BI Lifecycle - Wildwuchs oder klare Vorgaben?

Dirk Braunecker
Logica Deutschland GmbH & Co. KG, now part of CGI
Sulzbach (Taunus)

Schlüsselworte

BICC, DWH, BI, Framework, ETL

Einleitung

DWH- und BI-Projekte erleben häufig, hervorgerufen durch wechselnde Entwicklungsteams einen Wildwuchs im Bereich neuer Entwicklungen sowohl beim Source-Code als auch im Architekturbereich. Dies führt zu steigenden Wartungs- und Einarbeitungskosten und erschwert den flexiblen Einsatz verschiedener Entwickler.

Der Vortrag zeigt, wie man dies durch klare Rahmenbedingungen und Vorgaben verhindern oder zumindest verbessern kann, u. a. durch die Verwendung des Oracle Data Integrators.

Wildwuchs

Wildwuchs entsteht in der Regel in historisch gewachsenen DWH- und BI-Projekten. Die Gründe dafür sind u. a. die folgenden:

- Teilprojekte wurden mit unterschiedlichen Entwicklern realisiert
- Keine/Unspezifische Entwicklungsvorgaben und Standards
- Zeitdruck
- „Ich brauche schnell…"-Anforderungen
- Dokumentation wird auf das Projektende verschoben

Ein wichtiger Schritt hierbei ist das Erkennen des Wildwuchses und das Einleiten entsprechender Gegenmaßnahmen mit Hilfe von DWH Guidelines.

Das von Logica entwickelte BI Framework bietet Unterstützung anhand von Best-Practice Erfahrungen in vielen DWH- und BI-Projekten. Es dient u. a. dazu den Wildwuchs zu beseitigen oder in neuen Projekten komplett zu vermeiden.

Diese Guidelines beziehen sich auf die folgenden Abschnitte in einem Projekt:

Zielarchitektur

Die Effizienz eines DWH-Systems steht und fällt mit der vorgegebenen Zielarchitektur. Logica hat im Rahmen seines BI-Frameworks eine Referenzarchitektur entwickelt, die natürlich auf die Gegebenheiten des Kunden angepasst werden kann. Diese Referenzarchitektur ist in mehrere Layer aufgeteilt und erstreckt sich von den operationalen Daten bis zur Veröffentlichung von Daten.

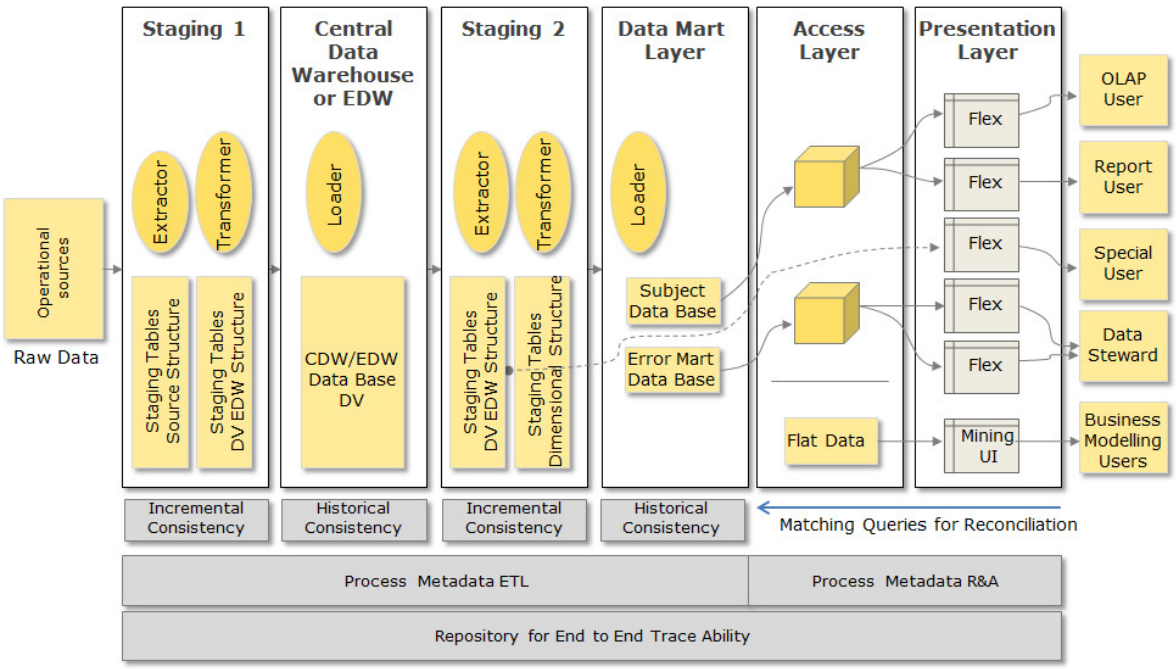

Abb. 1: Logica BI Framework Referenzarchitektur

Die Referenzarchitekur ist in mehreren Layern (Schichten) aufgeteilt, in denen die Daten transportiert werden:

Staging 1

> Speichern der Quellsystemdaten in Tabellen des Staging-Layers. Die Quelldaten werden 1:1 in der Tabelle gespeichert und anschließend für die Befüllung des EDWH in zusätzlichen Staging-Tabellen transformiert und gespeichert.

