Repository als Metadaten-Plattform

Dr.Reinhold Thurner
Markus Rahlff

Schlüsselworte:

Metadaten, Repository, Modellierung, CMDB, Entity-Relationship-Modell, Ontologien

Einleitung

In der IT verwaltet zunehmend umfangreiche Bestände von Daten, die für die Prozesse der IT selbst benötigt werden. Investitions-Vorhaben, Projekte, Anwendungen, Prozesse, Services, Programme, Schnittstellen, Datenbanken, Modelle, Server, Lizenzen u.v.a.m. zeichnen sich durch eine Vielzahl unterschiedlicher Eigenschaften und ein feinmaschiges Netz von Beziehungen aus.
Für eine integrierte Speicherung dieser Informationen wird ein DBMS mit einem Entity-Relationship-Attribut-Modell - ein Repository - eingesetzt.

1. Die Metadaten - die Daten der IT

Unternehmens-Daten: Moderne Unternehmen sind heute mehr denn je abhängig von der ununterbrochenen Verfügbarkeit von aktuellen Daten. Diese Daten werden in Datenbanken gespeichert und stehen für die Verarbeitung durch Geschäftsanwendungen zur Verfügung. Diese Anwendungen sorgen dafür, dass die Daten sicher gespeichert sind und ein umfassender und kontrollierter Zugriff auf die Daten bereitgestellt wird.

Management der IT-Daten: Wie werden aber die Daten gemanagt, die von der IT - der Führung, der Entwicklung und dem Betrieb - selbst für ihre eigenen Prozesse benötigt werden? Wo wird festgehalten, welche Applikationen, Programme, Datenbanken, Dateien und Datenfelder im Unternehmen vorhanden sind (und auch benutzt werden)? Welche Dokumente geben darüber Auskunft, was sich hinter dem Feld «Lieferdatum» verbirgt, das in einer Liste ausgewiesen wird? Und in welchen anderen Listen oder Datenbanken wird dieses Feld – wenn möglich mit der gleichen Bedeutung aber anderer Bezeichnung auch verwendet?

Vorhanden aber nicht auffindbar: Die meisten dieser Daten sind vorhanden – zum Teil in Dateien, Tabellen, Dokumenten, Datenbanken, zum größten Teil allerdings auch nur versteckt in Programmen oder Dokumenten oder gar nur in den Köpfen der Mitarbeiter: Der Grund dafür ist nicht etwa, dass Informationen absichtlich zurückgehalten werden. Es liegt meist daran, dass keine einfache, einfach recherchierbare und zentrale Ablage für diese Informationen und Zusammenhänge, für das viele «Gewusst-Wo» vorhanden ist.

Komplexität der Werkzeuge: Viele "IT-Anwendungen" und Werkzeuge sind nach dem klassischen Silo-Modell aufgebaut, besitzen eigene Datenbestände und tauschen Daten mit anderen Werkzeugen über Punkt-zu-Punkt-Verbindungen aus. Dies wird zusätzlich dadurch erschwert, dass die Daten nicht unmittelbar vergleichbar sind, weil meist gemeinsames Datenmodell fehlt. Das Ziel einer Prozess- und Service-Orientierten IT ist mit einer solchen Dateninfrastruktur schwerlich erreichbar.

Repository als Plattform für die IT-Daten: Um diese Probleme zu lösen muss auch die IT ihre eigenen Daten systematisch verwalten. Es wird daher eine Datenplattform benötigt, die spezialisiert ist für die Speicherung, Verwaltung und Publikation von solchen umfangreichen und komplex vernetzten Beständen. Für diese Aufgabe ist ein DBMS (Datenbank-Management-System) mit einem Entity-Relationship-Modell (also nicht einem Relationalen Modell) optimal geeignet. Da es keine einheitliche Lehrmeinung darüber gibt, was ein Repository "ist" bezeichnen wir im Folgenden ein solches System als Metadaten-Repository oder kurz als Repository.

Metadaten = Informationen der IT

Unter Metadaten verstehen wir heute mehr als nur «Daten über Daten» nämlich alle Daten ÜBER die Informationssysteme, die benötigt werden um die Informationssysteme zu planen, zu entwickeln oder zu beschaffen und für die Kunden zu betreiben.

