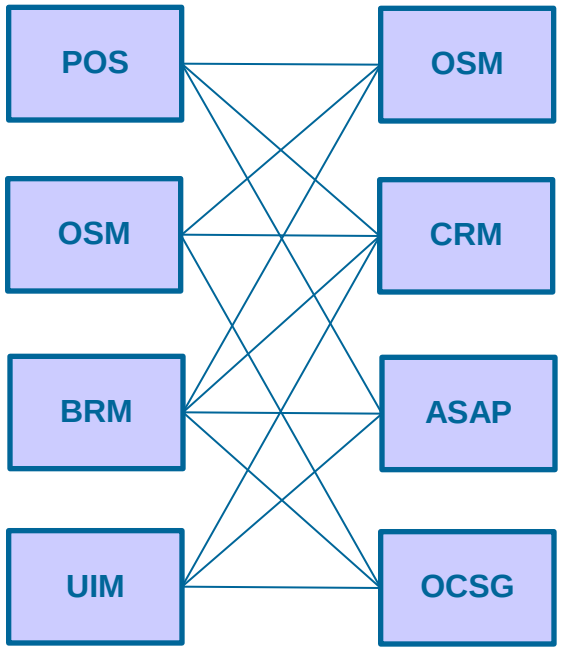


Integration Pattern: Oracle Communications

- A complete, end-to-end solution for Communication Service Providers (CSP) to deliver mobile TV, 3G wireless networks, video conferencing, and more
- The Oracle Communications Suite consists of the following applications:
  - Siebel Customer Relationship Management (CRM)
  - Communications Billing & Revenue Management (BRM)
  - Supply Chain Management (SCM)
  - Point-of-Sale (POS)
  - Service Fulfillment (SF)
  - Communications Order & Service Management (OSM)
  - Communications Unified Inventory Management (UIM)
  - Automatic Service Activation Program (ASAP)
  - Communications Services Gatekeeper (OCSG)
  - Oracle E-Business Suite & Financials

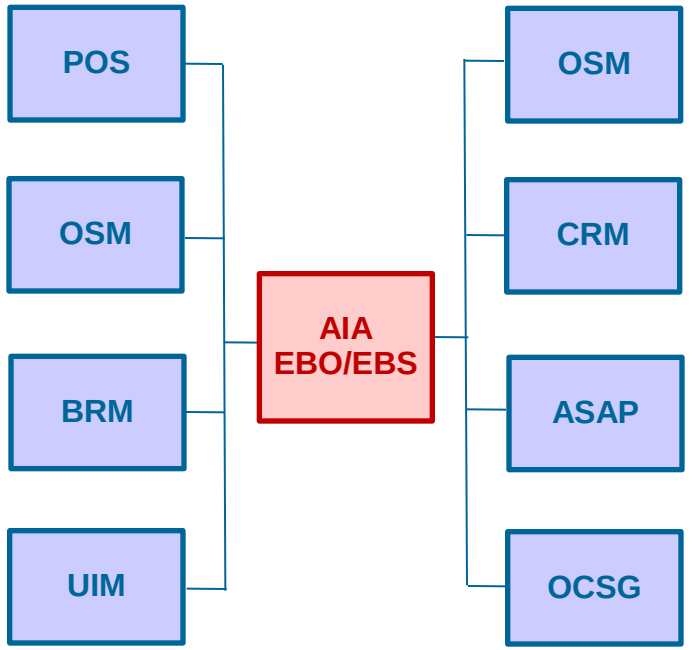
Integration Architecture: Using Web Services

- Using web services alone is not enough
- Integrating these applications can be very difficult and time consuming
- Upgrading any application will require code modification to the integrations



Integration Architecture: Using AIA

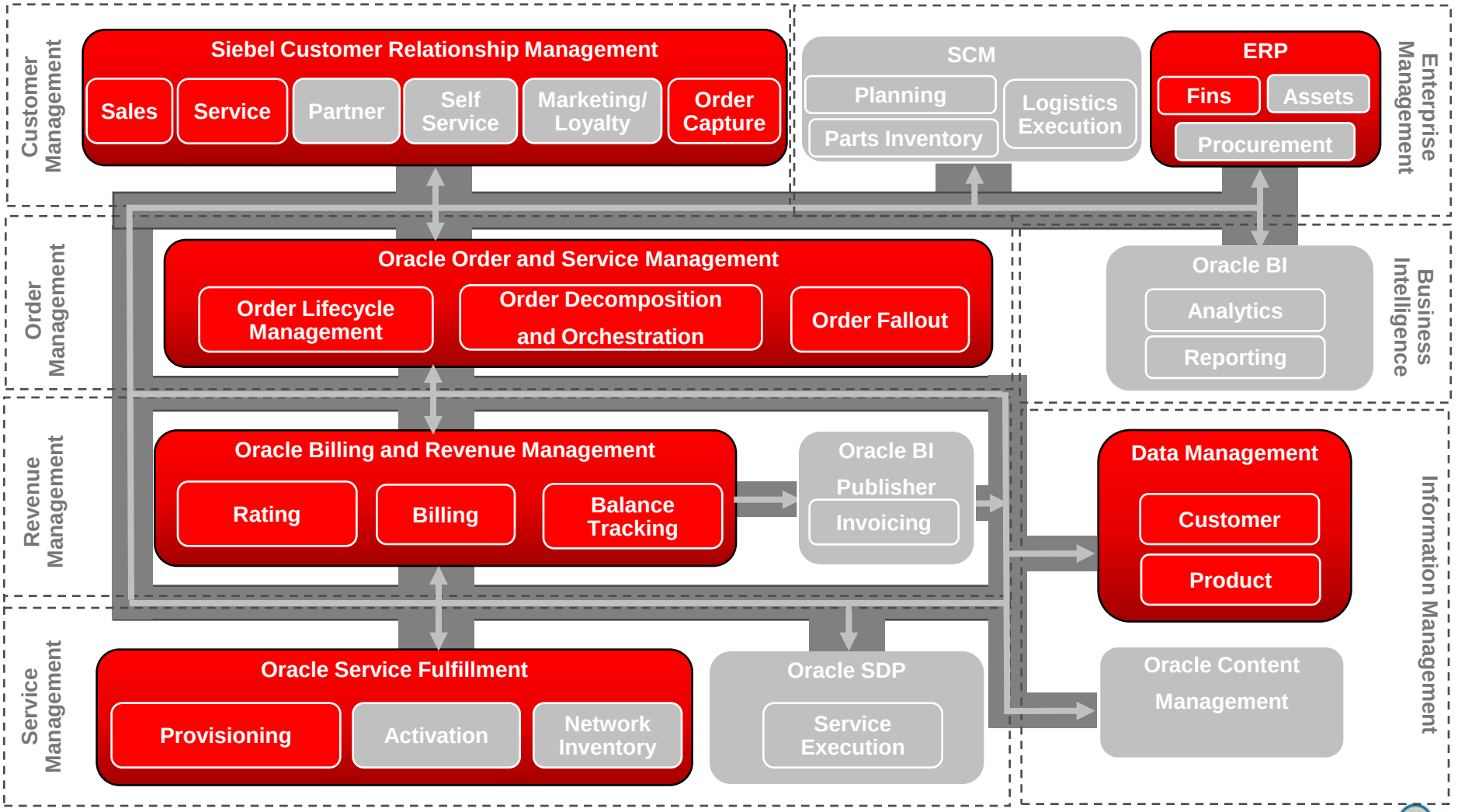
- By leveraging AIA, we allow for *loose coupling* of applications
- A change to any one application has a minimal impact on the integrations



