Wrapper und Konnektoren – geht die Rechnung auf?

Klaudia Hergula

DaimlerChrysler AG
Forschung und Technologie
Wissensaustausch / Austauschgruppe (FTK/A)
HPC 0516, Epplestr. 225, D-70546 Stuttgart
klaudia.hergula@DaimlerChrysler.com

Kurzfassung: In diesem Aufsatz wird die Einsatzfähigkeit der Wrapper- und Konnektoren-Ansätze zur Daten- und Funktionsintegration untersucht. Hierzu werden Anforderungen an eine solche Integrationslösung gestellt und aufgezeigt, wie sie mit Hilfe der beiden Konzepte realisiert werden kann. Darüber hinaus wird die Kombination von Wrappern und Konnektoren vorgeschlagen, um die Stärken beider Ansätze nutzen zu können.

1 Motivation

In den letzten Jahren haben sich zwei grundsätzliche Formen der Integration von Informationssystemen entwickelt: die Daten- und die Anwendungsintegration. Während die Thematik der Datenintegration sehr stark im wissenschaftlichen Umfeld bearbeitet wurde und Konzepte wie die der Föderierten Datenbanksysteme hervorbrachte [SL90], wird die Anwendungsintegration eher von der Industrie vorangetrieben. Hier findet man unter dem Akronym EAI (*Enterprise Application Integration*) Ansätze, die eine Interoperabilität zwischen sogenannten Anwendungssystemen ermöglichen sollen, welche keine Datenschnittstelle wie SQL, sondern den Zugriff nur über eine Menge an Funktionen, ein sogenanntes API (*Application Programming Interface*), unterstützen. Die Relevanz dieser Integrationsthemen wird auch durch zunehmende Standardisierungsbemühungen unterstrichen. Auf der Datenseite wird der SQL-Standard [ANSI99] um Part 9: SQL/MED (*Management of External Data*, [ISO00]) erweitert. Dieser Teil des Standards legt die Schnittstellen zwischen dem Datenbank-Server und sogenannten Wrappern fest, über die heterogene Systeme angebunden werden können. Auf der anderen Seite gibt es im Rahmen der Applikations-Server die Definition der J2EE-Konnektoren (*Java 2 Enterprise Edition*, [Sun00]), die es ermöglichen sollen, unterschiedlichste Systeme in standardisierter Form über einen Applikations-Server anzusprechen. Diese Standards sind vor allem für Unternehmen von großer Bedeutung, um die Portabilität ihrer realisierten Integrationslösungen zu garantieren. So wäre ein Wechsel des strategisch gesetzten Datenbanksystems oder auch Applikations-Servers mit erheblich weniger Aufwand verbunden, als dies momentan der Fall ist.

Wir wollen nun den Fall betrachten, in welchem Unternehmensdaten aus allen Informationssystemen, also aus Datenbank- als auch Anwendungssystemen zusammengeführt werden sollen. Das bedeutet, daß ein Ansatz in der Lage sein muß, sowohl Daten als auch Funktionen zu integrieren. Auf den ersten Blick versprechen die Standardisierungsbemühungen zu Wrappern und Konnektoren Lösungen für Probleme der Integration von Daten und Anwendungssystemen. Mit beiden Konzepten kann man theoretisch eine Daten- und Funktionsintegration realisieren. Doch zeigt sich, daß es Integrationsszenarien gibt, die nach wie vor nicht richtig abgedeckt werden können.

In diesem Aufsatz wollen wir zunächst in den Kapiteln 2 und 3 die beiden Ansätze kurz vorstellen. Anschließend führen wir in Kapitel 4 unsere Anforderungen an eine Integrationslösung auf, um daraufhin die Realisierungsmöglichkeiten auf Basis von Wrappern bzw. Konnektoren zu untersuchen. Des weiteren schlagen wir vor, die beiden Ansätze zu kombinieren und skizzieren hierzu eine mögliche Architektur. Kapitel 5 faßt unsere Gedanken abschließend zusammen.

2 Wrapper

Im folgenden soll kurz der Aufbau der in SQL/MED definierten Wrapper-Architektur [ISO00] vorgestellt werden. Die grundlegende Idee dieser Architektur sieht vor, daß beliebige Datenquellen – relational oder nicht-relational – über Wrapper an einen SQL-Server angebunden werden können. Globale Anfragen werden von dem SQL-Server bearbeitet, der als eine Art Integrations-Server fungiert und die globale Anfrage in die entsprechenden Teilanfragen für die angebundenen Systeme zur lokalen Verarbeitung aufteilt. Die Wrapper stellen hierzu eine standardisierte Schnittstelle zu den Datenquellen bereit, so daß für den SQL-Server alle integrierten Quellen über SQL angesprochen werden können. Zu den Aufgaben des Wrappers gehören die Übersetzung der Teilanfrage in den entsprechenden Schnittstellenaufruf des lokalen Systems sowie die Aufbereitung des Ergebnisses zu Tabellen, die anschließend wiederum an den SQL-Server zurückgereicht werden. Handelt es sich bei den lokalen Systemen um Anwendungssysteme mit Funktionsschnittstellen, so müssen die Funktionen auf abstrakte Tabellen im SQL-Server abgebildet werden, um sie in SQL-Anfragen referenzieren zu können. Abbildung 1 illustriert die Architektur.