Im Gegensatz zu den großen eher homogenen Anwendungsdaten (eine Kundendatenbank, Auftragsdatenbank etc.) sind Metadaten feingranular und hochgradig vernetzt: Metadaten sind strukturiert in viele unterschiedliche Entitätstypen (hunderte) mit vielen Attribut- und Beziehungstypen (hunderte), wobei alle Typen selbst wieder durch TypAttribute zu beschreiben sind (Ontologien). Pro Typ gibt es zwar eher relativ wenig Instanzen (einige wenige bis einige 10.000), einige davon jedoch in Versionen und Konfigurationen. So werden für die Steuerung der Veränderungs- und Planungsprozesse (Change-Mangement) Metadaten nicht nur in einem (aktuellen) Stand sondern in den verschiedenen Zuständen z.B. in_Produktion, in_Abnahme, in_Entwicklung, in_Planung , also in Versionen und Konfigurationen benötigt.

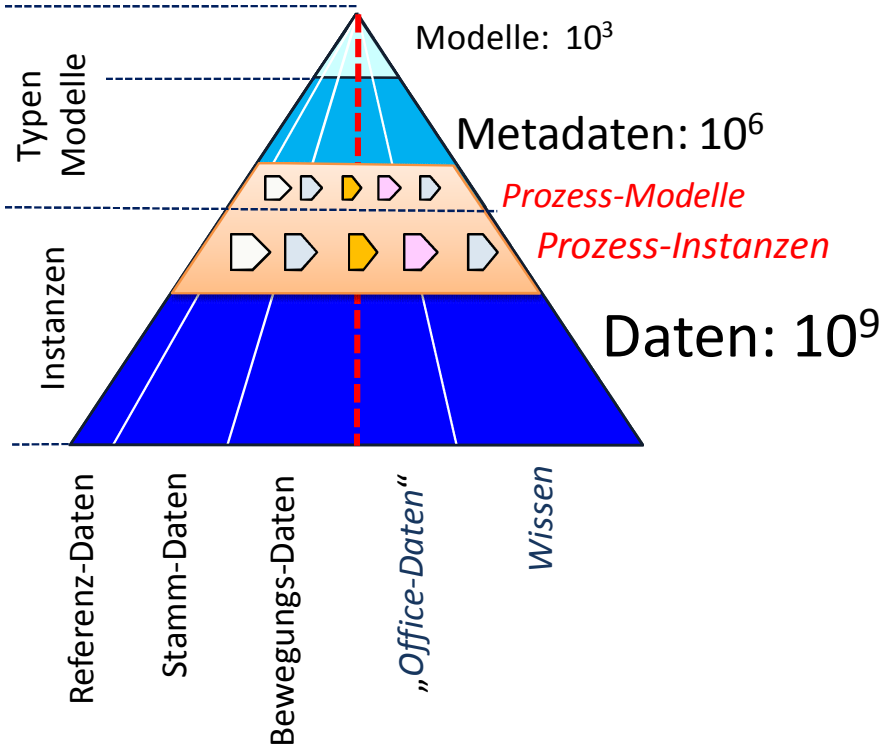

Abb. 1: Daten und Metadaten: Strukturen und Mengen

2. Repositories im Einsatz

Repository-basierte Anwendungen und Werkzeuge nutzen das Repository als gemeinsame Datenplattform. Jede Anwendung "sieht" jeweils nur einen Ausschnitt aus dem gesamten Datenuniversum und mit den Einschränkungen, die im Rechte-System des Repository definiert sind. Die ausführliche Beschreibung der Metadaten durch TypAttribute ermöglicht den Einsatz von generischen Werkzeugen, die direkt durch die Metadaten gesteuert sind. Solche "modell-getriebene" Anwendungen passen sich flexibel auch an Erweiterungen und Änderungen des Repository an. Beispiele dafür sind u.a. der ModellEditor, der MetaEditor, der Anschluss von Office-Tools (Excel) oder Auswertungssystemen (BIRT).