Enterprise Data Warehouse (EDWH)

> Die vortransformierten Daten werden im EDWH Datenmodell gespeichert. Hier werden die Daten persistent und nachvollziehbar gespeichert. Zur Anwendung kommen hier üblicherweise eine 3NF Modellierung oder Data Vault.

Staging 2

> Die EDWH-Daten werden für den Transport der Daten in den Data Mart Layer aus dem EDWH selektiert und in Staging-Tabellen gespeichert. Anschließend werden diese Daten in zusätzlichen Staging-Tabellen transformiert zur Befüllung des Data Mart Layers.

Data Mart Layer

> Speicherung der Daten in Dimensionen und Fakten unter Anwendung der Star Schema Modellierung. Faktentabellen, die sich mehrere Dimensionen teilen bilden hierbei eine Galaxie.

Access Layer

Der Access Layer bildet die Datengrundlage für das Reporting und liefert aggregierte Daten, die in Views oder Materialized Views abgelegt sind. Die Aktualisierung der Materialized Views erfolgt im Rahmen des ETL-Prozesses.

Presentation Layer

Hier werden die am Markt gängigen BI Tools für Adhoc- und Standardreporting-Abfragen genutzt. Eine wichtige Rolle spielt hierbei auch das Publishing der Daten und kann wie folgt durchgeführt werden:

- Email: Die periodisch erzeugten Berichte werden an bestimmte Mail-Gruppen gesendet
- Mobil: Zugriff auf Reports des BI-Tools mit entsprechenden Apps.
- Dashboards: Abrufen der Reports in bestimmten, vordefinierten Bereichen des BI-Tools

Datenmodellierung

In der Praxis ist es entscheidend, wo und wie das Datenmodell erstellt wird. Nur allzu oft sind verschiedene Versionen des Datenmodells an unterschiedlichen Speicherorten hinterlegt und niemand kann sagen, welche dieser Versionen nun die Aktuelle ist.

Für das Design der Datenmodelle und deren abhängigen Code-Artefakte (z. B. Datenbanktrigger) ist ein Coding Standard zu verwenden. Dieser schreibt den Entwicklern vor, welche Namensgebungen für bestimmte Datenbankobjekte und Code-Fragmente zu vergeben sind. Im Datenmodell-Design betrifft dies z. B. Tabellen, Indizes, (Mat-)Views, Foreign-Keys und Constraints. Im programmatischen SQL-Teil (PL/SQL, TSQL u. a.) sind dies z. B. Variablen, Typen sowie Prozeduren und Funktionen.

Des Weiteren ist vorzugeben, welche Art der Datenmodellierung für das Enterprise Data Warehouse (EDWH) und den Data Mart Layer zu verwenden sind.

Zur Anwendung kommt im EDWH zum Beispiel das DataVault-Modell. DataVault ist eine Modellierungsmethode, die die Daten aus der Geschäftssicht (Businesssicht) betrachtet und modelliert. Dabei liegt der Fokus auf der Erreichung einer hohen Ladeperformance sowie auf der Gestaltung eines flexiblen Systems, dass

- sich schnell und einfach anpassen lässt,
- skalierbar und
- nachvollziehbar ist sowie
- auditierbar bleibt.

DataVault liefert gleichzeitig eine implizierte Versionierung und Historisierung.

Im Data Mart Layer sollten Star-Schema bzw. Snowflake eingesetzt werden.

In allen Projekten sollte ein einheitliches Tool für die Datenmodellierung verwendet werden wie z. B. Sybase Power Designer. Das Tool sollte über ein zentrales Design-Repository inkl. Versionierung

verfügen zur Vermeidung von unterschiedlichen Versionen der Datenmodelle an verschiedenen Speicherorten.

Werden Testdaten in einem Datenmodell verwendet, so ist festzulegen, wer für die Anonymisierung der Daten verantwortlich ist. Hierfür gibt es verschiedene Szenarien – z. B. könnten die Daten schon anonymisiert vom Quellsystem geliefert werden, oder die Daten werden mittels einer Verschlüsselungslogik zur Ausführung anonymisiert.