Oracle Introduces AIA

- Everybody was building these integrations themselves
- AIA is not a new concept
- Oracle standardized these integrations and published a framework for integration development

AIA for Communications 2.5



Introducing Oracle AIA

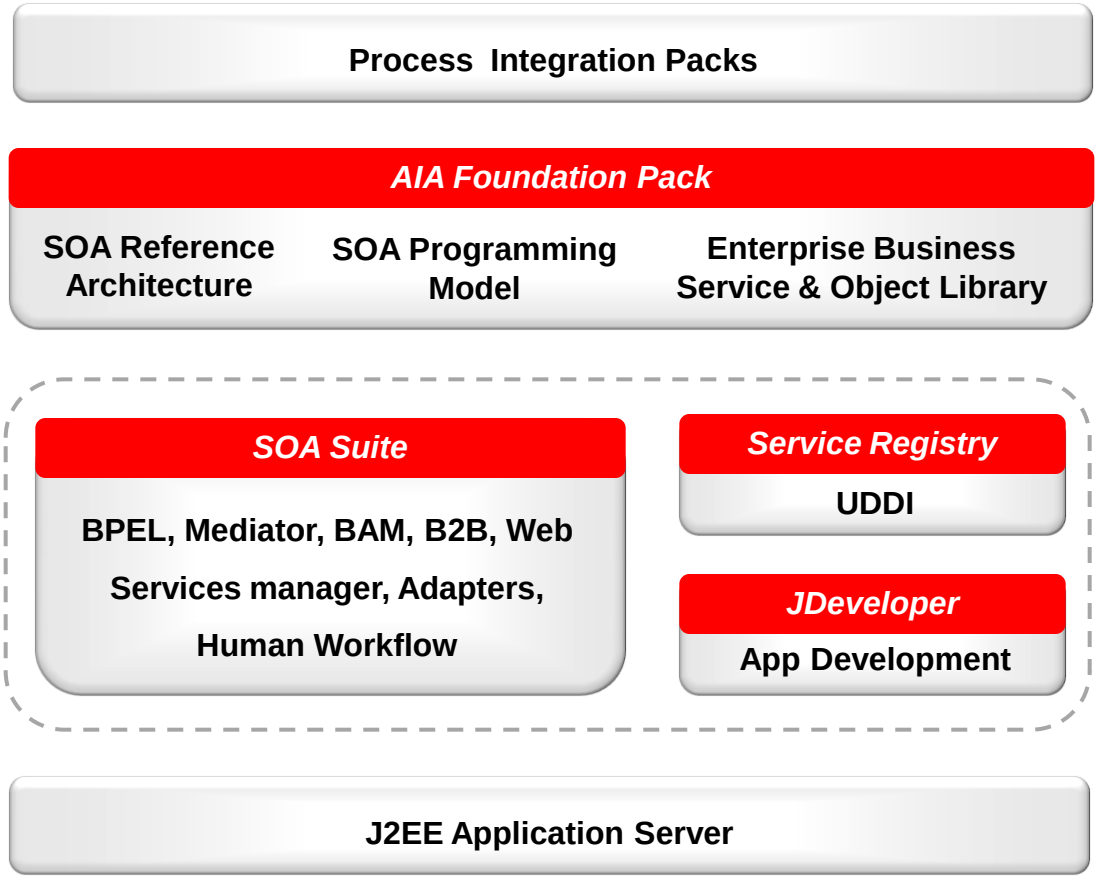
Oracle AIA Components

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

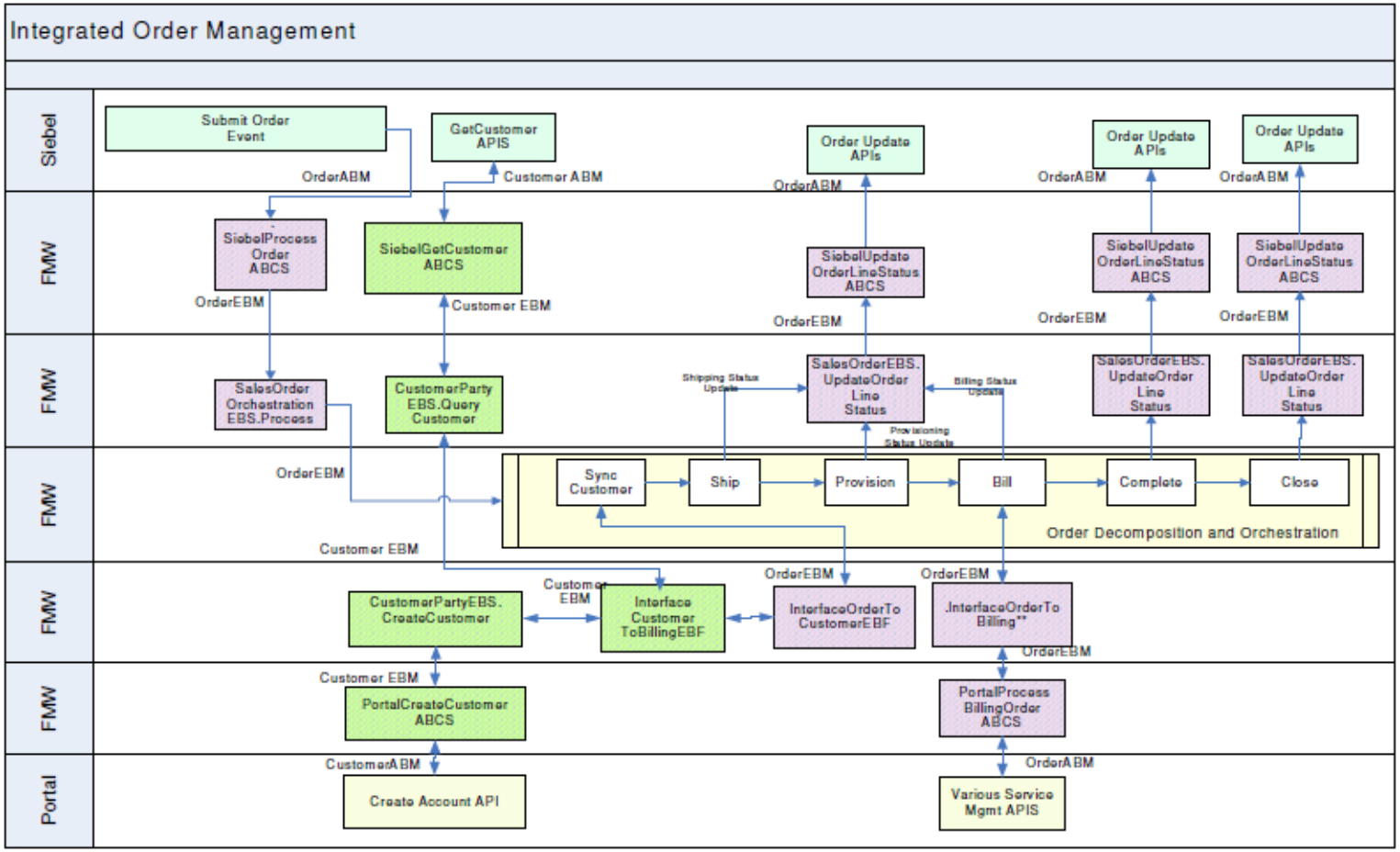
Process Integration Packs (PIPs)

- PIPs are a pre-built set of integrated orchestration flows, application integration logic, extensible EBOs and EBSs
- 9 cross-industry PIPs, 8 vertical PIPs
- Developing these from scratch is time consuming
- PIPs are supported, upgraded, and maintained by Oracle

Process Integration Packs (PIPs)



“Order to Bill” PIP



AIA for Communications 2.5 PIPs

Licensable Components	Overview
1. Order to Activate PIP *Siebel CRM to Oracle OSM*	• Automated product spec synch between CRM and OSM • Automates order fulfillment through central order management, service order management and billing
2. Order to Bill PIP *Siebel CRM to Oracle BRM*	• Automates data synchronization to ensure accuracy & real-time availability of customer, billing, product, and pricing data
3. Agent Assisted Billing Care PIP *Siebel CRM to Oracle BRM*	• Integrates the billing management process providing an integrated, real-time and actionable view of billing data from the CRM system
4. Revenue Accounting PIP *Oracle BRM to Oracle EBS*	• Provides a performance optimized General Ledger (GL) integration improving GL accounting, reporting and accuracy
5. Customer MDM PIP *UCM to multiple apps*	• Enables single source of truth in UCM and maintains consistent customer info in Siebel, EBS, BRM, and SAP
6. Product MDM PIP *PIM to multiple apps*	• Enables faster product launch by synchronizing product information from a central hub to Siebel, Oracle BRM, and E-Business Suite.

Enterprise Business Objects (EBOs)