Der Wrapper-Ansatz weist die Eigenschaft auf, daß die integrierten Systeme in die datenorientierte SQL-Welt überführt werden und somit deren Vorteile genutzt werden können. Dazu gehören die standardisierte deklarative Datenschnittstelle, die aufgrund ihrer hohen Flexibilität auch Ad-hoc-Anfragen unterstützt. Des weiteren können Daten auf sehr effiziente Art und Weise verarbeitet werden, wobei die Anfrageoptimierung eine wichtige Rolle spielt.

3 J2EE-Konnektoren

Betrachten wir nun die Architektur der J2EE-Konnektoren [Sun00], so stellen wir fest, daß sie der Wrapper-Architektur sehr ähnelt. Auch hier stellt der

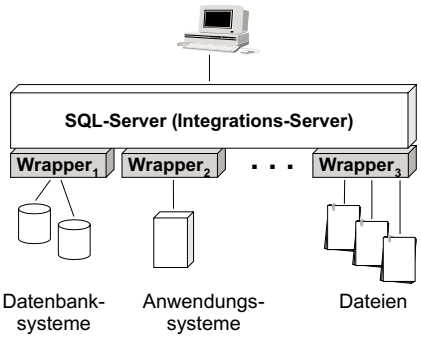

Abb. 1: Wrapper-Architektur.

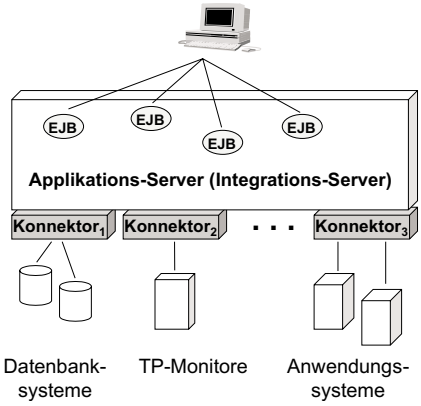

Abb. 2: Konnektoren-Architektur.

Applikations-Server eine Art Integrations-Server dar, an welchen über J2EE-Konnektoren (auch *Resource Adapter* genannt) unterschiedliche Informationssysteme angebunden werden können, von Anwendungssystemen über Transaktionsmonitore bis hin zu Datenbanksystemen. Die Konnektoren stellen ebenfalls standardisierte Schnittstellen zu den lokalen Systemen dar, so daß die Entwickler von Java-Anwendungen über den Applikations-Server in standardisierter Form auf unterschiedliche Informationssysteme zugreifen können, unabhängig von Plattform und Hersteller (siehe Abbildung 2).

Betrachten wir nun die Eigenschaften dieses Ansatzes, so muß klar herausgestellt werden, daß es sich bei diesem Konzept primär um die Bereitstellung standardisierter Schnittstellen und Dienste (Unterstützung von Transaktionen, Sicherheit und *Connection Pooling*) handelt. Die Konnektoren sind in erster Linie als Werkzeuge für den Java-Entwickler gedacht. Es liegt keine Ausführungskomponente vergleichbar dem SQL-Server vor, die bereits eine bestimmte Funktionalität zur Verfügung stellt. Statt dessen muß die eigentliche Integrationslogik im Applikations-Server mittels EJBs (*Enterprise JavaBeans*) und in der globalen Anwendung implementiert werden.

4 Anforderungen und deren Realisierungsmöglichkeiten

Nachdem wir die Konzepte der Wrapper und J2EE-Konnektoren kurz umrissen haben, zeigen wir nun unsere Anforderungen an eine Integrationslösung auf. Anschließend sollen die beiden vorgestellten Konzepte bezüglich ihrer Einsetzbarkeit untersucht und verglichen werden. Basierend auf diesen Ergebnissen betrachten wir die Kombinationsmöglichkeit der beiden Ansätze.

