BI-Architekturen der Zukunft

Teil II: Selfservice-BI und eine BI-Architektur ohne klassisches Data Warehouse

von Rita Hennig und Mathis Manz

Im ersten Teil der Artikelreihe über BI-Architekturen der Zukunft haben wir die Dauerbrenner-Themen (Near-)Realtime und Analytic Excellence beleuchtet.[1] Doch es gibt noch mehr Aspekte und Ideen, um die Möglichkeiten der klassischen Drei-Schichten-Data-Warehouse-Architektur zu erweitern.

Selfservice-BI – eine Einführung

Von Selfservice-BI spricht man, wenn Anwender aus den Fachabteilungen eigenständig und weitgehend unabhängig von der IT-Abteilung auf wichtige Unternehmensinformationen zugreifen und diese analysieren. Selfservice-BI umfasst zwei Bausteine: zum einen das klassische Ad-hoc-Reporting, zum anderen die gemeinsame Nutzung von zentralem Inhalt mit privaten Daten, wobei zentraler Inhalt die bereits im DWH vorhandenen Daten und Auswertungen oder sogar Rechenkerne sein kann.

Da der erste Baustein – Ad-hoc-Reporting – bekannt und weit verbreitet ist, in die klassische Architektur passt und es zahlreiche Werkzeuge gibt, mit denen sich Anwender aus einem zentralen Daten- und Berechnungspool (Kennzahlen-Repository) eigene Auswertungen zusammenstellen können, ist er nicht Gegenstand dieses Artikels.

Interessant ist der zweite Baustein, also das zusätzliche Einbeziehen von privaten Endanwenderdaten, die eigentlich nicht zum Datenhaushalt der Enterprise-BI-Landschaft gehören, in Auswertungen und Berechnungen, wobei diese privaten Daten keinesfalls für andere Nutzer sichtbar sein dürfen und – in der Regel mittels Sandboxing – sicher abgeschottet sein müssen. Eine BI-Architektur, die Selfservice-BI unterstützen soll, benötigt intuitiv bedienbare Werkzeuge zur Report- und Auswertungsdefinition. Diese Werkzeuge müssen nicht nur mit den bekannten Strukturen der Zentrallandschaft (dem Enterprise Data Warehouse und seinen Data Marts) umgehen, sondern auch neue, unbekannte Strukturen aufgreifen können.

Eine erhebliche Bedeutung hat in diesem Zusammenhang ein vollständiges Metadaten-Repository. Das darin enthaltene Wissen über Datenstrukturen, Formeln und Datenbeziehungen muss sowohl maschinell ausgewertet werden können (zum Beispiel generisch maschinell ausführbare Berechnungsvorschriften) als auch für den Endanwender verständlich sein. Jede Datenstruktur, jede Datenbeziehung und jede Datenherkunft muss erklärt sein.

1 Siehe Hennig, Manz, BI-Architekturen der Zukunft, Teil I: (Near-)Realtime und Analytic-Excellence, msgGillardon NEWS 01/2017

Idealerweise erlaubt das Metadaten-Repository dem Nutzer ein Browsen (Drill-down und Drill-through) durch die Datenarchitektur und erklärt die Zusammensetzung von Ergebnissen.

Einer der wichtigsten Aspekte beim Selfservice-BI ist der Schutz der Nutzerdaten vor unberechtigtem Zugriff. Zum Einbringen der privaten Daten des Endanwenders ist eine persönliche Firewall, die diese Daten strikt von allen anderen Nutzern abschirmt, unabdingbar. Dazu werden vom Nutzer ausgewählte (Berechnungs-)Programme gemeinsam mit seinen privaten Daten und wiederum ausgewählten Daten des Zentraldatenhaushalts in eine eingeschränkte Umgebung gebracht, auf die nur der einzelne Endanwender Zugriff hat. Im Idealfall wählt der Nutzer die Bausteine für seine Berechnungen, die er aus der zentralen öffentlichen BI-Landschaft nutzen möchte, selbst aus. Dabei unterstützt ihn ein aussagekräftiges Metadaten-Repository, das die Abhängigkeiten zwischen den Elementen kennt. Anderenfalls stehen dem Benutzer alle Kennzahlenformeln und Rechenkerne zur Verfügung, was allerdings sehr unübersichtlich sein kann. Das Abschirmen von anderen Nutzern geschieht mittels ausgefeilter Techniken: vom „Umbiegen" des Dateisystems bis zur Simulation eines kompletten Rechners – ergänzt um Grenzkontrollen, wie sie von den Internettechniken bekannt sind.