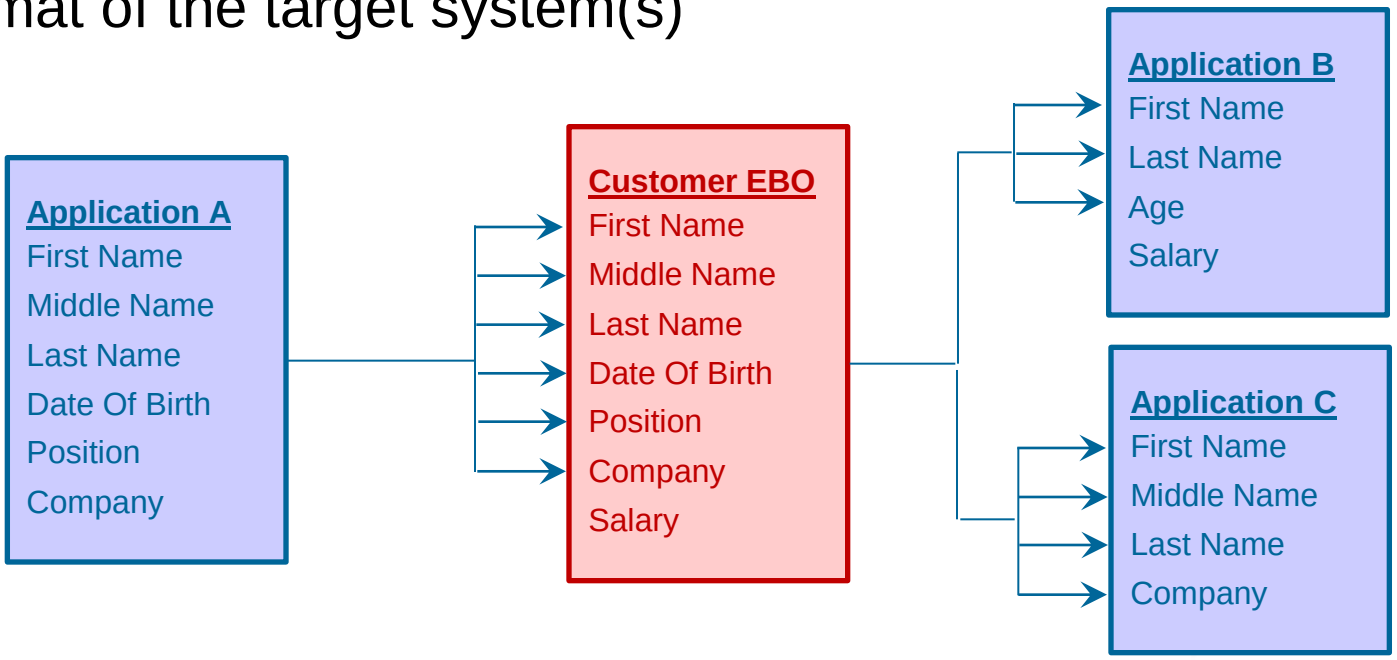
- EBOs are canonical data models
- Companies often spend many months just trying to agree on a corporate data standard to represent their key business objects
- EBOs are based on the open source OAGIS data model
- Designed with extensibility in mind
- Eliminates the need to comprehensively analyze your environment to determine a common message format

EBOs

- EBOs are XML schemas (i.e., xsd files)
- Examples:
  - BankAccount
  - CurrencyExchange
  - CustomerParty
  - Invoice
  - Item
  - Location
  - PurchaseOrder

Explaining the Canonical Data Model

- All applications are responsible for publishing as much information to the EBO as possible
- The integration will transform the data from the EBO to the format of the target system(s)



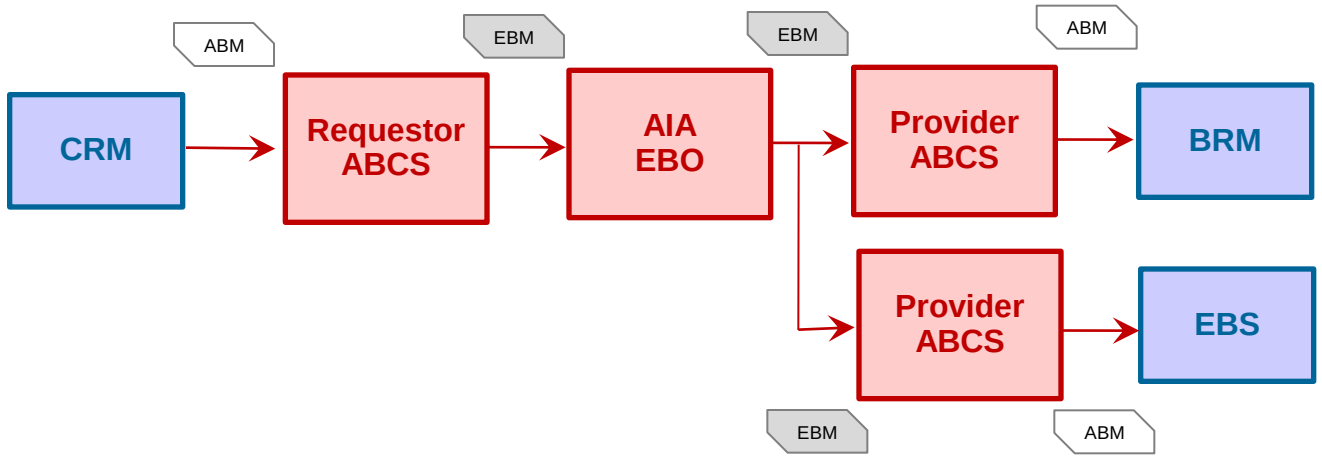
Canonical Data Model

- Below is an example of the BankAccount EBO
- It contains a superset of all account related fields

