Data Warehouse Metadaten, wie und wozu?

Name des Autors
Firma
Standort

Schlüsselworte
Metadaten, OWB, Migration, Hub and Spoke, Bus-Architektur, Industrialisierung.

Einleitung
Metadaten sind die Daten über die Daten. Jedes ETL-Tool sammelt sie auf seine Weise, möglichst transparent. Wozu also selbst noch in dieser Sache aktiv werden? Wozu OWBSYS.ALL_-Views bemühen, wenn der OWB so eine schöne Anwendung darauf gebaut hat?

Kennen Sie die OWB-Repository-Objekte owbSys.cmpMap_v, owbSys.cmpMapOperator_v, owbSys.cmpMapAttribute_v, owbSys.cmpMapAttributeGroup_v, .cmpMapAttributeBindee_v, .all_iv_xForm_Map_Parameters, .owm_View_Utilities.Parameter_Expression2() / ..Operator_Type(), Sie führen Sie durch Ihr ganzes OWB-Mapping-Netz. Aber leider bleiben sie nur innerhalb des OWB.

Der OWB selbst bildet proprietär nur seine Daten, und nicht systemunabhängig alle Daten des real existierenden Data Warehouses ab. Verschiedene ETL-Tools konsolidieren die Daten verschiedener Quellsysteme über das Datenlager bis in Reports und Dashboards. Durchs ganze Unternehmen geht der DWH-Informationsfluss, über Werkzeug- und Datenbanktyp-Grenzen hinweg. Deshalb braucht es eine systemunabhängige Metadatenbeschreibung. Dort werden Zusammenhänge aufgedeckt und Kontrollmöglichkeiten gesucht. Rohdaten sollen durchgängig, verständlich und widerspruchsfrei zum Sprechen gebracht werden. Bei der abschließenden Klärung von solchen Fragen hilft nur die selbst geschriebene umfassende Metadatenbeschreibung.

Der analysierende Controller benötigt eine Quelle-Ziel-Zuordnung. Die „Landkarte“ legt offen, wie welche Information aus welchen Daten schrittweise resultiert, welche Controller-Logik die operative Unternehmens-Physik jetzt gerade meistert. Über die Metadaten lassen sich dezentrale, oft sogar redundante, widersprechende Logik-Einsätze identifizieren und zentralisieren.
Eine unternehmensweite Metadaten-Anwendung ist das Kontroll- und Steuerungselement für die Kontroll- und Steuerungselemente des Unternehmens. Sie ist das „Navi“ der Informationsdrehscheibe, das „Google-Maps des Unternehmens“. Sie schafft homogenen Überblick ohne Medienbrüche, revisionssicher und risiko-minimierend.

Was wäre eine Bibliothek ohne ihr Register, welche Bücher wann gekauft und an wen ausgeliehen sind? - Dasselbe, wie ein Data Warehouse (auf Deutsch: Daten-Lager zur Steuerung des Unternehmens) ohne Metadaten (Lager ohne EDV-System). Das ist dann ein meist mit viel Aufwand betriebener, zufällig einzelnen Zwecken unterworfener Auswertungs-Haufen ohne Dynamik und systematische Nutzung von Synergien.

Wie propagieren wir Tausch, Änderungen und Erweiterung der operativen Systeme industrialisiert in das Unternehmens-Datawarehouse? Im Vortrag wird eine mit Oracle und Java konzipierte Metadaten-Anwendung vorgestellt, die diesem Anspruch genügt. Ob Oracle-, DB2- oder SAS-SQL, geparst zerlegen wir Mengenoperationen in seine Select-, From-, Where-Elemente, Joins, Funktionen und Attribute, um die Datenfluss-Bedingungen der durchgekommenen Daten lückenlos offenzulegen.
Per Reverse-Engineering reichen wir Änderungen bis ins Ziel automatisiert weiter.