Eine weitere wichtige Komponente sind geeignete Lademechanismen, um die privaten Daten des Nutzers in die Verarbeitung einzubringen. Im ungünstigen Fall werden diese Daten direkt in die Auswertungen (zum Beispiel einen Cube oder Report) eingebunden, wodurch zentrale Formeln oder Rechenkerne der darunterliegenden Businesslogikschicht des BI-Systems nur begrenzt genutzt werden können. Um diese Funktionalitäten auch für private Daten vollumfänglich nutzen zu können, müssen diese bereits an die darunterliegende Daten- und Verarbeitungsschicht angebunden werden. Es ist oft sinnvoll, dieselben Staging-Mechanismen und dieselbe Datenarchitektur zu nutzen wie für zentrale Daten. Die privaten Daten stellen dabei einfach ein weiteres Quellsystem dar. Ein weiterer Vorteil dieser Lösung ist die Nachvollziehbarkeit und Wiederaufsetzbarkeit von Auswertungen.

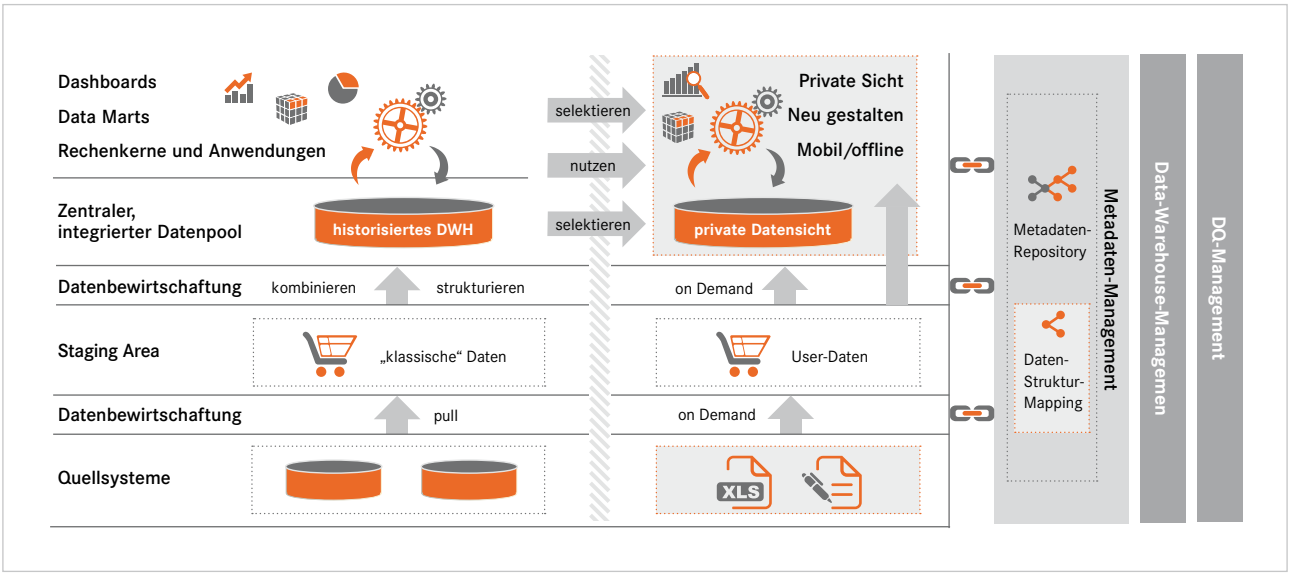

Abbildung 1: Sandboxing und Selfservice-BI auf einen Blick und völlig sicher: eigene Daten – zentrale Methoden – zentrale Daten – eigene Auswertungen

