OSB 12 general concepts and usecases

Cosmin Giru
virtual7 GmbH
Karlsruhe

Schlüselwörter:
OSB12, SOA, Architektur, Best Practices

Einleitung

Inhalte der Präsentation:

- Allgemeine Informationen: Was ist der OSB und wofür wird er verwendet
- Die einzelnen Bausteine eines Flows und wie die Adapter eingesetzt werden. Dies beinhaltet auch einen Vergleich zwischen Service Implementierung mit Composite und BPEL und Proxy Service mit Pipe Lines
- Verbesserungen von Version 11 zu Version 12, Performance- und Entwicklungs-Tools. Was ist besser in JDeveloper im Vergleich zu Eclipse OEPE? Was sind die Limitierungen von "sbconsole"?
- Richtlinien: Wann muss der OSB verwendet und wann müssen andere SOA oder andere Oracle Produkte eingesetzt werden

OSB Bausteine

Die einzelnen Bausteine des OSB können einfach in drei Schichten eingeteilt werden: Eingang, Verarbeitung und Ausgang.

Einfach gesagt: ein Flow wird getriggert, hat eine gewisse Logik und einen Output.
Eingang und Ausgang sind eigentlich SOA Adapter Wrappers, die durch ein spezifisches Verfahren getriggert werden oder selbst triggern können.
Wenn man einen Service betrachtet, der im OSB implementiert wurde im Vergleich zu einem Service, der mit Composite implementiert wurde, kann man im OSB 12 sehr viele Ähnlichkeiten entdecken. Die graphische Darstellung des JDevelopers von Composite und Pipelines Pairs zeigt, dass sich bei beiden die Logik in der Mitte befindet. Dort wo Daten durch verschiedene Umwandlungen und der Entwicklung von Sprachen, welche die Input-Informationen verändern, verarbeitet werden. Die Partner Links im Composite, die links und rechts positioniert sind, können mit dem Proxy und Business Services im OSB verglichen werden. Beide verwenden SOA Adapters um verschiedene Endpunkte aufzurufen oder zur Verfügung zu stellen.
Als Sprachen, die den Input in Output verwandeln, kann man beliebig Xpath, XQuery, XSLT Transformationen, Java und auch JavaScript benutzen.

Von OSB 11 zu 12

Warum sprechen wir noch über OSB 11? Ganz einfach, denn es gibt da draußen sehr viele Projekte die es noch benutzen! OSB vereinfacht die Kommunikation zwischen Applikationen oder Systemen.

Was ist besser und neu im OSB12?

Zu einem kann jetzt nur noch JDeveloper für das Development verwendet werden. Das heißt keine Konfiguration mehr von Adaptern und Transformationen im JDeveloper und anschließende Implementierung des Proxy und der Business Services in Eclipse.
Im Zusammenhang mit dem JDeveloper muss man auch über die Performance sprechen. Für Projekte die Proxy Services mit mehr als 5000 Linien in XML im Einsatz haben, ist Eclipse mit OEPE überfordert, der JDeveloper nicht. Hier ist man gezwungen die Web Konsole von OSB zu verwenden, wo ein anderes Problem auftritt. Die Herausforderung ist die Ladezeit der Logik im Stage. Wenn wir mit der Konsole eine Stage bearbeiten, generiert sie eigentlich eine Form mit Inputs, welche die Logik im Flow beschreibt. Um diese Form zu generieren dauert es eine Weile, wenn sie zu groß ist und nicht alle Felder generiert wurden als man eine Änderung gemacht hat, verliert man die Informationen von Stage.

Alle die mit OSB 11 bereits gearbeitet und auch Applikationen gebaut haben, bekommen dieses Gefühl, wenn man etwas in Java machen wollte. Die Statischen Klassen! Als Java Entwickler möchte man Objekt-orientiert Programmieren, warum dann aber nur statische Klassen im OSB benutzen? Die Antwort ist einfach, denn es sollten hier nur gezielte Tasks in Java und keine kompletten Applikationen implementiert werden. Wenn so etwas benötigt wird, kann jederzeit ein Service gebaut, ein Deployment auf dem Server gemacht und durch einen Business Service aufgerufen werden.
In Version 12 kann hier JavaScript verwendet werden. Dann gibt es keine neuen Jars zu deployen, statische Java Klassen zu implementieren und sie im OSB aufzurufen.

Pipelines sind in Version 12 getrennt von Proxy Services. Dies verbessert dir Architektur. Dadurch ist die eigentliche Umwandlung oder Logik von Eingangsdefinitionen des Service getrennt. Auf diese Weise kann das Projekt besser organisiert und mit einer Versionierung das System besser verwaltet werden. Eine Pipeline-Definition kann jetzt auch von mehreren Proxy Services verwendet werden.