Damit hat der Anwender eines Repository die Möglichkeit auf einfache Weise mit Standardwerkzeugen die Auswertungen und Datenausgaben selbst zu erstellen, die er effektiv benötigt. "Fertige" Werkzeuge mit weitgehend fest vorgegebenen Funktionen sind fehl am Platz in einem Gebiet, das noch so wenig standardisiert ist wie die IT.

Benutzer eines Repository

Das Repository wird von Benutzergruppen mit sehr unterschiedliche Informationsbedürfnissen bezüglich Inhalt und Detaillierungsgrad verwendet, die sich jedoch auf Ausschnitte der gleichen Daten beziehen. Ein Repository verwaltet die Daten auf verschiedenen Ebenen und stellt den Benutzern individuelle und konsistente Sichten (Views, Subschemas) auf die Daten zur Verfügung:

- **Info-User:** Benutzer, die rasch und auf einfache Weise auf Informationen zugreifen möchten.
- **Daten-Download:** Benutzer, die Daten für die Weiterverarbeitung (meistens in Office-Anwendungen) benötigen.
- **Andere IT-Systeme:** Verwendung ausgewählter Daten mit der Exportfunktion des Repository, aber mit einheitlichem Datenmodell.
- **Entwickler von eigenen Werkzeuge** nutzen den direkten Anschluss an das Repository über das API.
- **Anwendungs-Entwickler:** Recherchen über bestehende Anwendungen und Pläne, die Dokumentation von Anwendungen, die Erstellung von Auswertungen oder Berichten.
- **Administratoren, Revisoren:** Nutzen automatisierte Berichtsysteme, die Daten aus dem Repository auswerten.
- **IT-Betrieb:** nutzt das Repository als CMDB oder MetaCMDB in dem verdichtete Informationen aus den verschiedenen Managementsystemen zusammen geführt werden.

3. Das Repository - ein DBMS für Metadaten

Ein Repository ist ein Datenbank-Management-System für die Verwaltung eines umfangreichen und ausführlich dokumentierten Datenmodells und von umfangreichen Beständen von vernetzten, versionierten Metadaten (den Instanzdaten).

Das (Meta-) Datenmodell eines Repository ist ein erweitertes Entity-Relationship-Modell und bildet die Struktur vernetzter Daten direkt ab. Der Vergleich mit dem (Meta-) Modell eines RDBMS zeigt die Unterschiede im Modell und in der Speicherung der Instanzen. Attribut-Werte werden direkt im Repository gespeichert oder als "externe Attribute" in anderen Datenhaltungen angesprochen.

Mengen und Häufigkeiten: Ein Repository als Integrationsplattform speichert Modelle mit tausenden von Elementen und Millionen von Instanzen. Wenn der Umfang eine gewisse Größe (nn GB) überschreitet verwendet man mehrere spezialisierte Repository-Instanzen, die jedoch auf einem gemeinsamen Datenmodell basieren. Solche Repositories bilden einen Verbund von federated Repositories: Anwendungen können parallel auf mehrere Repositories zugreifen.

