SOA-basierte Integration mit AIA im praktischen Einsatz

Tjark Bikker
PROMATIS software GmbH
Hamburg

Schlüsselworte:

AIA, SOA, Erfahrungsbericht.

Einleitung

Dieser Beitrag berichtet von den verschiedenen Erfahrungen, die während des praktischen Einsatzes des Application Integration Architecture Frameworks in der Projektarbeit beim Kunden gesammelt wurden.

Ein Aspekt des Beitrags behandelt dabei sowohl die Einschränkungen, denen das Framework zum Zeitpunkt des Einsatzes aufgrund der Implementierung unterworfen war, als auch die Möglichkeiten, diese Beschränkungen zu umgehen.

Anhand einer für das Horus Werkzeug implementierten Lösung wird gezeigt, wie die Funktionalitäten des Frameworks automatisiert genutzt werden kann, um massive Zeitersparnis zu erzielen und um einige Möglichkeiten für durch Fehlbedienung generierte Fehler zu minimieren.

Anschließend wird gezeigt, wie die Design-Time- und die Development-Time-Informationen automatisiert ausgewertet werden können, um jeder Zeit aktuelle Dokumentationen zur Entwicklung erstellen zu können.

Das Projekt

Zur Evaluation des AIA-Frameworks wurde in Zusammenarbeit mit einem Kunden ein Projekt durchgeführt, in dem verschiedene heterogener Systeme miteinander verbunden werden sollten.
Die einzelnen Systeme bestehen dabei aus verschiedenen Kombinationen aus Betriebssystemen und Datenbanken.

Das Szenario

Der Business Case, der im Projekt identifiziert wurde und in diesem Beitrag vorgestellt werden soll, betrifft die Synchronisation von Warenbewegungen. Warenbewegungen können in diesem Beispiel in verschiedenen Systemen erfasst werden~~,~~ und müssen an die anderen Systeme verbreitet werden.
Das beschriebene Szenario sollte so umgesetzt werden, dass die Nachrichten nach dem Fire-And-Forget-Muster verteilt werden. Dies bedeutet, dass eine Nachricht über eine Warenbewegung, die von System A erfasst wird, an die Oracle SOA Suite übergeben werden soll, und die SOA Suite ist verantwortlich für die Zustellung der Nachricht. Falls eine Nachricht an ein bestimmtes System nicht weitergeleitet werden kann, soll die Zustellung von der SOA Suite automatisch wiederholt werden.
Die Zustellungsschwierigkeiten sind für das initialisierende System nicht relevant.

Da bisher für keines der Systeme eine XML-Struktur definiert worden war, sollten die Application Business Messages grundsätzlich den Formaten entsprechen, die von den jeweiligen SOA-Adaptern bei der Implementierung generiert werden.

Design

Als ersten Schritt wurde die Struktur der Integrationslösung definiert. Nach der AIA-Systemarchitektur wurde für jedes Enterprise Business Object ein Projekt in der AIA Lifecycle Workbench definiert, das je nach Anforderung eine oder mehrere Business Tasks beinhaltet.
Das in diesem Beitrag vorgestellte Projekt beinhaltete zum Beispiel zwei Business Tasks: eine für den Wareneingang und eine für den Warenausgang. Der jeweilige Nachrichtenverlauf ist in den Abbildungen 1 und 2 zu sehen.