Man kann eine Pipeline als REST-Dienst aussetzen. Dabei wird eine REST-Bindung erstellt, wie sie vielleicht schon bei einem SCA Composite bekannt ist. Der Nachteil hierbei ist, dass man nur Pipelines ausstellen kann, die eine WSDL-Schnittstelle haben.

Ein Re-sequencing in OSB kommt auch mit dem OSB 12. Mediator hatte dies in SOA schon seit Version 11, aber jetzt kann es auch im OSB benutzt werden. Der Re-Sequenzer im Oracle Service Bus erstellt Nachrichten auf der Grundlage von sequentiellen oder chronologischen Informationen und sendet anschließend die Nachrichten ordnungsgemäß an die Zieldienste. Die Sequenzierung erfolgt auf Basis der ausgewählten Sequenzierungsstrategie.

Service Bus 12 unterstützt auch die Verwendung von Vorlagen. Dies funktioniert jedoch ein wenig anders als gewohnt. Vorlagen sind Pipeline-basiert, deshalb kann man beim Erstellen einer neuen Pipeline eine Vorlage auswählen. Es gibt zwei Arten von Vorlagen; Unlinked, dies ist eine Kopie der Vorlage, und Linked, die Pipeline bleibt mit der Vorlage verbunden. In der Vorlage können Platzhalter verwendet werden, um Änderungen an diesem Teil der

Pipeline zuzulassen. Wenn die Pipeline mit einer Vorlage verknüpft ist und die Vorlage geändert wird, dann übernimmt die verknüpfte Pipeline diese Änderungen.

Wie einfach ist es ein Projekt von OSB 11 nach 12 zu migrieren?
Mit dem JDeveloper kann man ein exportiertes Project Jar importieren und auch in Version 12 migrieren. Jetzt stellt sich jedoch die Frage, ob es etwas gibt, dass JDeveloper nicht migrieren kann. Wie immer bei einer Migration gibt es mehrere Probleme, die auftreten können. Diese sollten aber nicht vorkommen, wenn der OSB richtig angewendet wurde. Dies wirft gleich eine andere Frage auf: Wie und wann sollte man OSB verwenden?

Wie und wann sollte man OSB verwenden

Man sollte nicht versuchen die gesamte Lösung nur auf OSB zu entwickeln: der OSB ist keine alleinstehende Lösung, er ist Teil der Oracle SOA Suite. In einem Projekt kann der Grundfehler gemacht werden, dass die gesamte Lösung auf OSB entwickelt wird.

Statefull Transaktionen sollten innerhalb des OSB nicht vorhanden sein. Der OSB ist ein state-lose Produkt. Es ist nicht angedacht darin State-Full-Transaktionen zu behandeln. Für die State-Full-Transaktionen sollte man BPEL oder BPM anstelle von OSB verwenden.

Verwenden Sie nicht OSB, um komplexe SQL-, SQL-Prozeduren auszuführen. Obwohl der OSB Mechanismen zur Ausführung von SQL- und SQL-Prozeduren in seiner Pipeline bereitstellt, müssen diese jedoch für den Abruf (Select) -Operation eingeschränkt werden.

Verwenden Sie keine Java-Callouts, um Ihren geschäftlichen Anforderungen gerecht zu werden: Java-Callout ist die am meist missbrauchte Funktion von OSB. Die Business-Logik, die nicht durch den OSB erreicht wird, macht man mit einer statischen Methode, die durch Java-Callout aufgerufen wird. Wir erhalten dann eine fixe und unflexible Funktionalität, die schwer zu ändern und zu verwalten ist.

Komplexe Geschäftslogik innerhalb der OSB Message Pipeline ist falsch. Definieren Sie die Geschäftsregeln in Oracle Business Rules, um sie als Web-Service auszusetzen und Ihren Geschäftsregel-Webdienst in OSB zu verwenden.

WS Policy Security sollte man nicht für die Security benutzen. Um die Sicherheit des OSB-Diensts zu implementieren, verwenden Sie OWSM (Oracle Web Service Manager).

Nicht den gesamten Nachrichtenfluss mit einer einzigen Stage entwickeln. Jede Stage sollte nur relevante Handlungen haben und der Stagename sollte eindeutig erklären, was diese Stage eigentlich macht.

Es sollte keine große Datei durch OSB verarbeitet werden. OSB ist nicht für schweres Heben gedacht; man sollte ODI(Oracle Data Integrator) für die Verarbeitung großer Dateien verwenden.

Wenn möglich verwenden Sie keine Wildcard-Xpaths. Vermeiden Sie die Verwendung von

Wild Card Zeichen * oder rekursiven xpath Pfad Operator "//" in Xpath Ausdrücken. Diese verursachen erhebliche Leistungsaufwände, vermeiden Sie so viele wie möglich.

Kontaktadresse:

Cosmin Giru
virtual7 GmbH
Zeppelinstraße 2
76185 Karlsruhe

Telefon: +49(0)17-664711875
E-Mail cosmin.giru@virtual7.ro
Internet: www.virtual7.de