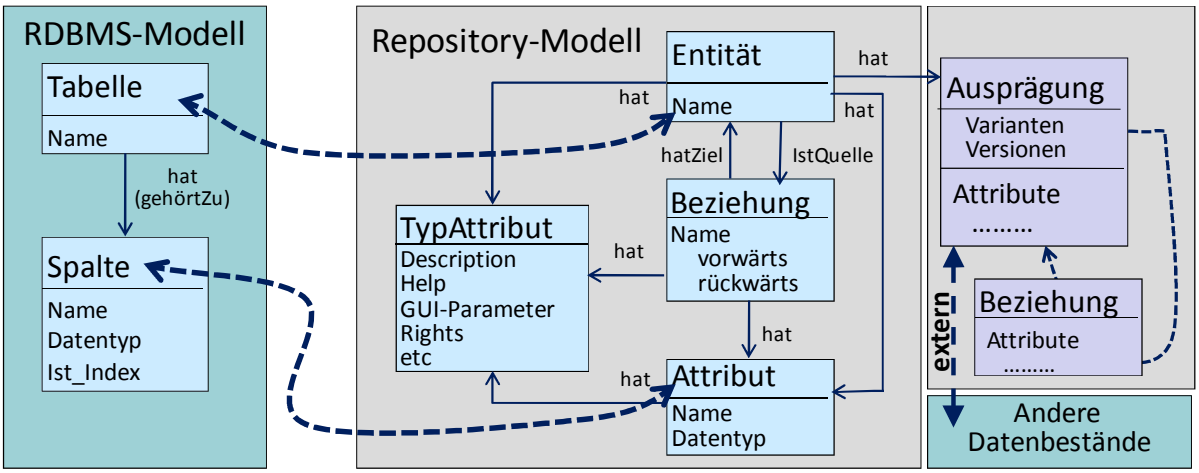

Abb. 2: MetaModell eines Repository im Vergleich mit einem RDBMS

Multi-User-Fähigkeit: Die Informationen eines Repository werden im Lese- und im Update-Modus von mehreren Benutzern oder Anwendungen angesprochen. Ein Repository muss daher transaktionsgeschützt Updates verarbeiten. Die Zugriffsrechte der Benutzer sind im Repository definiert.

Repository-nahe Werkzeuge: Einzelne generische Werkzeuge gehören zu einer Repository-Plattform: Modell-Manager, Daten-Editor, Import/Export, Auswertungswerkzeuge etc. Diese Werkzeuge sind direkt vom Repository-Modell gesteuert.

API - das Application Programming Interface: Entscheidend für die Erstellung von Repository-basierten Anwendungen ist das API. Eine leistungsfähige Funktionsbibliothek stellt Methoden für den Zugriff, die Recherche und die Navigation im Repository zur Verfügung. Über das API kann auch das Modell abgefragt oder auch erweitert und angepasst werden. Anwendungen können ein Repository als "embedded Database" so einsetzen, dass das Repository selbst für Anwender völlig transparent bleibt.

4. Aufbauen eines Repository

4.1 Erstellung des Modells

Das (Anwendungs-) Modell bildet den Kern einer Repository-Anwendung. Dieses (konzeptuelle) Modell soll die zu verwaltenden Konzepte möglichst unmittelbar und verständlich abbilden. Das betrifft auch die Bezeichnung der Entitäten, der Beziehungen und der Attribute. Das Modell wird mit Hilfe des Modell-Editors erstellt, gewartet und dokumentiert. Nicht der erste Entwurf eines Modells ist entscheidend sondern die Fähigkeit das Modell flexibel zu erweitern.

- **Der MetaDictionary** (Glossar) bildet die Grundlage des Modells. Im MetaDictionary werden die Entiätstypen, die Beziehungstypen und die Attributtypen mit ihren MetaAttributen beschrieben.
- **Das Masterschema** beschreibt das gesamte logische Informations-Netz mit den Begriffen des MetaDictionary. Im Masterschema werden den Entitätstypen Attribute zugewiesen und sie werden über Beziehungstypen miteinander verknüpft. Die MetaAttribute der Elemente (Attribute und Beziehungen) werden, wo erforderlich dem Kontext entsprechend angepasst. Das Masterschema kann sehr gross und für einzelne Benutzer zu unübersichtlich werden.
- Mit Hilfe von **SubSchemas (Views)** bildet man Teilsichten als Ausschnitte aus dem Repository und versieht sie mit Zugriffsrechten. Damit kann man einzelnen Benutzergruppen Views für ihre spezifischen Bedürfnisse zur Verfügung stellen.