Die effiziente Weiterentwicklung des DWH durch Metadaten

Metadaten sind ein weites Feld: Wir unterscheiden in fachliche, technische und organisatorische Metadaten. Fachliche Metadaten sind beispielsweise Tabellen und Felder-Mappings, Kennzahlen-Algorithmen, die Bedeutung der Quelldaten und Aktualität und Relevanz der Berichte. Technische Metadaten sind beispielsweise Ladezeitpunkte und abgeholte und geladene Datensatz-Anzahlen, die Lade- und Leseprotokolle. Organisatorische Metadaten sind beispielsweise Organigramme und Verantwortlichkeiten, die Data Ownership und Stakeholder, Urlaube und Vertretungsregelungen, Tools und Strategien über die Zeit.

Beispielhaft fokussiere ich nun auf die fachlichen Metadaten: Den Datenfluss-Datamart. Das ist die Abbildung des fachlichen Informationsflusses, wie er technisch tatsächlich passiert und sich ändert. Wir bilden die Felder, die im Berichtswesen gebraucht werden, und ihre Quellen in den operativen Systemen im kompletten Datenfluss ab.

Das folgende Schaubild zeigt, wie sich die Anwendung im gesamten DWH-Datenfluss einordnet. Durch die Transparenz aller Verarbeitungsschritte wird das saubere Zerlegen des ETL in 1. Beschaffung, 2. Herstellung und 3. Auslieferung systematisch. Diesen Dreischritt kennen wir in der produzierenden Industrie schon lange. Er bringt große Effizienz-Vorteile auch im ETL des DWH als Wissensproduktion und Informationsdrehscheibe.