4.1 Anforderungen

In unserem Integrationsszenario gehen wir davon aus, daß Unternehmensdaten aus unterschiedlichen Informationssystemen integriert werden sollen. Bei den

Systemen handelt es sich um Datenbank- als auch Anwendungssysteme, d. h. es muß eine Daten- und Funktionsintegration unterstützt werden. In erster Linie sollen Daten effizient verarbeitet und eine entsprechende flexible Datenschnittstelle zur Verfügung gestellt werden. Des weiteren ist es wünschenswert, die Integrationslogik und deren Mechnismen nicht selbst implementieren zu müssen. Außerdem sollte der Entwicklungaufwand zur Anbindung der lokalen Systeme so niedrig wie möglich gehalten werden. Abbildung 3 zeigt ein mögliches Integrationsszenario.

4.2 Unterstützung durch die beiden Konzepte

Wir wollen nun untersuchen, wie die in Abschnitt 4.1 geforderte Integration jeweils von den beiden Konzepten realisiert werden kann. Betrachten wir den **Wrapper-Ansatz**, so ist die Anbindung der relationalen Datenbanksysteme aufgrund des übereinstimmenden Datenmodells nicht allzu schwer und soll hier nicht weiter untersucht werden. Schwieriger gestaltet sich die Anbindung der Anwendungssysteme und deren Funktionen. Hier müssen drei Problemfelder unterschieden werden, die der jeweilige Wrapper bewältigen muß. Erstens müssen relationale Teilanfragen in die entsprechenden Funktionsaufrufe der unterschiedlichen Systeme abgebildet werden, wobei verschiedene Programmiersprachen und unterschiedliche Funktionalitäten abgedeckt und eingebracht werden müssen. Zweitens müssen Funktionen auf Tabellen und damit zwei vollkommen unterschiedliche Konzepte aufeinander abgebildet werden. Drittens muß auch die durch die Systeme und deren Funktionen unterstützte Funktionalität in die SQL-Welt abgebildet und entschieden werden, ob diese Funktionalität durch den Wrapper noch erweitert werden soll. Dies kann vor allem die verteilte Verarbeitung der globalen SQL-Anfrage beschleunigen, indem beispielsweise die Projektion durch den Wrapper unterstützt wird und somit weniger Daten zwischen lokalem System und SQL-Server transportiert werden müssen. Als großer Vorteil bei diesem Ansatz ist hervorzuheben, daß mit dem SQL-Server bereits eine implementierte Ausführungskomponente, sozusagen die *Engine*, zur Verfügung steht, welche die Integration umsetzt. Außerdem kann mit SQL auf eine flexible Schnittstelle aufgesetzt werden, die mehr oder weniger von allen Software-Herstellern als Datenschnittstelle unterstützt wird.

Konzentrieren wir uns nun auf den **Konnektoren-Ansatz**, so sind die kritischen Punkte anders gelagert. Da es sich um einen funktionsorientierten Ansatz handelt, scheint eher die Anbindung der Datenbanksysteme eine Herausforderung darzustellen. Hier kann jedoch auf JDBC zurückgegriffen werden. Bei den Anwendungssystemen hingegen kann eine Abbildung der lokal angebotenen Funktionen auf die Konnektor-Schnittstelle erfolgen. In dem Konnektor müssen außerdem die Dienste zur Unterstützung der Transaktionen, der Sicherheit und des *Connection Pooling* realisiert werden. Was die eigentliche Umsetzung und Durchführung der Integration betrifft, so kann hier nicht auf eine bereits vorhandene *Engine* wie bei SQL-Servern zurückgegriffen werden. Statt dessen muß der Java-Entwickler die eigentliche Integration selbst im Applikations-Server oder

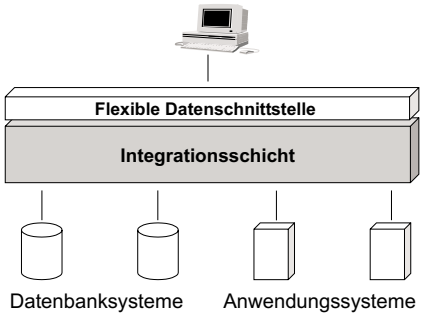

Abb. 3: Beispielhaftes Integrationsszenario.

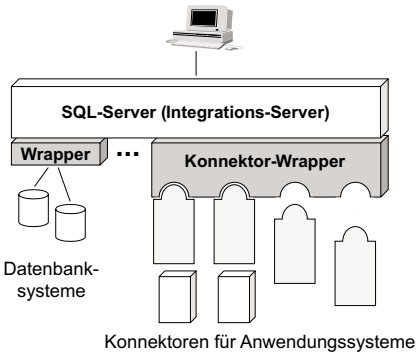

Abb. 4: Kombination von Wrappern und Konnektoren.

der globalen Anwendung realisieren. Des weiteren liegt keine flexible Datenschnittstelle für den globalen Anwender vor[1].

