Thomas Strehlow ist Gründer und Geschäftsführer der Oraylis GmbH, Düsseldorf. Er verantwortet den Bereich Data-Warehousing. Mit seinem Software-Architektenteam ist er für zahlreiche DWH-Projekte verantwortlich.
E-Mail: T.Strehlow@oraylis.de

Fachartikel

Geschäftsregeln

ETL trifft auf Objektorientierung

ETL (Extract, Transform, Load) ist der Standard für die Bewirtschaftung von Data Warehouses (DWHs) oder Datamarts, so viel ist klar. Was aber, wenn auf der ETL-Strecke überaus komplexe Geschäftsregeln erforderlich sind, die an die Grenze des klassischen ETL stoßen? In einem Projekt werden Verträge, Ereignisse an den Verträgen sowie eine Vielzahl weiterer Informationen nach etwa 50 Geschäftsregeln verarbeitet. Das Volumen beträgt mehrere 100 Millionen Datensätze und soll täglich innerhalb von zwei Stunden verarbeitet werden.

Die Herausforderung

Für ein komplexes Analysemodell im Mobilfunkmarkt werden die wichtigsten Informationen herangezogen. Hierzu gehören zum Beispiel etwa 100 Vertragsereignisse, die aus unterschiedlichen Quellen bereitgestellt werden. Die Kernereignisse sind die Aktivierung eines Vertrags oder die Vertragsverlängerung. Aber auch die Rufnummernänderung (Mobile Number Portability), der Tarifwechsel oder die Übergabe an einen anderen Vertriebspartner sind Vertragsereignisse. Jeder Vertrag hat einen Tarif sowie mehrere Optionen. Eine Option, vergleichbar mit den Extras bei PKWs, stellt eine individuelle Tarifeigenschaft dar. Auch hier gibt es eine Vielzahl von Optionsereignissen, die zum Beispiel eine Aktivierung, eine Verlängerung oder der Wechsel einer Option darstellen. Analytisch soll der gesamte Kundenstamm betrachtet werden. Die führenden Systeme sind aber verteilt, zum Beispiel nach Prepaid oder Postpaid. Ferner gibt es weitere Quellsysteme für spezielle Vertriebskanäle. Zusätzliche Informationen wie der Deckungsbeitrag oder Vertriebsprämien kommen aus ganz anderen Datenquellen. Darüber hinaus gibt es das gesamte Analysemodell dreifach: die Realität, reflektiert aus den Datenquellen. Systemisch bedingt steht der Umsatz eines neuen Vertrags erst nach Nutzung und Rechnungsstellung etwa sechs bis acht Wochen nach der Vertragsaktivierung zur Verfügung. Analytisch besteht der Bedarf, den abgeleiteten Deckungsbeitrag direkt zum Aktivierungszeitpunkt bereitzustellen. Dieser kann aufgrund des Tarif- und Optionsmix prognostiziert werden, das heißt ein Prognoseprozess liefert weitere Daten zu den Verträgen. Als Drittes gibt es noch weitere umfängliche Planungsinformationen, die je nach Vertriebskanal wiederum auf einer vollständig anderen Datenbasis beruhen. Technisch sind die meisten Informationen im DWH abgelegt. Fachlich sind sie jedoch nicht gegeneinander auswertbar.

SQL Server 2008 R2 als relationale Datenbank
SQL Server Integration Services als ETL Tool
Visual Studio und C# als objektorientierte Sprache und Entwicklungswerkzeug
Visual Studio Profiler zur Aufspürung von Leistungsengpässen
NUnit als Unit Testing Tool http://www.nunit.org
BI.Quality als Testframework zum automatisierten Vergleich von Query Ergebnissen http://biquality.codeplex.com

Tab. 1: *Verwendete Tools*

Eine weitere Problematik ist die teilweise unzuverlässige Datenlieferung. So gibt es zum Beispiel eine signifikante Nachzüglerproblematik, das heißt, es werden teilweise Daten der letzten sechs Monate nachträglich korrigiert. Das System wird in Anlehnung an SCRUM nach einer agilen Projektmethode weiterentwickelt und in Abständen von etwa sechs Wochen mit neuer Funktionalität im Produktivsystem bereitgestellt. Damit einher geht eine weitere Anforderung an die Bewirtschaftung – es muss ein Voll-Deployment und damit eine Komplettbewirtschaftung der letzten drei Jahre innerhalb von drei Tagen möglich sein. Täglich werden die letzten zwei Monate bewirtschaftet – das sind etwa 200 Millionen Datensätze –, am Wochenende die letzten sechs Monate, etwa 600 Millionen Datensätze (siehe Tabelle 2). Zu einem Voll-Deployment werden, aufgrund weiterer Artefakte, etwa 50 Milliarden Datensätze bewegt. Aus den oben aufgeführten Gründen scheidet eine Optimierung der ETL-Strecke im Sinne „Reduziere das Volumen und lade nur die Änderung“ aus. Vielmehr lautet die Devise: „Lösche immer einen ganzen Monat und lade diesen schnellstmöglich nach.“ Je nach Bewirtschaftung werden die Monate iterativ nacheinander bewirtschaftet.