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.siebel.com/xml/CMU%20AccSync%20Account%20Io">
  <xsd:complexType name="ListOfCmuAccsyncAccountIo">
    <xsd:sequence>
      <xsd:element name="Account" type="xsdLocal:Account" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="Language" type="xsd:string"/>
    <xsd:attribute name="Locale" type="xsd:string"/>
    <xsd:attribute name="MessageId" type="xsd:string"/>
    <xsd:attribute name="EnterpriseServerName" type="xsd:string"/>
  </xsd:complexType>
  <xsd:complexType name="Account">
    <xsd:sequence>
      <xsd:element name="AccountStatus" type="xsdLocal:string24" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="AccountClass" type="xsdLocal:string30" minOccurs="1" maxOccurs="1"/>
      <xsd:element name="CurrencyCode" type="xsdLocal:string15" minOccurs="1" maxOccurs="1"/>
      <xsd:element name="MainPhoneNumber" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="AccountName" type="xsdLocal:string100" minOccurs="1" maxOccurs="1"/>
```

Development Methodology

- AIA proposes a methodology to follow when developing custom integrations
- Designed to *decouple* the source and target applications
- PIPs are developed using this methodology
- Updates to any application requires little to no changes to your integrations

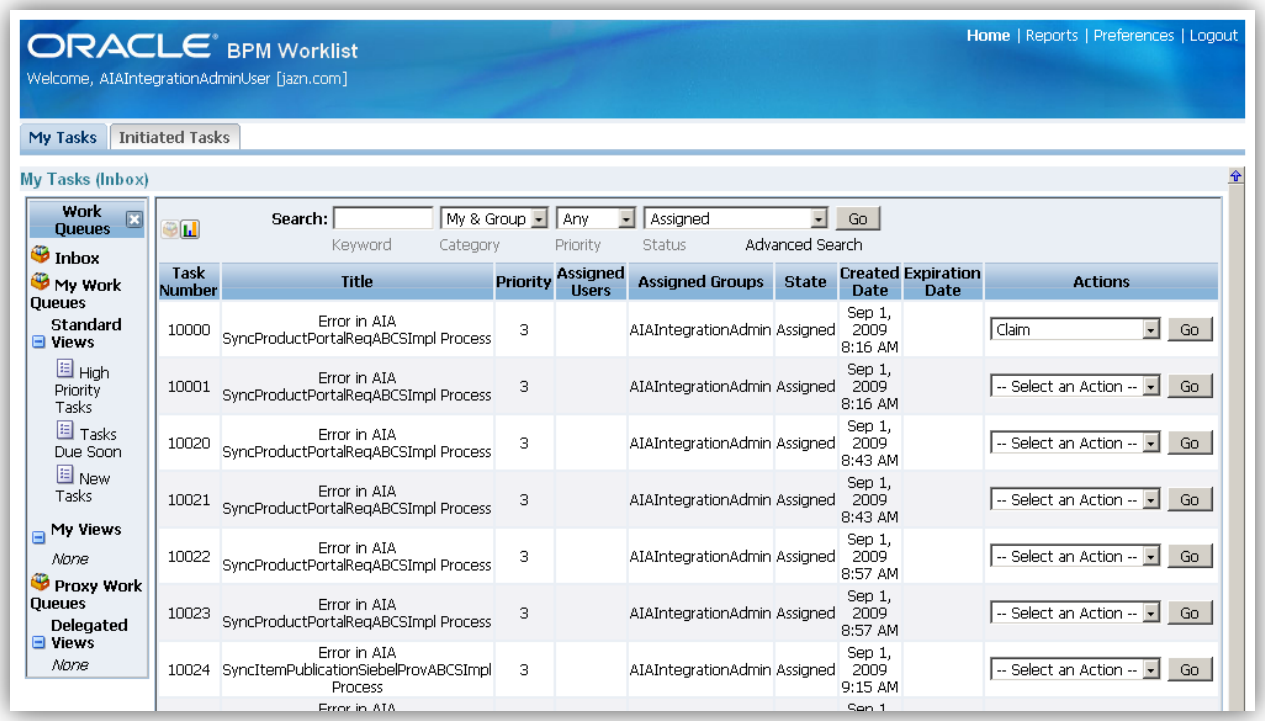


Error Handling Framework

- Captures errors
- Notifies on errors
- Workflow processes surround the error framework
- Can be used by all custom code
- Errors can be published to other systems or tools