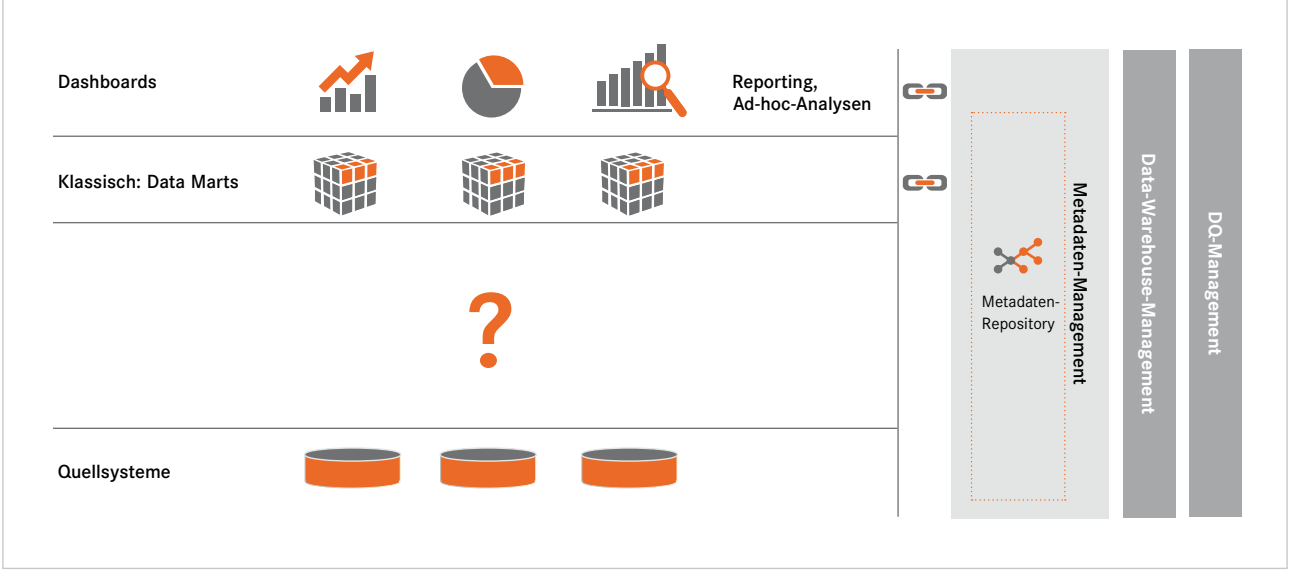

Abbildung 2: Ist das klassische Data Warehouse eigentlich noch zeitgemäß?

Wenn Rechenkerne oder granulare Berechnungen genutzt werden sollen, sollte es auch im Selfservice-BI eine (logische) Integrationssicht im DWH-Datenmodell geben. Zentrale Daten können selektiert und als Kopie oder View bereitgestellt werden. Die privaten Daten werden dann dazugemappt. Hier kommt wieder das Metadaten-Repository zum Einsatz, das durch seine Strukturkenntnis das Mappen erst ermöglicht. Auf Historisierung kann in der Regel verzichtet werden. Für einfachere Anwendungsfälle können die aufbereiteten Daten aber auch direkt in bestehende/zentrale oder neue/eigene Auswertungen gezogen werden.

Im Unterschied zum klassischen, zentralen Data Warehouse, wo in der Regel ein tägliches Laden im Batch stattfindet, müssen die Ladeprozesse für Selfservice-BI nach Bedarf (on Demand) ausgeführt werden können. Dies liegt in der Verantwortung der Data-Warehouse-Management-Komponente, die mit diesen Zusatzprozessen umgehen können muss. Außerdem muss die Datenstruktur der Fremddaten gescannt und gegebenenfalls bereits ein Mapping auf das zentrale Datenmodell vorgeschlagen werden können.

Selfservice-BI ist ein prädestinierter Anwendungsfall für mobile Techniken. Beispiele sind die Bereitstellung von Rechenkernen als Webservice und der gesicherte Zugriff auf zentrale Daten über eine Enterprise Cloud.