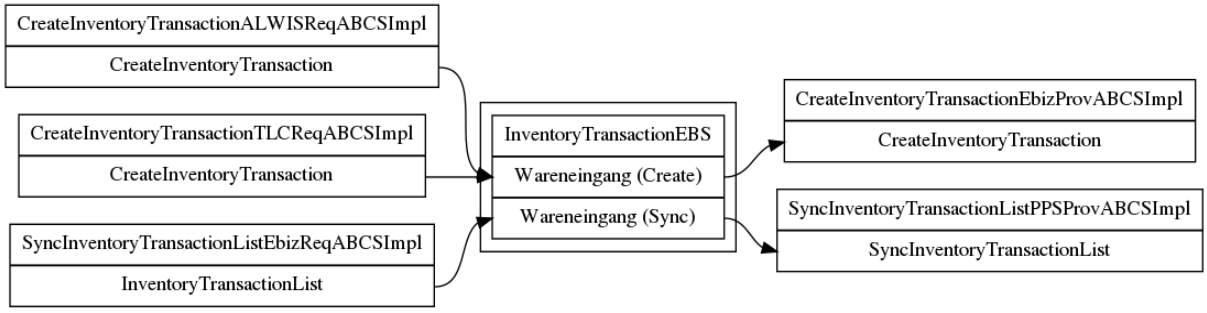

Abbildung 1: Nachrichtenverlauf beim Wareneingang

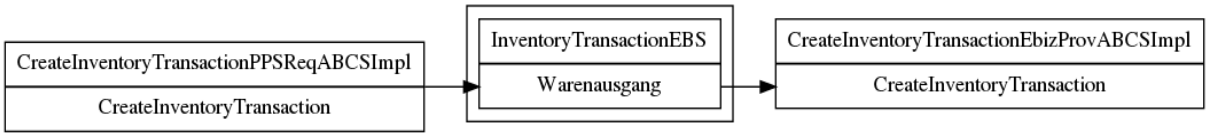

Abbildung 2: Nachrichtenverlauf beim Warenausgang

Nachdem die grundsätzliche Systemarchitektur definiert wurde, wurden die Mappings von den Application Business Objects (ABO) auf die Enterprise Business Objects des AIA-Frameworks definiert. Dazu wurden zunächst die zu übertragenden Daten und die zugehörigen Felder in den Enterprise Business Objects identifiziert. Zusammen mit der Identifikation der jeweiligen Quellen und Ziele für diese Daten wurde in diesem Schritt auch gleich das Mapping der Enterprise Business Objects auf die Application Business Objects definiert.

Nachdem die bisherigen Daten in die Lifecycle Workbench eingetragen worden sind, ist der Design-Abschnitt abgeschlossen, und die Entwickler können mit der Umsetzung der Definitionen beginnen.

Implementierung

Dieser Abschnitt beschreibt zunächst das Vorgehen bei der Implementierung und stellt im Anschluss die Erfahrungen dar.

Vorgehen

Da zu Beginn keine Application Business Objects existierten, mussten diese zunächst definiert werden.

Um die jeweiligen Systeme anzubinden, wurde zunächst für die beteiligten Systeme Transportadapter definiert. Hierbei wurden verschiedene SOA-Adapter verwendet. Aus performance-Gründen wurden, wo immer möglich, Transportadapter erstellt, die einen Webservice zur Verfügung stellen, und von den jeweiligen Systemen aufgerufen werden. Bei einigen Systemen war das so jedoch nicht möglich, beziehungsweise nicht erwünscht. Bei diesen Systemen wurden dann beispielsweise sogenannte „pollende Transportadapter“ erstellt, die alle n Minuten auf neue Datensätze überprüfen. Durch diese Transportadapter wurde automatisch eine XML-Struktur definiert, die gleich in das MDS hochgeladen und als Application Business Objects verwendet wurden.
Mit dem JDeveloper-Plugin des AIA-Frameworks kann mithilfe dieser Definitionen der ABOs für die Requestor- und Provider-Komponenten der Business-Tasks der AIA-Architektur entsprechende Artefakte generiert werden. Die Erstellung dieser Komponenten wird durch einen Wizard getrieben, bei dem mit wenigen Eingaben nahezu fertige Komponenten erstellt werden. Damit die jeweiligen Komponenten auch die Daten übertragen, müssen nur noch die Transformationen von den Application Business Messages auf die Enterprise Business Messages erstellt werden.

Als nächster Schritt wurde der Enterprise Business Service mithilfe eines einfachen Mediators definiert, der den Transport von Requestoren zu Providern verwaltet. Nachdem alle Komponenten miteinander verknüpft wurden, wurden erste Funktionstests durchgeführt um die Implementierung zu testen. Für diese Tests wurden die Komponenten mit dem JDeveloper auf der Entwicklungsumgebung deployt, und Nachrichtenübertragungen angestoßen.
Abschliessend wurden die für die weitere Verarbeitung mit dem AIA-Framework notwendigen Annotationen in den Komponenten eingetragen, und die fertiggestellten Komponenten mit dem zugehörigen Tool des AIA-Frameworks in der Lifecycle Workbench registriert.