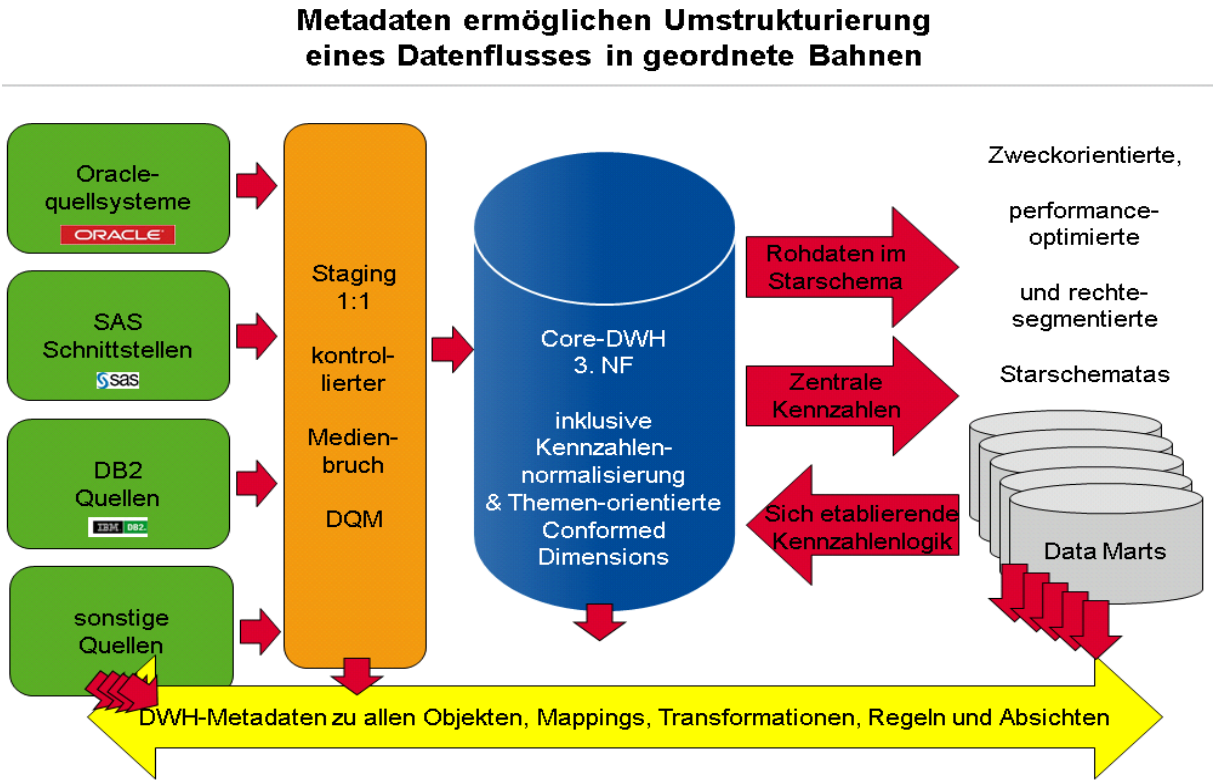

Abb. 1: Mit Metadaten zur klassischen DWH-Architektur

Wir ziehen die fachlichen Metadaten über Database-Links aus beispielsweise den operativen Schnittstellentabellen der DB2 (sysibm.sys…, all_tab_columns…) und historisieren den Alt-Stand. Dann gleichen wir den aktuellen Stand der Schnittstellen mit den Create Tables des letzten Release-Standes ab. Damit lässt sich das Delta ermitteln, was in den Schnittstellen-Tabellen an neuer Information im Angebot steht und im DWH noch fehlt, verändert oder überholt ist.

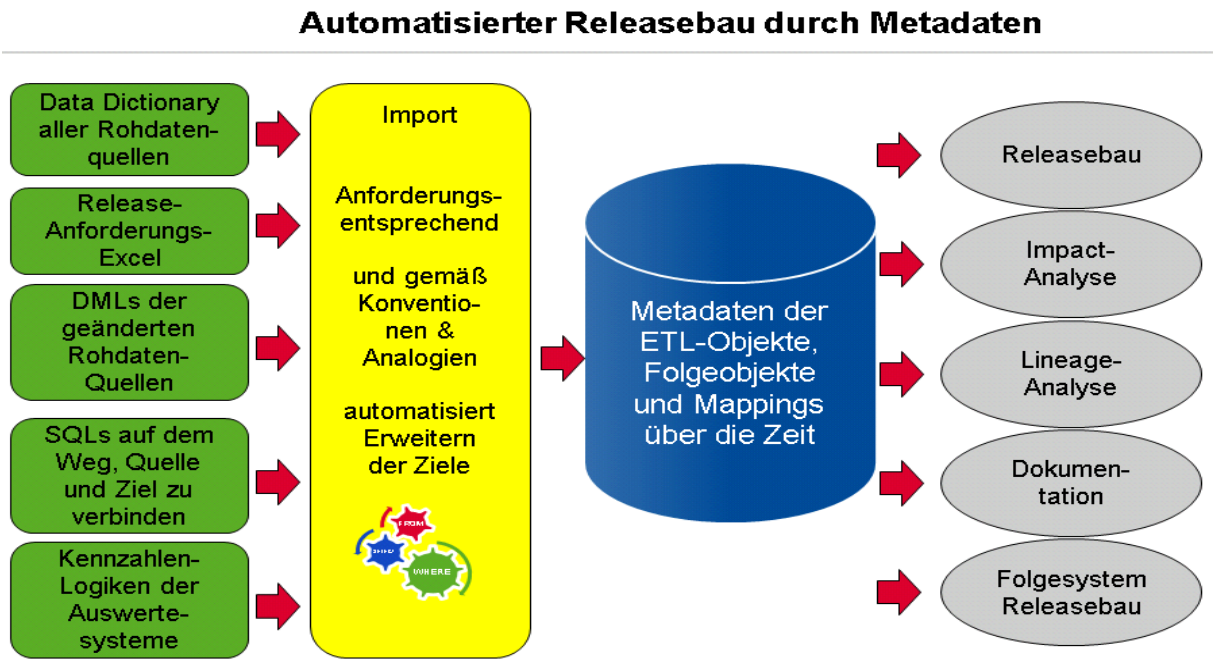

Abb. 2: Der Datenflussdatamart

Die automatisierte Erweiterung des kompletten nun folgenden Datenflusses aus den Neuen und geänderten Datenbank-Elementen ist somit einfach: Create Tables (für einen etwaigen Neuaufbau des DWHs und das Anlegen neuer Tabellen), Alter Tables (für die produktive Datenbank-Änderung), Create Views und die Mappings lassen sich nach den erprobten Konventionen und Vorgabe der Betriebsorganisation folgerichtig dem Änderungs-Volumen der Quell-Schnittstellen entsprechend erweitern.