ETL-Framework

Ein ETL-Framework enthält alle Komponenten, die in den jeweiligen Layer (z.B. Foundation Layer) zur Anwendung kommen. Dieses Framework bildet die Grundlage für die spätere Datenbewirtschaftung. Die Aufgabe liegt nun darin, für das jeweilige Projekt, ein an die fachlichen und technischen Anforderungen des Kunden angepasstes ETL-Framework zu definieren

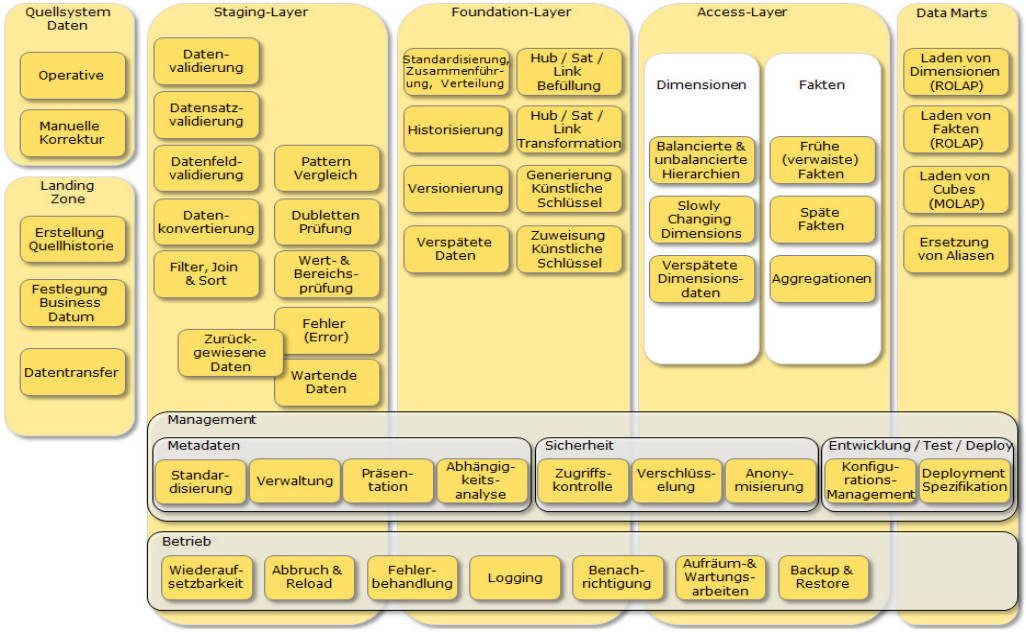

Abb. 2: Beispiel ETL-Framework

Welche einzelne Vorgänge/Module in dem jeweiligen Layer verwendet werden, ist abhängig von den funktionalen und nicht-funktionalen Anforderungen des Projekts sowie der Vorgaben anhand von Coding- und Naming-Standards. Diese Vorgaben betreffen die grundsätzliche Struktur der ETL-Prozesse und deren unmittelbaren Mappings. Die Namensvorgaben werden z. B. für Workflows, Interfaces, Mapping-Objekte wie Joiner oder Splitter sowie Agenten für die Job-Ausführung verwendet.

Die Vorgaben für die Datenbewirtschaftung der einzelnen Layer sind ebenfalls zu definieren. Diese betreffen z. B. die Art des Datentransports aus dem Quellsystem (Push, Pull) oder in welcher Form die Daten geliefert (Delta, Full) werden. Weitere wichtige Vorgaben sind die Wiederanlauffähigkeit von ETL-Prozessen sowie das Fehlerhandling der Prozesse bei fehlerhaften Daten.

Müssen innerhalb des ETL-Frameworks für die Entwicklung Testdaten bereitgestellt werden, so ist genau festzulegen, wann und wo die Anonymisierung der Testdaten stattzufinden hat. Die

Anonymisierung kann z. B. unter Verwendung einer bestimmten Logik durchgeführt werden, oder durch Produkte von Drittherstellern, die am Markt angeboten werden.

Ein weiterer wichtiger Punkt ist das Management und der Betrieb eines ETL-Frameworks. Hierfür bedarf es eines detaillierten Betriebshandbuchs sowie einer umfassenden Schulung und Einweisung der für den Betrieb abgestellten Mitarbeiter.

Fazit

Grundsätzlich wird empfohlen, ein Handbuch bzw. Leitfaden für DWH- und BI-Projekte auf Basis eines existierenden BI Frameworks, wie es z. B. Logica bietet einzuführen, um die unterschiedlichen Themen wie Datenmodellierung, Datenbewirtschaftung, Konfigurations- und Entwicklungsvorgaben abzubilden. Dadurch wird gewährleistet, dass neue Projektmitarbeiter wesentlich schneller eingearbeitet werden und ein allgemeines Verständnis der Umgebung geschaffen wird.

Des Weiteren dient das Handbuch bzw. der Leitfaden als Grundlage für zukünftige Neu- bzw. Weiterentwicklungen mit dem Ergebnis der verbesserten Wartung und Qualitätssicherung. Ein weiterer Vorteil liegt in der flexibleren Ressourcenplanung, da die Einarbeitung in das jeweilige Projekt aufgrund der Vorgaben verkürzt wird.

Insgesamt führen klare Vorgaben und Richtlinien, sowie deren Überwachung langfristig zu einer Kostenreduktion in der Neu- und Weiterentwicklung von BI- und DWH-Projekten.

Kontaktadresse:

Dirk Braunecker

Logica Deutschland GmbH & Co. KG | Now part of CGI

Am Limespark 2

D-65843 Sulzbach (Taunus)

Telefon: +49 151 4223 1010

E-Mail dirk.braunecker@logica.com | dirk.braunecker@cgi.com

Internet: www.logica.de | www.cgi.com