Periode	Datenvolumen aus Monaten	Bewegte Records	maximales Zeitfenster für ETL-Prozesse
täglich	2	200Mio	2h
wöchentlich	6	600Mio	6h
ca. 8-mal im Jahr zum Deployment	>36	20Mrd	3d

Tab. 2: *Anforderungen an ETL*

Warum stößt klassisches ETL an seine Grenzen?

Die Stärken der ETL liegen darin, große, mächtige Datenströme in Batches in den Arbeitsspeicher zu laden und mit anderen Datenströmen zu verbinden und gezielt Zeilen- oder Gruppenoperationen durchzuführen. Im vorliegenden Fall geht es aber darum, ganze Vertragshistorien zu bewerten und gezielt zu korrigieren. Es werden zum Beispiel die Sequenzen eines Vertrags analysiert. Dabei werden die Daten aus unterschiedlichen Quellsystemen zusammengeführt. Die Daten durchlaufen teilweise die Quellsysteme, in denen sie manuell korrigiert beziehungsweise ergänzt werden. So wird zum Beispiel ein Neuvertrag in einem Prämiensystem mit einem Vertriebspartner und einer Vertriebsprämie versehen. Hierbei kann es zu Verzögerungen kommen. Der analytische Anwender will die Daten aber schnellstmöglich im System sehen. Die Geschäftslogik muss infolgedessen individuell je Datensatz entscheiden, welches System das führende ist. Leider gibt es nicht nur zwei Quellsysteme mit einer solchen Problematik, sondern fünf, Tendenz steigend. Eine andere Herausforderung sind Sequenzanalysen und -korrekturen. Verträge werden teilweise storniert, bevor sie deaktiviert werden. Teilweise fehlt aber auch das eine oder andere Ereignis. So ist das Quellsystem für Optionen rein vertrieblich orientiert und kennt keine Deaktivierungen. Für eine korrekte Bestandsmessung sind aber richtige und vollständige Ereignisse zwingend erforderlich. Für eine tagesgenaue Bestandsmessung ist jedoch nur das erste Deaktivierungsereignis sinnvoll. Alle weiteren Ereignisse sind nicht mehr bestandswirksam, denn der Vertrag ist bereits im fachlichen Sinne deaktiviert. Da im weiteren Verlauf mit mächtigen OLAP-Cubes gearbeitet wird, ist es Aufgabe der ETL-Strecke, die Bestandswirksamkeit eines jeden Vertragsereignisses zu ermitteln. Dies erfordert mit klassischen ETL-Mitteln ein Aufstauen der Informationen oder ein Lookup gegen mehrere 100 Millionen Datensätze. Beim ersten Lösungsansatz ist mit Scripting schnell die Wartbarkeitsgrenze erreicht. Ein Lookup, zumal es sich hier um einen Range-Lookup handelt, der auf Daten zugreift, die im Schritt vorher gegebenenfalls aktualisiert oder gelöscht wurden, stellt unter derartigen Laufzeitvoraussetzungen ebenfalls keine Option dar. In Summe waren rund 50 wie hier beschriebene Geschäftsregeln umzusetzen, die ein Umdenken hinsichtlich der Herangehensweise erforderten.

Die Lösung

Das verwendete ETL-System, „SQL Server Integration Services“ verfügt über eine Schnittstelle (API) zur Integration sogenannter Custom Components. Damit können zum Beispiel über die Programmiersprache C# eigene Komponenten in den ETL-Strom eingebettet werden. Im vorliegenden Fall wurden über ein geeignetes ETL-Design die fachlich erforderlichen Datenströme aufbereitet und korrekt sortiert an eine Komponente übergeben.