Manche Abteilungen verfügen über große fachliche Mapping-Excel, welche Quell- und Ziel-Tabellen verbinden. Um sich unabhängiger zu machen von zeitintensiver Dokumentation, ist es hilfreich, die in der Regel greifbaren SQLs der Daten-Weitergabe-Schritte (create-tables, -views, -mappings, -reports…) zu parsen. So lassen sich auch überholte Berechnungsformeln per Select finden und für alle betroffenen Mappings bzw. SQLs auf einmal und gleichartig korrigieren. Die Berechnung einzelner Attribute kann dann bis zu ihrem Ursprung zurückverfolgt werden. Die Analyse gelingt damit auch innerhalb von Ausdrücken, sodass Berechnungen vollständig nachvollzogen werden können.

Das bringt Kostenersparnis und Entlastung der ETL-Entwickler durch reduzierten Analyseaufwand in Nerv tötenden Wiederholschleifen. Die ETL-Entwickler widmen sich dann den sinnvollen Aufgaben, die immer genug bleiben für ein erfülltes Arbeitsleben. Nicht jedes mögliche Sprach-Element, das in SQL vorkommen kann, ist jetzt schon von der konkreten Metadaten-Anwendung abgedeckt. Das hat noch niemand bezahlt. Die häufigsten aber schon, und die Selteneren (Pivot, Unpivot…) lassen sich auch von Hand in das Datenmodell parsen, falls die Regel des Automatismus noch nicht ganz klar ist. Per Reverse-Engineering von der Metadaten-Visualisierung in den jeweiligen Objekt-Typ reichen wir Änderungen bis ins Ziel automatisiert weiter.

In entsprechenden Oracle-Tabellen des Metadaten-Datenmodells stehen, verbunden über die Parent-Child-Beziehung neben den System-Namen, Tabellen-Namen, View-Namen, Mapping-Namen, Attribut-Namen auch die Select-Ebenen, bei geschachtelten Selects mehrfach, mit deren Attribut-Namen, Funktionen, Joins und den anderen SQL-Sprachelementen.

Parsen aller SQLs über die Keywords (z.B.: select, from, where, as, join...)
Ergebnis: Datenfluss der Attribute durch Tabellen, Filter und Funktionen