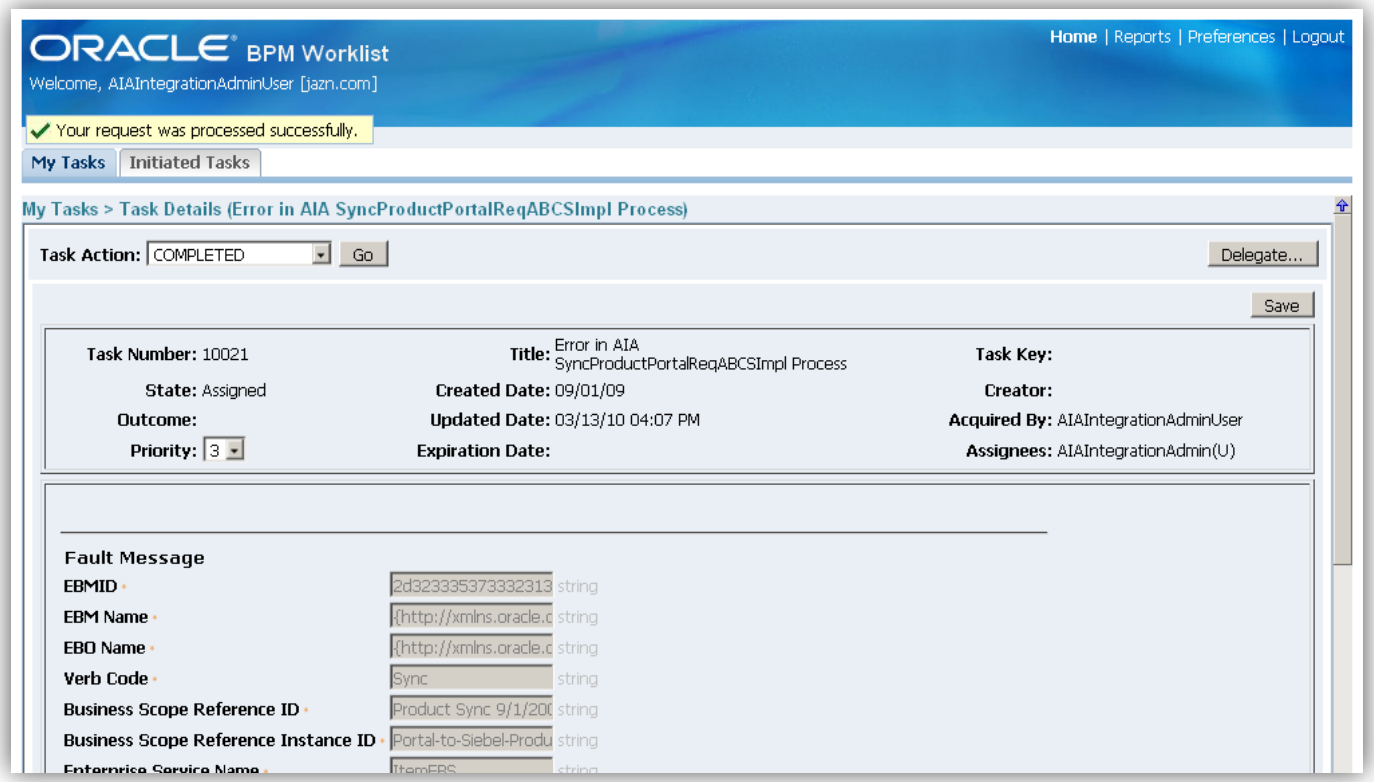
Error Handling Framework

- Errors can be *claimed*, *escalated*, and *delegated*



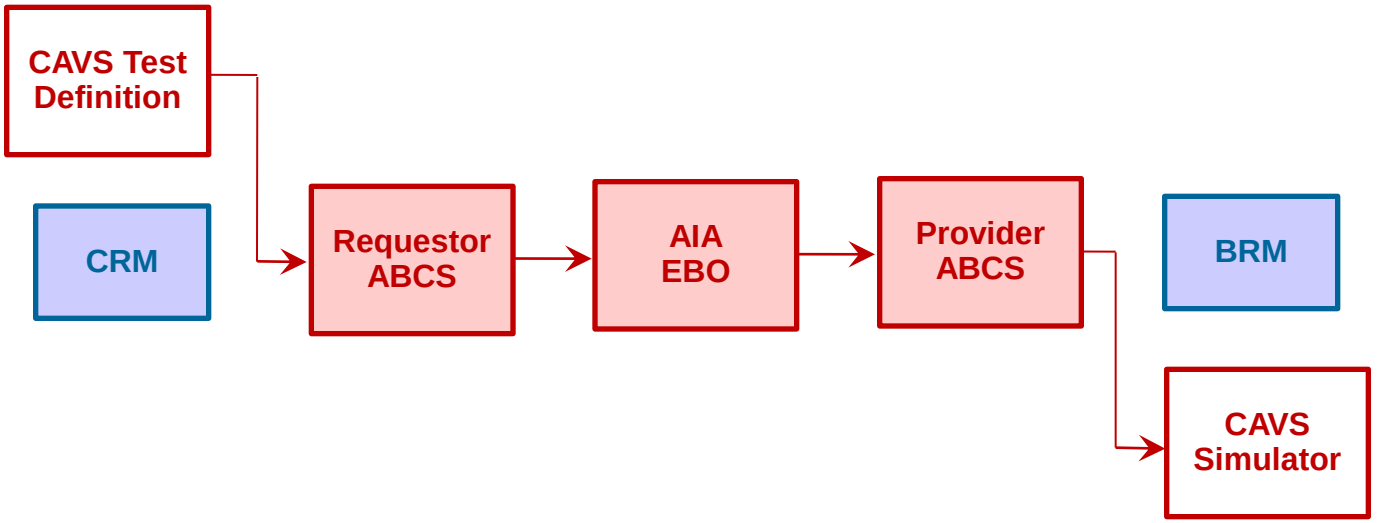
Error Handling Framework

- Error details are available to help in troubleshooting



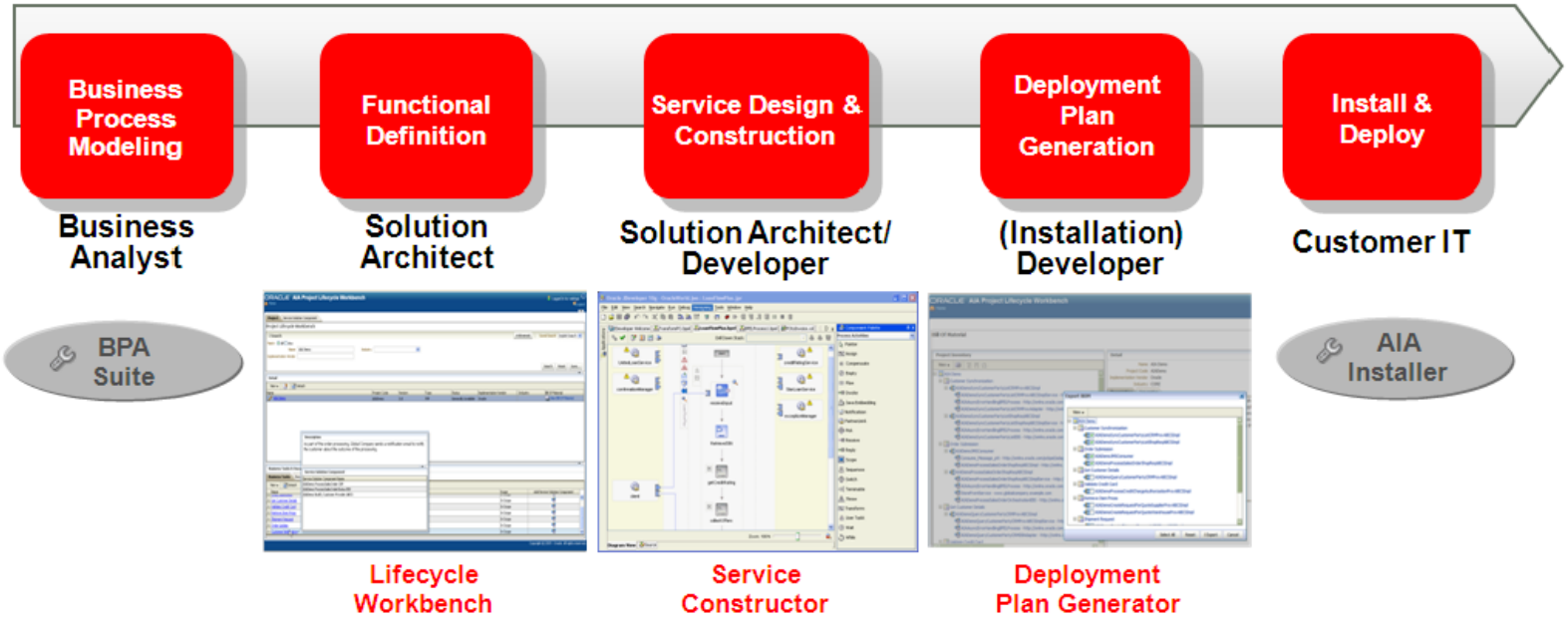
Composite Application Validation System

- Also known as CAVS
- CAVS is a framework to test integration of AIA services
- Provides test initiators that simulate web service
- Provides simulators that simulate service endpoints

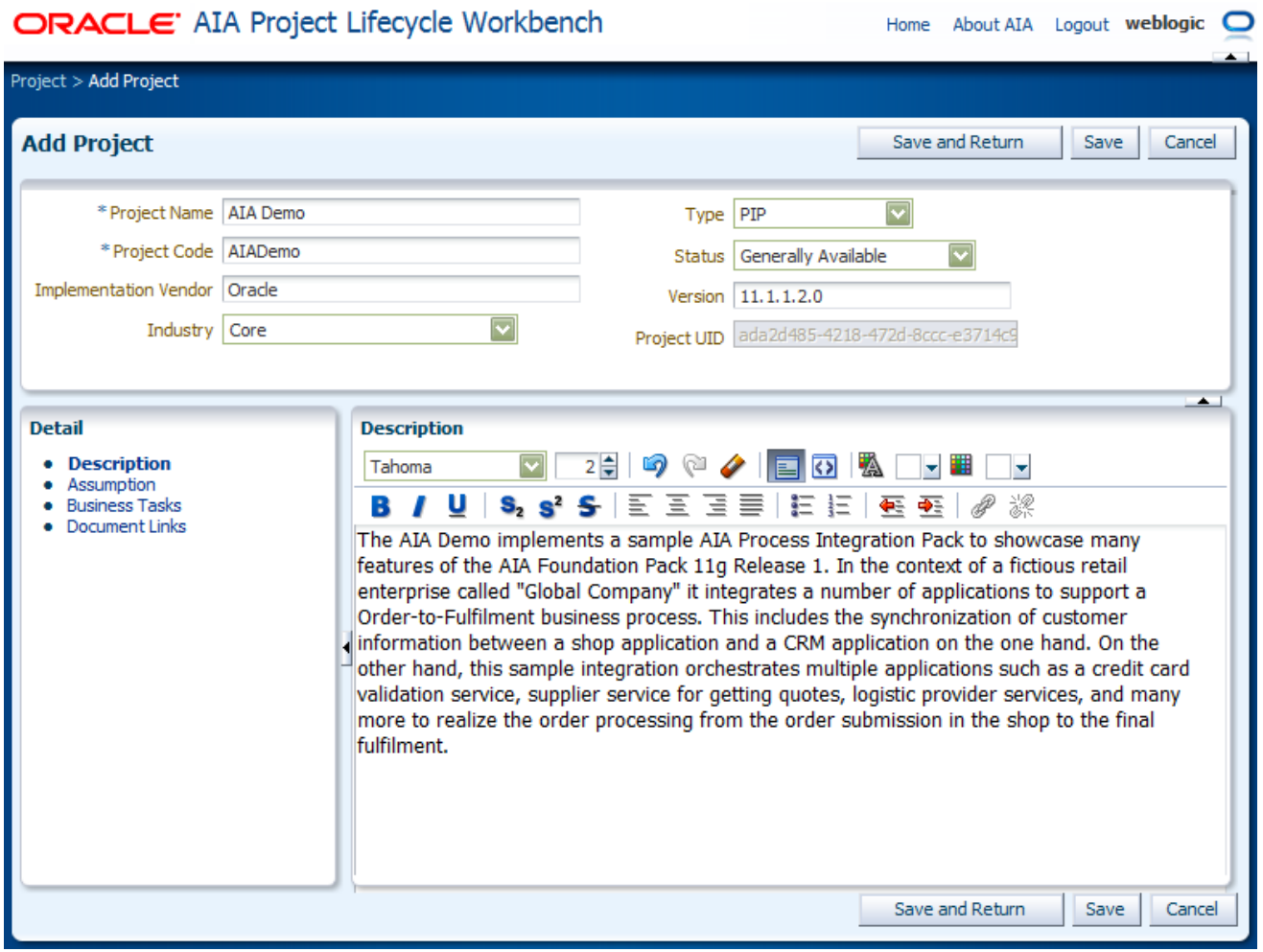


Project Lifecycle Workbench

- Functional designs are created to specify requirements that need to be implemented for an integration project
- Used to perform functional decompositions to break down overall projects into business tasks



Project Lifecycle Workbench



AIA Usage in the Real World

Customer A: Profile

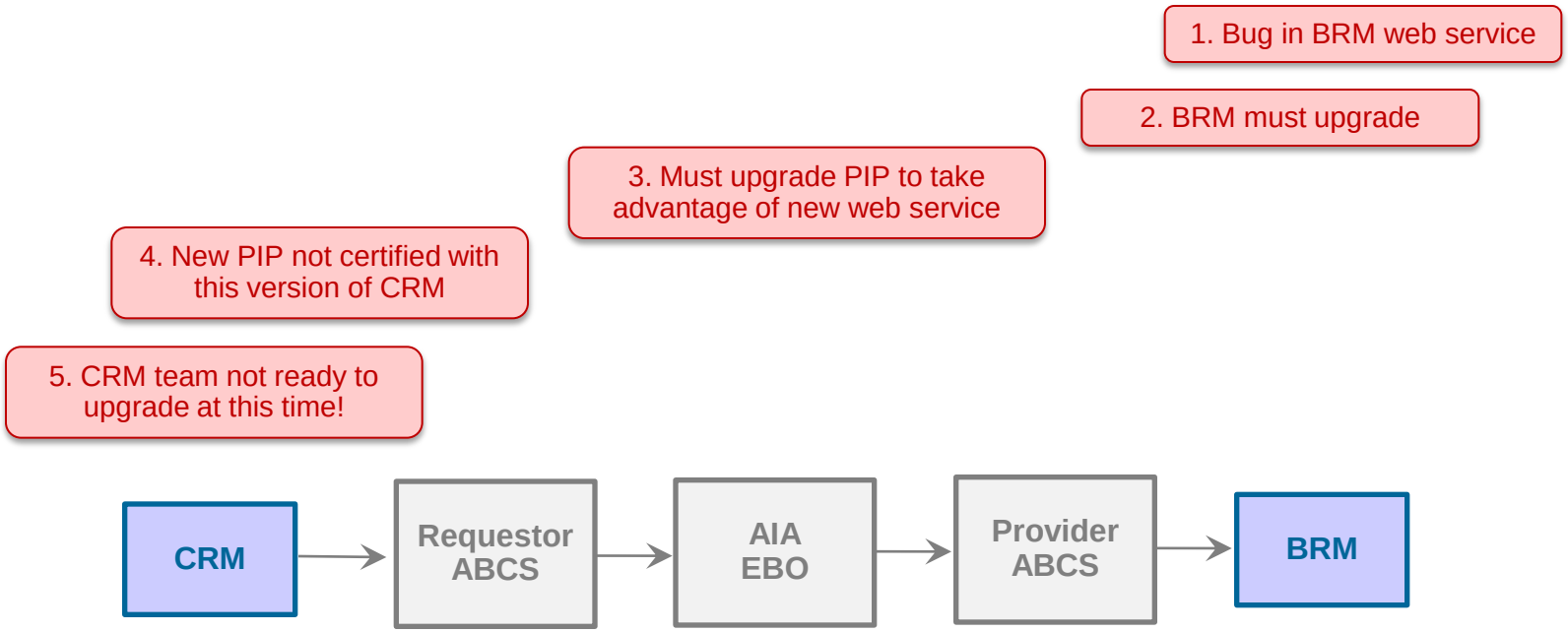
- Large telecom company
- Large number of enterprise applications:
  - 6 Oracle applications
  - 6 non-Oracle applications
  - 3 external applications
- Oracle AIA for Communications 2.0.1
  - 2 PIPs

Customer A: Reducing Initial Cost with PIPs

- 2 PIPs were used
- Massive reduction in time & cost for the initial integration development effort

Customer A: The Challenge of PIPs

- Not a very common problem
- Encountered one instance of PIPs dependency issues



Customer A: AIA Features Not Leveraged

- CAVS not used
  - Would not have impacted testing approach
  - All code must pass functional testing anyway