BI-Architektur ohne klassisches Data Warehouse?

Bei der Auseinandersetzung mit dem Thema BI-Architekturen der Zukunft stellt sich die provokante Frage „Wird ein Data Warehouse (DWH) eigentlich noch gebraucht?" beziehungsweise „Ist das klassische DWH überhaupt noch zeitgemäß?“, wobei klassisches DWH eine zentrale Datenbank mit einer relationalen Tabellenstruktur meint, in der der integrierte Datenpool mit dem Gesamthaushalt an Entitäten abgelegt ist, der für die Versorgung der BI-Landschaft benötigt wird und das mittels ETL-Prozessen (im Batch) beladen wird. Ist solch ein Monstrum nicht viel zu träge?

Mögliche Alternativen sind Design- und Architekturpattern, die ihren Ursprung außerhalb der BI-Welt haben. Könnten hier Mit-

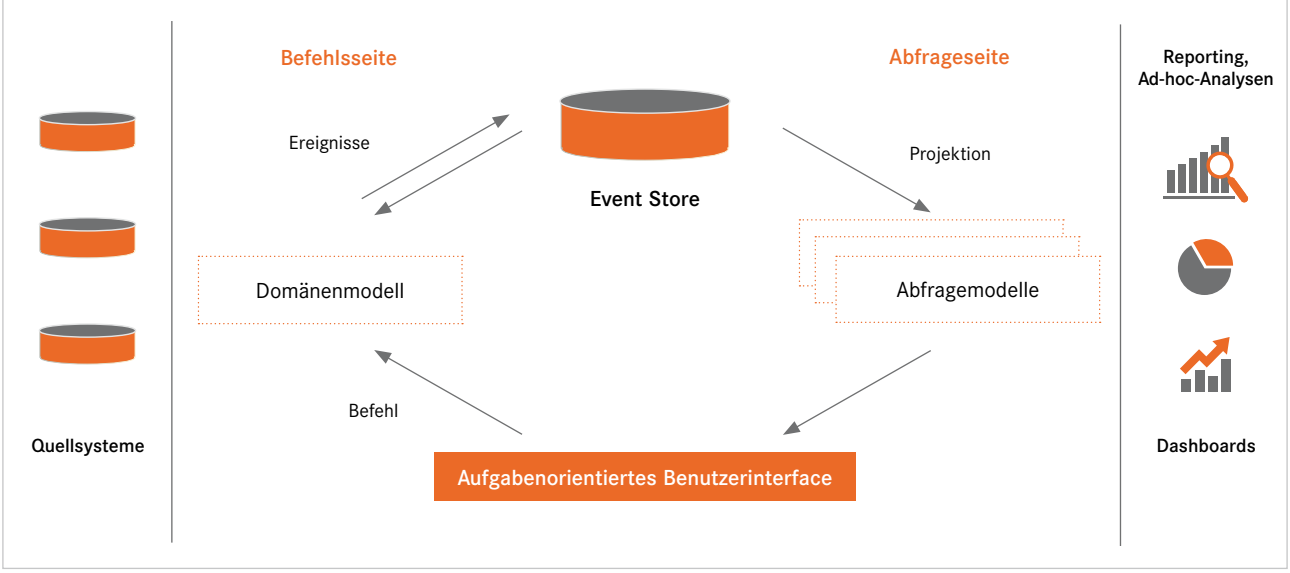

Abbildung 3: Command Query Responsibility Segregation (vereinfacht) in der BI-Architektur

tel und Prinzipien der Enterprise Application Architecture zum Einsatz kommen, die für die operative Welt entwickelt wurden? Zunächst geht es um eine Alternative zum Extract-Transform-Load(ETL)-Mechanismus, der sich in der klassischen Welt unidirektional durch die Schichten der BI-Architektur zieht und in der Regel durch zeitliche Regeln angestoßen wird. Beim ETL werden (in der Regel) normalisierte Entitäten ins DWH geschrieben und im selben Datenmodell von den Abnehmern gelesen. Ansatzpunkt für die folgenden Überlegungen ist die Tatsache, dass Datenlieferanten und Datenabnehmer völlig verschiedene Datenstrukturen kennen und benutzen.