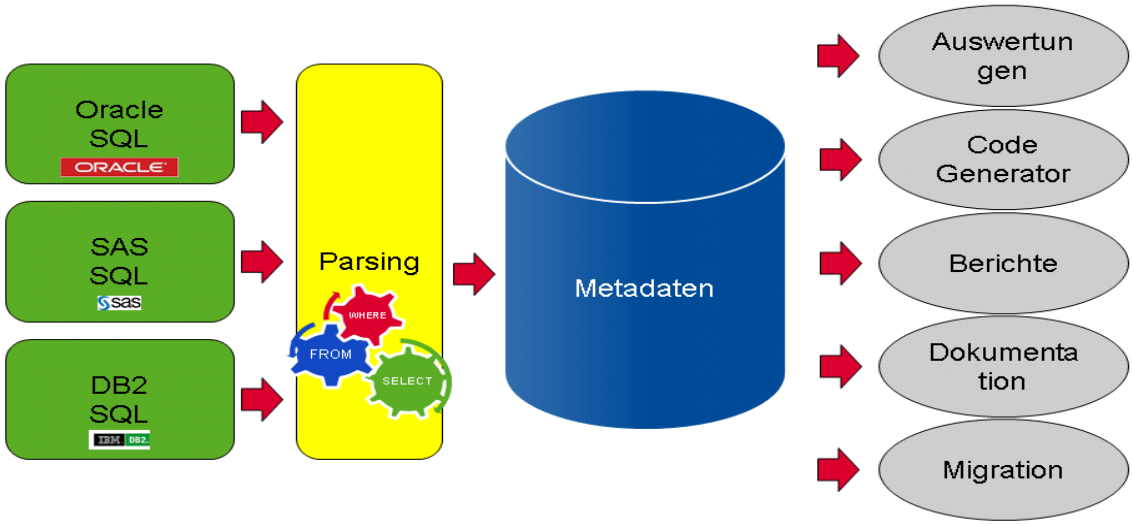

Abb. 3: Das SQL-Parsen im Datenfluss

Alle Datenfluss-Elemente, gleich welcher Ebene, wandern also in eine Tabelle. So wird das Filtern nach Attributen und Expressions einfach und umfassend. Wir ermöglichen die graphische Darstellung analog zu der von ETL-Mappings und unterstützen damit die industrialisierte Attributerweiterung.

Parsen eines Select-Statements (Beispiel)

```
SELECT A.FGABTID, B.KPIWERT AS ZIELPROD
FROM TBS_AASAS29 A LEFT JOIN TBS_AASAS34 B ON A.FGABTID = B.FGABTID
WHERE TRUNC (A.GDAT) <= TRUNC (SYSDATE);
```

Kennung	Tiefe	Objekt-Typ	Objekt-String
	0	CREATE-GENERAT	CREATE OR REPLACE FORCE VIEW BIBS.VBS_ORG_BMKPI (FGABTID,ZIELPROD) AS SELECT ...
S1	1	SELECT-OBJEKT	SELECT A.FGABTID,B.KPIWERT AS ZIELPROD FROM TBS_AASAS29 A LEFT JOIN TBS_AASAS34 B ...
S1-	2	ATTRIBUT-LISTE	A.FGABTID,B.KPIWERT AS ZIELPROD
S1-	2	QUELLOBJEKT-JOIN	TBS_AASAS29 A LEFT JOIN TBS_AASAS34 B ON A.FGABTID = B.FGABTID
S1-	2	WHERE-BEDINGUNG	TRUNC (A.GDAT) <= TRUNC (SYSDATE)
S1-A1	3	ATTRIBUT-STRING	A.FGABTID
	4	ATTRIBUT-EXPRESSION	A.FGABTID
	4	ATTRIBUT-NAME	FGABTID
S1-A2	3	ATTRIBUT-STRING	B.KPIWERT AS ZIELPROD
	4	ATTRIBUT-EXPRESSION	B.KPIWERT
	4	ATTRIBUT-NAME	ZIELPROD
S1-J1-	3	QUELLOBJEKT-STRING	TBS_AASAS29 A
	4	QUELLTABELLE	TBS_AASAS29
	4	QUELLOBJEKT-ALIAS	A
S1-J2-	3	JOIN-STRING	LEFT OUTER JOIN
S1-J3-	3	QUELLOBJEKT-STRING	TBS_AASAS34 B
	4	QUELLTABELLE	TBS_AASAS34
	4	QUELLOBJEKT-ALIAS	B
S1-J4-	3	ON-BEDINGUNG	A.FGABTID = B.FGABTID

Abb. 4: Das Parsen eines Selects bis auf Attribut und Tabellen-Ebene

Das folgende Bild zeigt das hier oben genannte einfache SQL graphisch aufbereitet vergleichbar einem OWB-Mapping. Die blau leuchtenden Punkte zeigen Join-Attribute, Group-bys-Attribute und/oder Filter-Attribute, deren konkrete Expression jeweils im rechten unteren Feld erscheint.

Die Erweiterung der Input-Felder um alle existierenden Felder der realen Schnittstelle zur transparenten Erweiterung des Selects leistet die Anwendung und unterstützt so das Reverse Engineering.