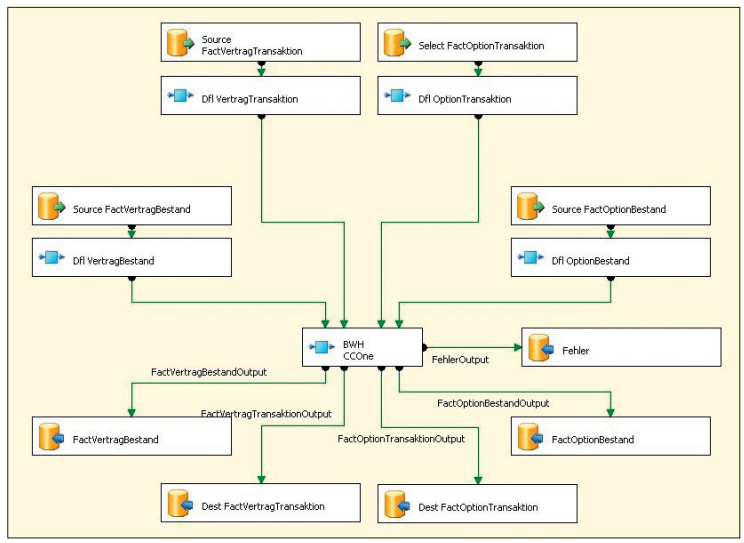

Abb. 1: Komplexe Geschäftslogik in Form einer Custom Component

Diese erhält über vier Datenströme Vertrags- und Optionsinformationen. Sie werden in der Komponente aus den

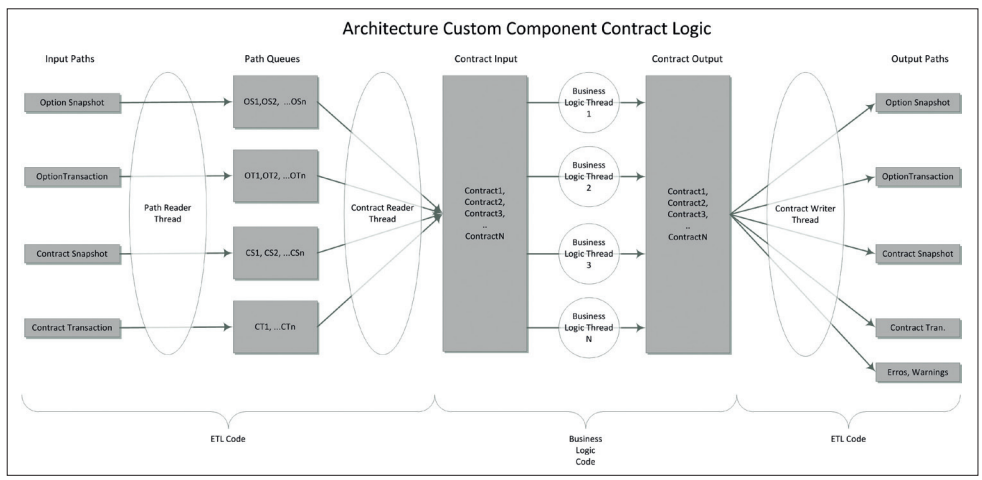

Abb. 2: Grundlegende Architektur der Custom Component mit Queues und Threads

vier Strömen gelesen. Dabei werden die Ereignisse jeweils eines Vertrags in Containern abgelegt. Diese Container werden Multithreaded, das heißt in parallelen Arbeitsprozessen von den oben genannten 50 Geschäftsregeln verarbeitet und entsprechend korrigiert. Jegliche Fehler und Warnungen werden über einen Fehlerkanal protokolliert. Am Ende wird der Vertrag wieder zerlegt und auf die Ausgabeströme geschrieben. In diesem Fall macht jeder Teil das, was er am besten kann. ETL sorgt für die großen Datenströme und die Sortierung. Die objektorientierte Komponente erhält stückweise einzelne Verträge und liest, korrigiert, priorisiert, generiert oder löscht die Daten jeweils eines Vertrages. Der Workflow ist in Abbildung 2 dargestellt, der Stapel der Geschäftsregeln in Abbildung 3.