Rückblick

Der Arbeitsaufwand bei der Implementierung hält sich durch das JDeveloper-Plugin sehr in Grenzen, da funktionsfähige Komponenten erstellt werden, bei denen nur noch kleine Anpassungen notwendig sind. Die zur Projektzeit zur Verfügung stehende Version des Service Constructors hatte leider ein paar kleinere Probleme, vor allem bei der Arbeit mit dem MDS und mit dem Übernehmen von Informationen beim Verwenden der „Vor“- und „Zurück“-Schalter. Der Service Constructor hat auch einige Einstellungen erlaubt, die nach Abschluss des Wizards zur Erstellung von unvollständigen oder leeren Komponenten geführt hat.
Der im JDeveloper integrierte XSL-Editor leistet gute Dienste, auch wenn er aufgrund der Komplexität der Enterprise Business Objects nahe an seine Leistungsgrenzen zu kommen scheint.
Bei der Implementierung ist aufgefallen, dass die einfache Erstellung der Komponenten dazu verleitet, Entwickler einzusetzen, die mit den zugrundeliegenden Techniken (XML, XSL) nicht oder kaum vertraut sind. Vor dem Einlesen der Komponenten ist zusätzlich aufgefallen, dass die Annotation überhaupt nicht durch Werkzeuge unterstützt wird und daher sehr mühsam ist.

Deployment

Das AIA-Framework soll es mit dem AIA-Installer ermöglichen, Projekte aus der Lifecycle-Workbench zu extrahieren, und mithilfe der dort eingelesenen Implementierungsdetails eine automatische Installation auf einem beliebigen SOA Suite-Server vorzunehmen.
Dabei sollen dann beispielsweise alle zugehörigen Datenquellen eingerichtet und die implementierten Komponenten deployt werden.
In diesem speziellen Projekt hat der Kunde, um die Qualität des Frameworks besser beurteilen zu können, Wert darauf gelegt, dass diese frische Installation reibungslos funktioniert. In einem einfachen

Testlauf, der eine einfache Requestor-EBS-Provider-Struktur zwischen zwei Oracle-Datenbanken aufspannen sollte, hat der AIA-Installer wunderbar funktioniert.

Wie Eingangs beschrieben bestand die Systemlandschaft beim Kunden allerdings aus einem sehr heterogenen Gemisch aus Betriebssystemen und Datenbanken. Hierbei hat sich gezeigt, dass der Installer nicht wirklich auf einem Stand war, der die gewünschte reibungslose frische Installation gewährleistet.
Vor allem die Erstellung von Datenquellen war sehr unbefriedigend, da der Installer lediglich Datenquellen mit dem Oracle-Treiber erstellen konnte, und auch nicht berücksichtigte hat, ob die Datenbank Transaktionen unterstützt wird oder nicht.
Leider war in dem Einsatzszenario lediglich die Datenbank, auf der die E-Business Suite lief, eine Oracle-Datenbank und eine Datenbank hat keinerlei Transaktionen unterstützt.
Um die neue Installation zu ermöglichen mussten einige Quellen des AIA-Installers angepasst werden.

Integration in Horus

Am Beispiel des Horus Werkzeugs wird gezeigt, wie das Design und die Implementierung durch geeignete Schnittstellen in ein prozessorientiertes Werkzeug eingebunden werden kann.

Automatische Dokumentation

Die Verknüpfung von Design-Informationen in der Lifecycle-Workbench und Implementierungsdetails durch die Annotationen und das Einlesen der Details in die Lifecycle-Workbench ermöglichen das automatische Erstellen von technischen Dokumentationen, die geeignet sind, um einen Überblick über die Implementierungen und den Stand der Entwicklung zu ermöglichen. Anhand eines Beispiels wird eine solche Erstellung vorgestellt.

Zusammenfassung

Als Vorteil des AIA-Frameworks ist anzuführen, dass durch den Einsatz der Werkzeuge die erstellte Integration einer Architektur entspricht, die durch die Entkopplung der einzelnen Applikationen größtmögliche Flexibilität garantiert. Auch der durch diese Architektur bedingte Entwicklungs-Overhead wird durch die bereitgestellten Werkzeuge sehr stark eingeschränkt. Diese Werkzeuge können den Entwickler soweit unterstützen, dass einfache Integrationen mit mehreren Systemen schneller entwickelt sind als entsprechende minimalistische Custom Developments.
Leider waren einige Werkzeuge noch nicht ausgereift, so dass je nach Anforderung die Basisfunktionalität erweitert werden musste.

Kontaktadresse:

Tjark Bikker
PROMATIS software GmbH
Notkestraße 9
D-22607 Hamburg

Telefon: +49 (0) 40-2533 269 0
Fax: +49 (0) 40-2533 269 99
E-Mail tjark.bikker@promatis.de
Internet: www.promatis.de