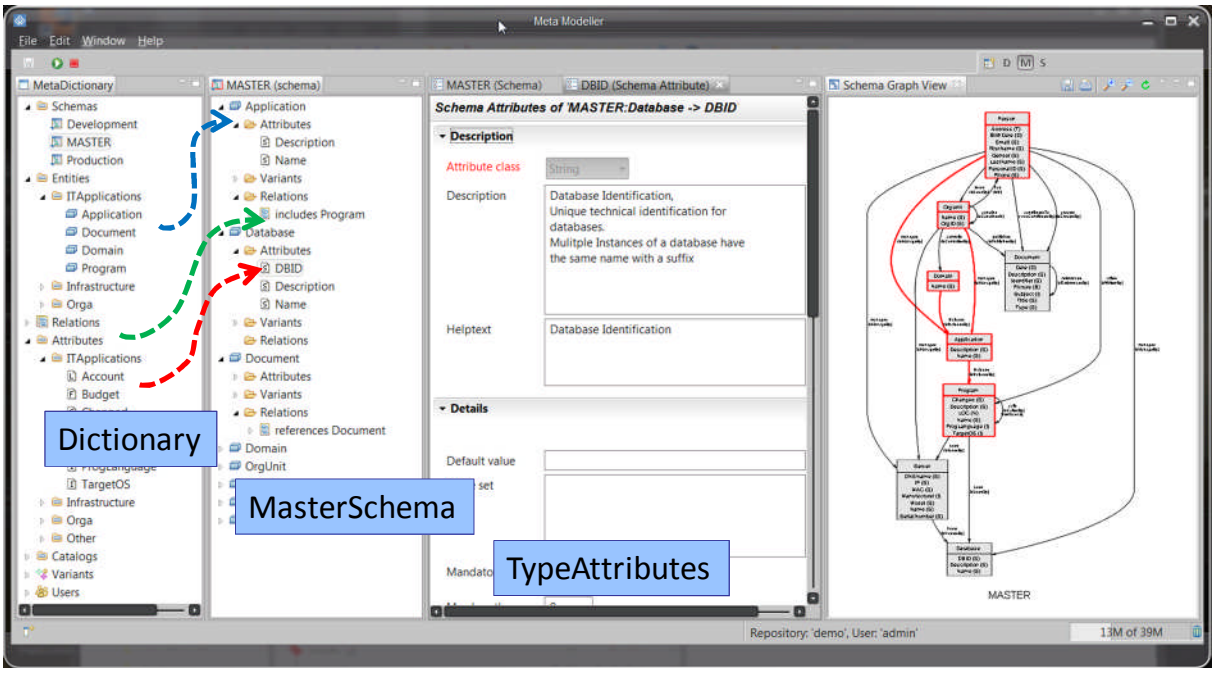

Abb. 3: Erstellen eines Modells im Repository

4.2 Testen eines Modells mit Daten

Ein Modell besteht seine Bewährungsprobe indem Daten in das Repository geladen werden. Hier zeigt sich ob die Modell-Abstraktion mit der Wirklichkeit übereinstimmt, ob die geplanten Informationen tatsächlich in der gedachten Qualität und Struktur zur Verfügung stehen. Testdaten können direkt über

den Dateneditor in das Repository eingegeben werden. Vorhandene Testdatenbestände können auch über die Importwerkzeuge (XML, XLS, CSV) oder mit eigenen Anwendungsprogrammen geladen werden.