Eine Alternative kann das Entwurfsmuster CQRS (Command Query Responsibility Segregation) sein. Laut Heise ist CQRS „ein erfolgreicher Gegenentwurf zum klassischen Schichtenmodell für Systeme mit parallelem Nutzerzugriff. Das Prinzip fordert eine Aufteilung in Verhaltens- und Abfragemodelle, mit der man die Geschäftslogik als wertvollsten Bestandteil einer Anwendung von der Datenbereitstellung für Benutzerschnittstellen und Reporting entkoppelt entwickeln kann.“[2] Zur grafischen Verdeutlichung sind die Schichten der BI-Architektur auf die Seite gekippt (siehe Abbildung 3). Auf der Befehlsseite steht ein Domänenmodell, in das die Daten aus den Quellen per Befehl eingefügt werden. Auslöser für das Laden wäre nach diesem Entwurfsmuster das Vorliegen neuer Quelldaten. Das Domänenmodell übernimmt hier die Integrationsaufgabe eines klassischen DWH.

Auf der Abnehmerseite (der Abfrageseite) stehen Abfragemodelle, die ähnlich wie Data Marts auf die Bedürfnisse der auswertenden Anwendungen zugeschnitten sind. Das Benutzerinterface kann als Bereitstellung von Schreib- und Leseoperationen für übergeordnete Ladeprozesse verstanden werden.

2 Quelle: Heise Medien: https://www.heise.de/developer/artikel/CQRS-neues-Architekturprinzip-zur-Trennung-von-Befehlen-und-Abfragen-1797489.html

Den Unterschied zur klassischen DWH-Architektur macht im Wesentlichen der Event Store aus, der beide Modelle miteinander synchronisiert. Das Prinzip des Event Store wurde von Martin Fowler[3] so beschrieben, dass Daten als generische Events abgelegt (und nicht transformiert) werden. Die Zuordnung eines Events zu seiner Bedeutung erfolgt über ein Repository. Damit zeigt auch dieser Ansatz die enorme Bedeutung des Metadaten-Repositorys. Jedes eintreffende Ereignis wird persistiert. Die Daten sind lückenlos nachvollziehbar. Wichtig ist, dass die Übermittlung von Datensätzen als einzelne Events massendatentauglich gestaltet werden.

In der deskriptiven Sicht auf die Daten, wie sie in der BI-Welt vorrangig ist, interessiert insbesondere ein bestimmter Zustand des Datenbestandes zu einem definierten Zeitpunkt. Dieser Zustand wird erst durch Projektion auf eine Reihe von Ereignissen abgeleitet. Ob dies persistent oder ad hoc geschieht, hängt unter anderem von der Leistungsfähigkeit der zugrunde liegenden Infrastruktur ab. Ebenso muss entschieden werden, ob die Bildung des Datenzustands auf der Seite der geschieht, weil für die Abnehmer relevant, oder im Domänenmodell, weil von zentralem Interesse. Letzteres entspricht eher der Idee des integrierten Datenhaushalts. In jedem Fall muss beachtet werden, dass eine Synchronität der Zustände verschiedener Ereignistypen (Entitäten) hergestellt wird. Es wird ein Zustand eines gesamten Bestandes zu einem Zeitpunkt benötigt.

Zusammenfassend zeigt sich bei diesem Ansatz einer modernisierten BI-Architektur, dass ins Zentrum anstelle des klassischen DWH ein Event Store tritt. Die Rolle von ETL-Prozessen übernimmt ein Event Bus mit Event-Handlern, die Publish-and-Persist-Methoden für Datenlieferanten und Subscribe-Methoden für Anwendungssysteme bereitstellen.

Eine solche Architektur ermöglicht eine hohe Aktualität der Daten. Der Event Store hält immer alle Daten – auch die aktuellsten. Jede Transaktion, jeder Cashflow, jede Kreditzusage wird als Event festgehalten und kann unmittelbar von allen anderen gelesen werden.