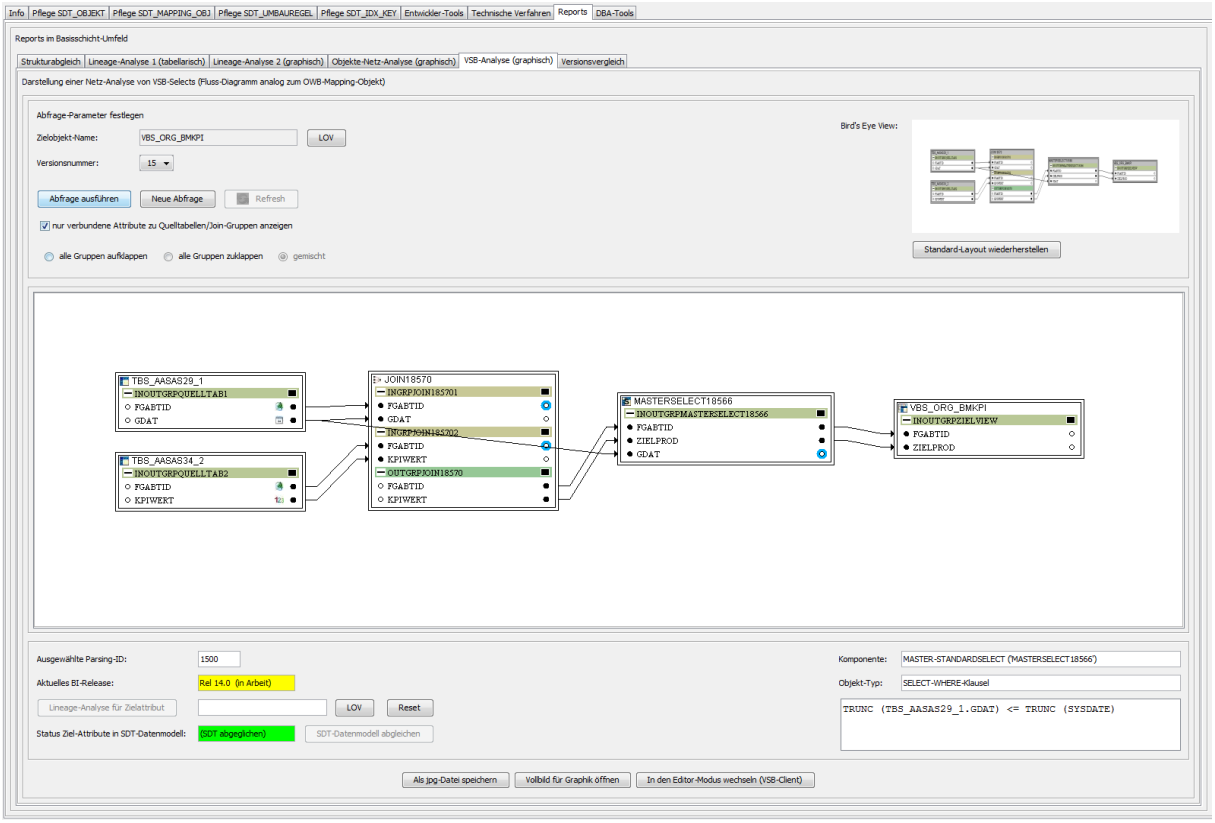

Abb. 5: Graphische Oberfläche des SQL-Mappings für Änderung und Reverse-Engineering

Migrationen vom einen ETL-Tool (den OWB wird es nicht mehr so lange geben…) in ein anderes werden durch so einen Metalayer kontrollierbar und automatisierbar. Denn die resultierenden SQLs werden auf ihre Mengenoperation hin reduziert und damit mess- und vergleichbar. Also weitgehend übertragbar, wenn man die Besonderheiten des neuen und des alten Mengenoperations-Dialekts versteht.

Warum sollte man Mengen-Operationen nicht mengenorientiert abarbeiten? Warum sollte man die OWB-Package Erstellung und Pflege nicht automatisiert betreiben, wo das möglich ist? Komplizierte Fragestellungen kann man immer noch von Hand machen, aber das ritualisierte Ergänzen der meisten Packages um neu hinzugekommene Felder ist öde und fehleranfällig, wenn man jedes von Hand macht. Die Fehler-Quote unseres Referats bei der Release-Entwicklung geht inzwischen durch Unterstützung des Metadaten-Tools gegen Null. Das waren mal viele Fehler bei einem Bruchteil des heutigen Änderungsvolumens.