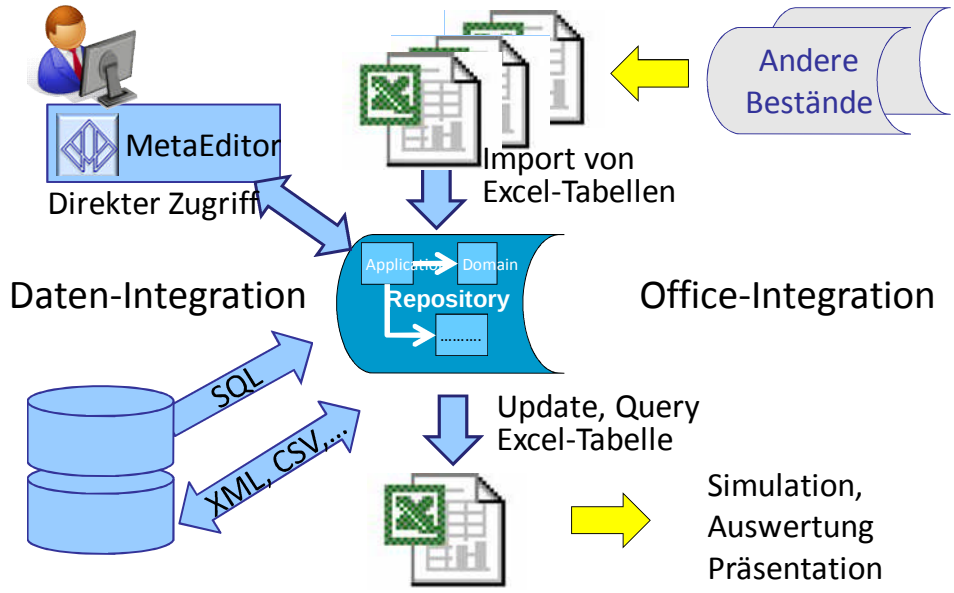

Abb. 4: Daten über Schnittstellen importieren und exportieren

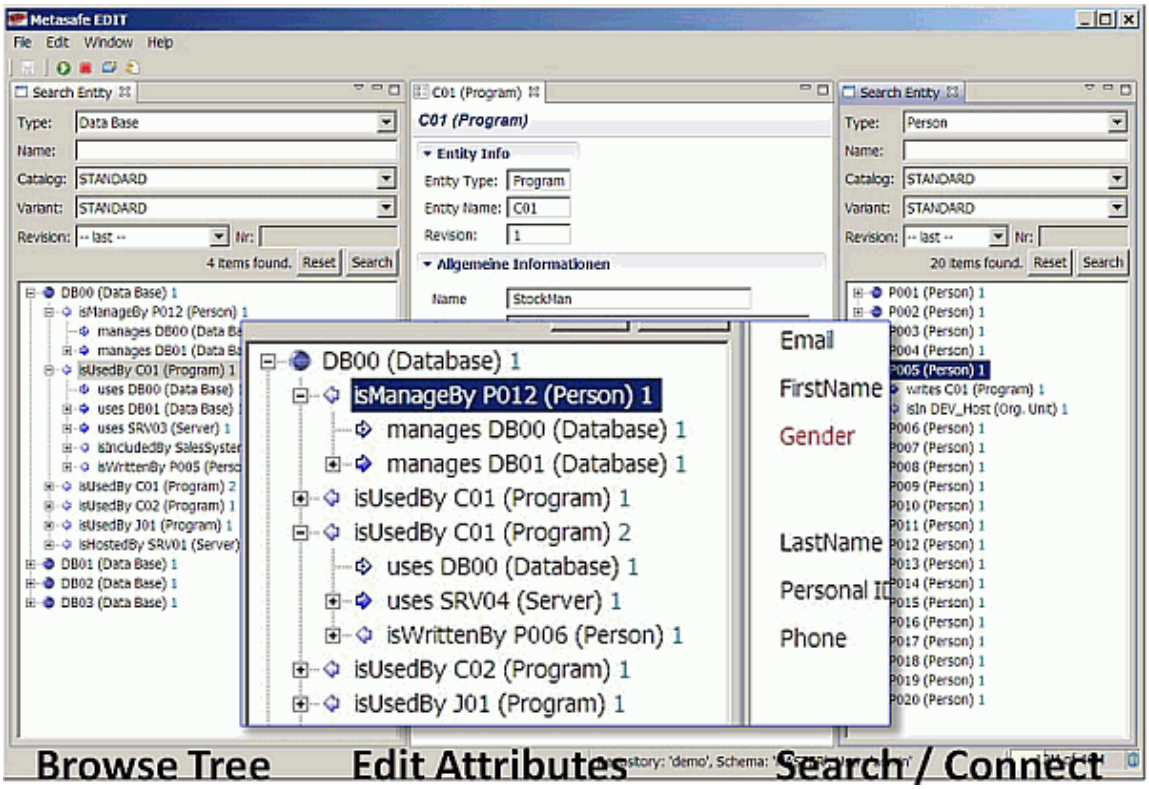

Abb. 5: Editieren der Metadaten mit einem modell-gesteuerten Browser

4.4. Daten auswerten mit BIRT

Das Metasafe Repository verfügt über eine Schnittstelle zu BIRT dem universellen Reporting- und Informationssystem. Mit BIRT können flexible Auswertungen und Druckausgaben designt und im Repository hinterlegt werden. Mit Parametern gesteuert können Auswertungen in PDF, Excel, Word generiert werden. BIRT greift über den Metasafe-ODA-Driver direkt auf das Repository zu. Damit stehen alle Funktionen von BIRT für Auswertungen zur Verfügung.