Es zeigt sich, daß grundsätzlich beide Konzepte die Integration von Daten und Funktionen unterstützen können. Stellen wir die Stärken und Schwächen der beiden Ansätze gegenüber, so ist offensichtlich, daß zur effizienten Verarbeitung der integrierten Daten und Umsetzung der Integration der Wrapper-Ansatz mit SQL-Server vorzuziehen ist. Die Realisierung der Wrapper scheint jedoch aufwendig und wird in den meisten Fällen selbst implementiert werden müssen, da bisher nur sehr wenige Software-Hersteller Wrapper gemäß SQL/MED unterstützen. Gerade andersherum zeigt sich die Lage bei den J2EE-Konnektoren. Zwar muß die Umsetzung der Integration selbst programmiert werden und es liegt noch keine flexible Datenschnittstelle vor. Mit der Konnektoren-Schnittstelle haben wir aber ein Konzept, das von den Herstellern sehr gut angenommen und bereits in zahlreichen Produkten umgesetzt wird. Es ist vor allem damit zu rechnen, daß alle bedeutenden Hersteller für ihre Systeme J2EE-Konnektoren anbieten werden und somit lediglich Konnektoren für im Hause entwickelte Systeme implementiert werden müssen. So gesehen scheint eine Kombination der beiden Konzepte eine interessante Alternative darzustellen, die wir im nächsten Abschnitt betrachten wollen.

4.3 Kombination der Konzepte

Bei der Kombination von Wrappern und Konnektoren gehen wir davon aus, daß unsere Integrationslösung grundsätzlich auf dem Wrapper-Ansatz basiert. Die Konnektoren kommen zum Einsatz, wenn Anwendungssysteme angebunden werden sollen. Dies scheint vor allem dann sinnvoll, wenn zukünftig damit zu rechnen ist, daß Software-Hersteller Konnektoren für ihre Systeme anbieten werden. Um die Konnektoren an den SQL-Server anzubinden, benötigt man wiederum einen Wrapper. Da die Konnektor-Schnittstelle standardisiert ist, können alle

[1] Es wird jedoch bereits an der Unterstützung von Anfragen gearbeitet.

Anwendungssysteme mit Konnektor über ein und denselben Wrapper integriert werden. Der Entwicklungsaufwand für die Integration kann auf diese Weise deutlich reduziert werden. Wurde nämlich einmal der Konnektor-Wrapper erstellt, so kann jeder beliebige Konnektor mit relativ geringem Aufwand in die Architektur eingehängt werden. Abbildung 4 verdeutlicht unsere Idee.

Natürlich gehen wir nicht davon aus, daß dieser Vorschlag alle Probleme löst. Vielmehr scheint er für unsere Anforderungen *eine* mögliche Lösung darzustellen. Grundsätzlich scheint es jedoch bei der Integration von Informationssystemen sinnvoll, die beiden Konzepte zu kombinieren, um somit die Stärken beider Ansätze nutzen zu können. Wie diese Kombination aussieht, hängt maßgeblich von den zu integrierenden Systemen, den Anforderungen an die Datenverarbeitung sowie der globalen Schnittstelle ab.

5 Zusammenfassung

In diesem Aufsatz haben wir uns der aktuellen Diskussion zu Wrappern und J2EE-Konnektoren angeschlossen, um deren Einsatzmöglichkeiten bei einem bestimmten Integrationsszenario zu untersuchen. Nachdem die beiden Konzepte kurz vorgestellt wurden, haben wir unsere Anforderungen an eine Integrationslösung aufgeführt. Anschließend wurde untersucht, wie die Integration mittels Wrapper bzw. Konnektoren realisiert werden kann: dabei wurden deren Stärken und Schwächen aufgezeigt. Es wurde deutlich, daß die beiden Konzepte alleine keine optimale Lösung bieten. Vielmehr erscheint es sinnvoll, Wrapper und Konnektoren in einer Architektur zu kombinieren, um somit die Vorteile beider Ansätze zusammenzuführen. Weiterhin wurde eine mögliche Kombinationsarchitektur skizziert und darauf verwiesen, daß auch andere Kombinationsmöglichkeiten sinnvoll sein können.

Referenzen

ANSI99: American National Standards Institute, Inc. (1999). Database Languages – SQL – Part 2: Foundation (SQL/Foundation). ANSI/ISO/IEC9075-2-1999. In: Amercian National Standard for Information Technology, Dezember 1999.

ISO00: ISO/IEC (2000). Database Language – SQL – Part 9: Management of External Data (SQL/MED), September 2000. Final Committee Draft.

SL90: A. P. Sheth, J. A. Larson (1990) *Federated Database Systems for Managing Distributed, Heterogeneous and Autonomous Databases.* In: ACM Computing Surveys, 22(3):183-236.

Sun00: Sun Microsystems (2000). J2EE Connector Architecture Specification. Version 1.0. http://java.sun.com/j2ee/connector/