Ein weiterer Vorteil dieser Architektur ist die Autonomie, die die angebundenen Systeme durch die generische Datenstruktur der ausgetauschten Informationen und durch den möglichen asynchronen Datenaustausch (Schreiben und Lesen per Event) bewahren können. Das Erzeugen und Verarbeiten der Events muss jedoch individuell auf die Quell- und Abnehmersysteme aufgesetzt werden.

Das Erfüllen der Nachweispflicht – in Finanzarchitekturen besonders wichtig – erfordert hier etwas Aufwand, denn die generischen Datenstrukturen müssen immer mithilfe des Meta-

3 Siehe: https://martinfowler.com martinfowler.com/eaaDev/EventSourcing.html

Abbildung 4: Projektion des Zustandes aus einer Folge von Ereignissen

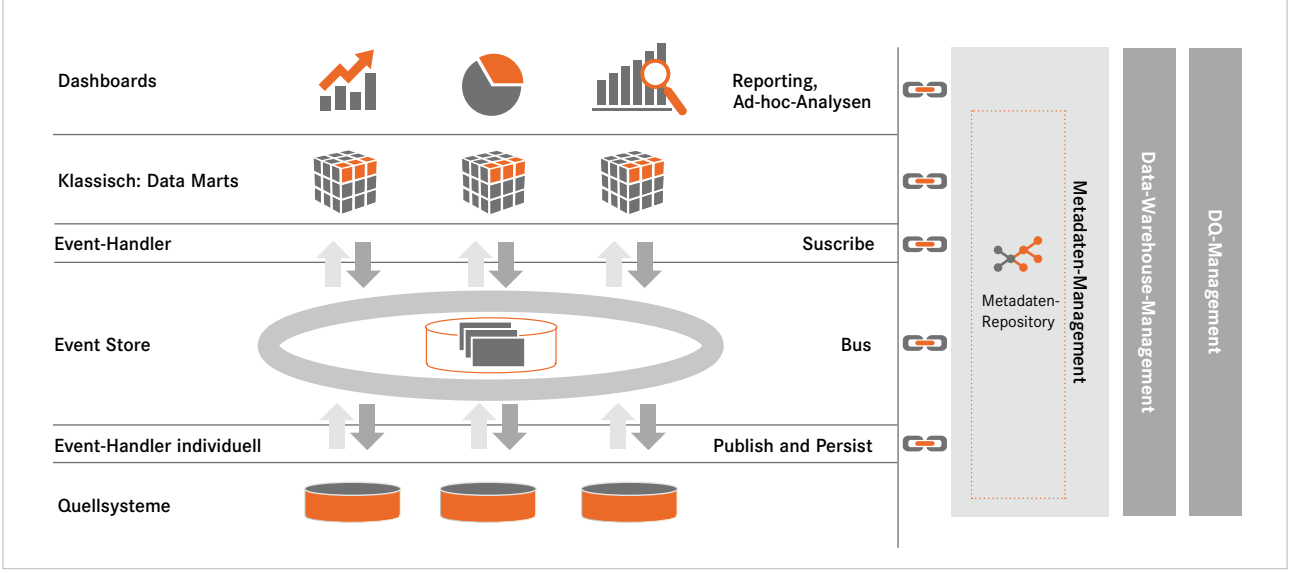

Abbildung 5: Die BI-Architektur der Zukunft?

daten-Repositorys ihrer Bedeutung zugeordnet werden. Doch es wird ausnahmslos jedes ankommende Event gespeichert. Datenänderungen, die verloren gehen können, gibt es nicht.

Auch die Einrichtung eines Berechtigungssystems erfordert neue Lösungen, da Rechte nicht, wie zumeist im klassischen DWH, allein auf Tabellen und Spalten definiert werden können. Die Definition der Zugriffsrechte muss sich immer auf den Dateninhalt beziehen.

Auch wenn mit diesem Ansatz noch längst nicht alle Fragen gelöst sind, ist er ein interessanter Denkanstoß für die zukünftige Weiterentwicklung, um Finanz-BI-Landschaften auch in Zukunft tragfähig gestalten zu können.

Ansprechpartner

Rita Hennig
Lead IT Consultant
> rita.hennig@msg-gillardon.de