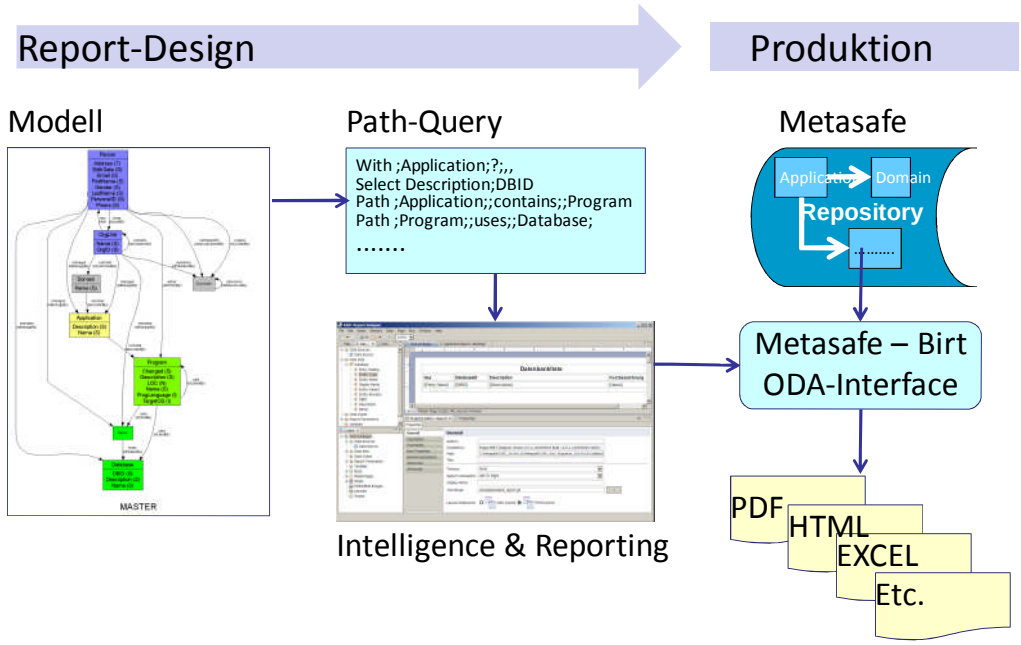

Abb. 6: Auswertungen mit BIRT

4.5 Ausbau des Repository:

Neue Anforderungen werden immer wieder an das Modell herangetragen und neue Entitäten, Attribute, Beziehungen müssen aufgenommen werden. Solche Erweiterungen können an einem bestehenden mit Daten gefüllten Repository mit Hilfe des Modell-Editors vorgenommen werden. Ein Umladen der Datenbank ist nicht erforderlich - im Gegenteil: Nach Ergänzung des Modells können sofort Daten für die neuen Elemente geladen und genutzt werden.

5. Ein Beispiel aus der Verwaltung von Datenbanken:

Das Beispiel (anonymisiert) ist in Planung bei einem Anwender, der mehrere tausend Server in Betrieb hat auf denen mehrere tausend Datenbanken mit unterschiedlichen DBMS installiert sind. Das (hier in Ausschnitten) dargestellte Gesamtkonzept soll in mehreren Schritten umgesetzt werden.

(1) Die erste Schicht bilden zunächst die klassischen Aufgaben der Datenverwaltung: Save/Restore sowie die Bereitstellung von Testdatenbeständen für mehrere Testdomains.

(2) Als Grundlage für die weiteren Arbeiten werden Informationen über die Datenbank-Instanzen sowie die Datenbank-Schemas aus den Datenbank-Katalogen in ein Repository übernommen UND zueinander in Beziehung gesetzt. Die Datenbank-Modelle werden sukzessive durch die

dahinter liegenden DATEN-Modelle ergänzt und dokumentiert. Dies gilt für alle produktiven Datenbestände - unabhängig vom DBMS.