Die fachliche Metadaten-Anwendung besitzt die folgenden etablierten Prozesse und Funktionalitäten:

- Automatisierte Befüllung der Metadaten im DWH-Entwicklungsprozess
- Nutzung der Metadaten-Anwendung als Generator-Tool
  für Tabellen-Create- , -Alter-, und -View-Skripte und für Standard-Mapping-Typen

- Unterstützung bei Struktur-Abgleichen zwischen Datenbank-Ständen
- Nutzung der Metadaten-Anwendung als Generator-Tool für normierte Install-Skripte
- Impact- und Lineage-Analyse ausgehend von jeder Tabelle und/oder jedem gewünschten Einstiegs-Feld im abgebildeten Gesamt-Datenfluss
- Anbindung von Quellsystem-Data-Dictionary und Domän-Werten von Lookup-Tabellen
- Parsen von ETL-SQLs verschiedener Hersteller
- Reverse-Engineering von SQLs durch Import des DMLs geänderter Rohdaten-Quellen und die Veränderung triggernde Oberfläche der Metadaten-Anwendung

Aufklärung über die Aufklärung ist notwendig

Die Metadaten-Anwendung ist ein Instrument, um das DWH immer wieder auf den Prüfstand zu stellen, und zu gewährleisten, dass es gemäß der Geschäftsprozesse aktualisiert wird. Unsere im BI-Team beim Kunden entwickelte Anwendung ist ein Tool, dies konsistent, schnell, versioniert und in stets gleicher Qualität zu bewerkstelligen. Es ist ein Meta Data Warehouse.

Durch die Konzentration des Tools auf den kleinsten gemeinsamen Nenner aller Anwendungen im Gesamtsystem, dem SQL, handelt es sich bei der besprochenen Metadaten-Anwendung nicht um ein Konkurrenz-Produkt zum OWB, oder anderer beliebiger ETL-Tools. Die gesammelten Metadaten unterstützen nur bei der effizienten Nutzung von ETL-Tools, wie dem OWB oder anderen, indem sie deren Scripting-Fähigkeiten (OMBplus beim OWB) via SQL parametrisiert zur aktualisierten ETL-Mapping-Erstellung nutzt. Die Heterogenität der Unternehmensrealität wird akzeptiert und mit SQL überwunden.

Die Metadatenfunktionalität Impact- und Lineage-Analyse erscheint mir die Fähigkeiten des OWB aber schon deshalb zu übersteigen, weil die Metadaten-Anwendung explizit nicht proprietär arbeitet und grundsätzlich jeden SQL-Dialekt akzeptiert. Die Sprach-Elemente des SQL sind doch sehr überschaubar. Die Syntax-Unterschiede zwischen DB2 und Oracle-SQL sind es auch.

So kann akzeptiert werden, dass verschiedene Systeme mit ihrer jeweiligen Struktur nebeneinander existieren, weil mit Hilfe der Metadaten trotzdem ein überprüfbares, konsolidiertes, aktuelles Gesamtbild entsteht. Die kulturelle Vielfalt in jedem Unternehmen wird so zum lesbaren Reichtum. Es ist wichtig, dass die kleinen Einheiten im System passgenau ineinander greifen. Das können sie umso besser, je transparenter das System im Ganzen ist. Genau das ist der Ansatz von Metadaten: Es ist das Kontroll- und Steuerungselement für die Kontroll- und Steuerungselemente des Unternehmens. Auf den Punkt gebracht könnte man sagen: **Effizienz im Kleinen fordert Transparenz im Großen.** Seien es die kleinen Felder in der großen Datenbank oder der einzelne Mitarbeiter im Gesamt-Unternehmen. Transparenz schafft Vertrauen und Durchschlagskraft.

Ich und meine Kollegen von der metafinanz, wir freuen uns auf einen fruchtbaren fachlichen Austausch, auf Ihre Fragen und Erfahrungen. Gerne auch per Mail.

Kontaktadresse:
Clemens Albrecht
Metafinanz Informationssysteme GmbH
Leopoldstrasse, 146
D-80804 München

Telefon: +49 (0) 89-360 531 5167
Fax: +49 (0) 89-360 531 115
E-Mail clemens.albrecht@metafinanz.de
Internet: www.metafinanz.de