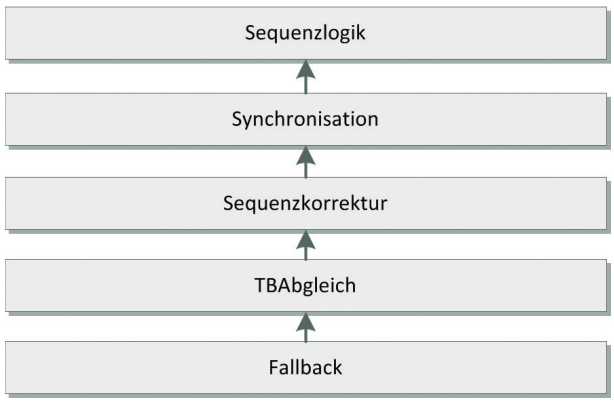

Abb. 3: Funktionsblöcke innerhalb der Geschäftslogik

Best Practices

Ist die Entscheidung für eine derartige Lösung gefallen, sollten die nachfolgenden Punkte beachtet werden: Die meisten ETL-Werkzeuge legen ein API offen, das die Integration eigener Data-Sources oder Transformationen ermöglicht. Diese sind, der Performance der ETL-Strecke geschuldet, oft recht komplex und meist zu technisch für Geschäftslogik-Entwickler. In der Architektur ist man gut beraten, den technischen Code der ETL-API über eine individuelle Schnittstelle oder besser noch durch ein separates Entwicklungsprojekt zu abstrahieren. Damit erreicht man auf der einen Seite, dass der „Abstraktionscode" wiederverwendbar ist. Auf der anderen Seite ist der Geschäftslogik-Code frei vom API-Code des ETL-Tools. Das erleichtert das Verständnis der Entwickler und Debugging ist einfacher möglich. Zusätzlich hat es sich etabliert, den Geschäftslogik-Code durch Unit Tests durchgängig zu validieren. Das hat den unschätzbaren Vorteil, dass der Code von Beginn an leichter testbar ist. Diese Test-Cases können direkt gestartet und debugged werden. Damit kann die Logik unabhängig von ETL und Daten sehr leicht und schnell geprüft werden. Es ist dabei allerdings zu beachten, dass dem Testcode die gleiche Aufmerksamkeit gewidmet werden muss wie dem Business Code selbst. Ist der Code fertig, hat es sich weiter bewährt, wenn ein abgestimmtes Set von Testdaten definiert wird. Hiermit kann sozusagen ein Ende-zu-Ende-Prüfstand aufgebaut werden. Damit wird die Geschäftslogik, eingebettet in die fertige ETL-Strecke, von Ende zu Ende getestet. Mit dem Open-Source-Framework „BI.Quality" können die Ausgabewerte in der Zieldatenbank mit Erwartungswerten verglichen werden. Dieses Vorgehen ist natürlich unabhängig von der hier vorgestellten Objektorientierung und bei allen BI-Projekten anzuraten. Es funktioniert übrigens bei jeder relationalen oder multidimensionalen Datenbank, die über einen OLE-DB-Treiber verfügt. Frei nach dem Motto „Make it Run, Make it Right, Make it Fast" wurde im hier vorgestellten Projekt die ETL-Strecke am Ende noch einer Leistungsmessung mit einem Profiler unterzogen. Dabei wird der Code automatisch mit Probing-Methoden ergänzt, und nach einem Durchlauf erhält man sehr detaillierte Informationen, an welcher Stelle wie viel Zeit der Gesamtlaufzeit benötigt wird. Im vorliegenden Fall konnte durch diesen Schritt die Bewirtschaftungszeit noch mal erheblich verbessert werden.

Performance-Vergleich

Auf Basis der vorherigen, reinen ETL-Implementierung benötigte die Bewirtschaftung mehrere Stunden für die etwa 100 Millionen Datensätze eines Monats. Mit dem neuen objektorientierten Ansatz können diese Daten jetzt in 17 Minuten bewirtschaftet werden. Wird die Custom Component abgeschaltet, so benötigt der reine Datentransport bereits 15 Minuten. Das heißt, die Laufzeit der gesamten Geschäftslogik kostet netto nur zwei Minuten. Die Leistung übersteigt die gesetzten Erwartungen und zeigt, dass mit diesem Verfahren bisherige Leistungsgrenzen überschritten werden können.

Fazit

Wenn ETL im klassischen Sinne an Grenzen stößt, dann ist noch lange nicht Schluss. Durch die Möglichkeit der Integration von Custom Components in den ETL-Strom ergeben sich ganz neue Möglichkeiten. Die komplexesten Geschäftslogiken können mit objektorientierten Sprachen sicher und wartbar entwickelt werden. Debugging- oder Unit-Tests werden umfänglich unterstützt und ermöglichen eine professionelle Qualitätssicherung. Im Umfeld eines agilen Projektmanagements ergibt sich durch dieses Verfahren eine ideale Symbiose. Bei der Befolgung der Best-Practice-Ansätze ist der anfängliche Mehraufwand überschaubar. Der einzige Wehmutstropfen besteht darin, dass neben der klassischen ETL-Entwicklung auch die objektorientierten Grundsätze beherrscht werden müssen. Im hier beschriebenen Projekt hatten einige ETL-Entwickler eine Softwareentwicklungsvergangenheit, und die neue Herausforderung führte zu einer großen Motivation. Bei der richtigen Rollenverteilung zwischen ETL und Objektorientierung spielt die Kombination ungeahnte Stärken aus.