- Error Handling Framework not used
  - Only aware of errors when customer reports it
- IAIADiagnostics interface was proposed, not used
  - Customer followed Solution Architect's direction of using this core AIA feature, instead of a simpler custom solution
- Lack of experience, comfort, and familiarity with AIA causes teams to not use these features

Customer B: Profile

- Global retail company
- Very large applications supporting global operations:
  - 1 Oracle application (4 more in the future)
  - 6 custom applications (50+ more in the future)
- 600+ SOA projects
- Oracle AIA Foundation Pack 11g

Customer B: Cumbersome AIA Methodology

- A single integration consists of 5 projects, sometimes resulting in high overhead under heavy load and complicating the development process
- Did not firmly adhere to AIA methodology
- Initial design mistakes were made as a result of learning curve, which introduces risk in a new implementation when trying to set the foundation

Customer B: Disappointment in CAVS

- CAVS was perceived to be great in the beginning
- Oracle does not recommend using CAVS in production, so validation of updated services in production not possible
- CAVS did not provided much value over time:
  - During development, the preferred route proved to be always hitting the target web service, not simulator
  - SoapUI is a more convenient client testing tool for developers

Customer B: SOA Suite 11g Upgrade Painful

- Major rewrites required in custom developed PIPs when upgrading from SOA Suite 10g to 11g