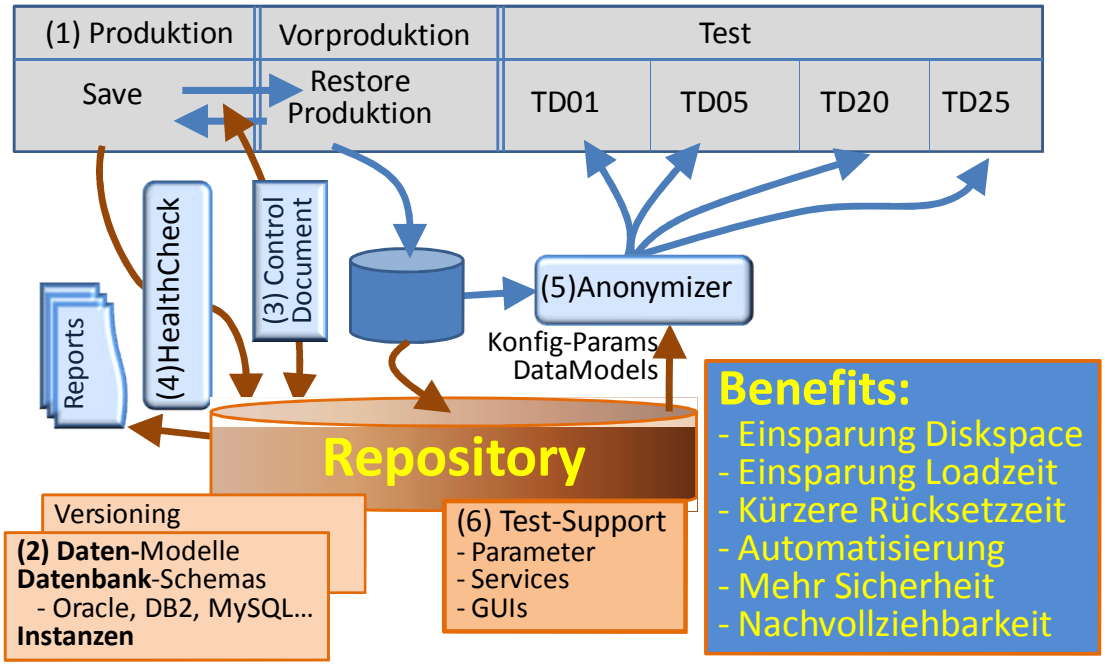

Abb 7: Data-Management Support System

(3) Die Save-Restore-Prozeduren werden im Repository verwaltet und geprüft. Sowohl die Beschreibung als auch jede Durchführung wird im Repository festgehalten.

(4) Regelmäßig durchgeführte Health-Checks auf die Datenbanken werden von Parametern gesteuert. Die Ergebnisse werden gegen Grenzwerte geprüft und fließen in das Reporting über die Datenbanken ein. Das alte Excel-Reporting wird abgelöst durch Online-Zugriff auf das Repository oder konfigurierte Auswertungen.

(5) Der (im Prinzip nicht stattfindende) Zugriff auf Produktions- und Vorproduktionsbestände soll vollständig unterbunden werden - auch für die diversen "privaten" Programme die Testdaten aus diesen Beständen abziehen. Die Funktion dieser Extraktionsprogramme übernimmt ein "Anonymizer" der von den Datenmodellen der Quellen und Zieldatenbestände und Konfigurations-Parametern gesteuert wird.

(6) Aus den Modellen werden Services für die Aufbereitung und den Zugriff auf die Daten generiert sowie Konfigurations-Parameter für die Testdaten-Generierung erzeugt.

Chancen und Nutzen des Systems

Einsparungen an Diskspace von ca. 25% des bestehenden Speichervolumens
Einsparungen Loadzeit: Damit Reduktion der kritischen Wartungsfenster
Einsparung Rücksetzzeit: Damit Verbesserung der MTBF (mean time between failure).
Automatisierung: Nicht nur Healthchecks und Anonymizer
Sicherheit: Fehlerrate reduzieren, insbesondere Handlingsfehler
Nachvollziehbarkeit: Anforderungen an systematische Protokollierung erfüllen

Kontaktadresse:

Dr. Reinhold Thurner
Metasafe GmbH
Rebweg 21
CH8700 Küsnacht

Telefon: +49 (0) 69-175374894
E-Mail: Reinhold.Thurner/at/Metasafe-Repository.com
Internet: www.Metasafe-Repository.com

Markus Rahlff
Metasafe GmbH
Bgm.-Schneider-Weg 135
D-85579 Neubiberg

Telefon: +49 (0) 69-175374895
E-Mail: Markus.Rahlff/at/Metasafe-Repository.com
Internet: www.Metasafe-Repository.com