- Many 10g features/capabilities have been made obsolete
- This would not be a problem with Oracle developed PIPs
- No issues with core AIA Foundation Pack features (EBOs, error handling) when upgrading from 2.x to 11g

Customer B: Long-Term ROI Evident

- Reuse of integrations became evident:
  - As the number of source applications increased
  - Due to the loose coupling recommended by AIA
  - After 2 years
- Standardizing on AIA EBOs eliminated early analysis efforts
- Although understanding EBO elements in the beginning took some time, is no longer an issue, especially after developing an internal “canonical mapping list”

Summary

Truth to Oracle's Claims

Based on survey of 4 leading Oracle AIA Solution Architects

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%

Summary

- AIA can be a solution to the fundamental challenge of application integration
- For smaller implementations, AIA is not recommended
- For larger implementations, expect to initially see cost reduction when using PIPS
- For large implementation that do not leverage PIPs, expect high initial cost and risk, but long term ROI afterwards
- PIPs help reduce risk
- PIPs are not available for all types of integrations; custom development and/or customization is almost always required
- Not enough expertise in the marketplace
- Skilled AIA Solution Architects are hard to find

Contact Information

Apurva Singh

Technical Director

apurva.singh@ipnweb.com

Ahmed Aboulnaga

Technical Director

ahmed.aboulnaga@ipnweb.com