Computer Aided Warehouse Engineering auf Basis der Model Driven Architecture

Christian Kurze, Peter Gluchowski

Technische Universität Chemnitz – Fakultät für Wirtschaftswissenschaften
Professur Systementwicklung / Anwendungssysteme

1 Einleitung

1.1 Motivation und Problemstellung

Die Model Driven Architecture (MDA) ist ein Standard der Object Management Group zur modellgetriebenen Software-Entwicklung (Object Management Group 2003). Hinter dem Ansatz verbirgt sich im Kern die Idee einer strikten Trennung von Spezifikation und Implementierung eines Systems. Modelle auf verschiedenen Abstraktionsebenen und automatisierte Transformationen zwischen Modellen setzen diese Idee um. Als neuerer Ansatz im Kontext der MDA adressiert die Architecture Driven Modernization (ADM) die Modernisierung bzw. das Reengineering von Systemen sowohl aus fachlicher als auch aus technischer Perspektive und stellt formale Modelle für diesen Kontext bereit (Khusidman und Ulrich 2007, Khusidman, 2008).

Data Warehouses stellen wichtige Informationen zur Unterstützung strategischer und taktischer Entscheidungen bereit, und verkörpern somit zentrale Komponenten moderner Entscheidungsunterstützungssysteme (Inmon 2005). In den letzten zehn Jahren hat das Data Warehousing einen hohen Reifegrad erreicht und flächendeckende Akzeptanz gefunden. Datenschutz, Datensicherheit und Compliance erfordern zunehmend höhere Anstrengungen in der Entwicklung und Dokumentation von Data Warehouses sowie im Metadatenmanagement. Zusätzlich ist eine Schwerpunkverschiebung der Initiativen auf die fachliche Seite – weg von der IT – zu verzeichnen (Kimball, Ross, Thornthwaite, Mundy und Becker 2008). Diese Punkte stellen den Mehrwert einer modellgetriebenen Data Warehouse-Entwicklung heraus: Das Data Warehouse muss Daten in einer für Fachanwender verständlichen Art und Weise bereitstellen. Eine detaillierte Systembeschreibung erhält nunmehr zwei Bedeutungen: Zum einen dient sie als Zusammenstellung der Anforderungen an das System und damit einhergehend als Dokumentation für Endnutzer und Entwickler; gleichzeitig aber auch als solide Basis für die Implementierung

des Zielsystems, da die Befolgung des MDA-Paradigmas die Transformation in ein reales System erlaubt.

Die aktuelle Literatur, die MDA auf die Entwicklung von Data Warehouses anwendet, berücksichtigt den ADM-Ansatz nur unzureichend. Dennoch erweist sich vor allem das Framework von (Mazón und Trujillo, 2008) als ein guter Ausgangspunkt für die Entwicklung von CAWE-Werkzeugen auf Basis des MDA-Paradigmas. Die Autoren stellen eine Fallstudie vor, um die Machbarkeit des vorgestellten Ansatzes aufzuzeigen, geben allerdings keine konkreten Hinweise zur Implementierung. Ähnlich der genannten Arbeit konzentriert sich der vorliegende Beitrag auf die Datenhaltungsschicht einer Data Warehouse-Lösung – speziell auf die Modellierung mehrdimensionaler Datenstrukturen. Dabei repräsentiert der ADM-Ansatz ein solides und tragfähiges Fundament, um auf bestehenden Data Warehouse-Systemen ein Reverse-Engineering durchzuführen.

Das hier vorgestellte Werkzeug ist sowohl für die Wissenschaft als auch für Praktiker relevant. Es stellt eine Plattform für Forscher bereit, um aktuelle Methoden des Software-Engineerings auf die Spezifika des Data Warehousings anzupassen; Praktikern hilft es bei der Bewältigung von komplexen Fragestellung im Rahmen der (Weiter-) Entwicklung von Data Warehouse-Systemen.

Aus den bisherigen Ausführen ergeben sich unmittelbar zwei zentrale Forschungsfragen: Wie sind die Ansätze MDA und ADM in eine Architektur für ein CAWE-Werkzeug, das zunächst die Aspekte der Datenmodellierung abdeckt, zu integrieren? Wie lässt sich einen solche Architektur implementieren?

Die vorliegende Arbeit stellt sich der Aufgabe, diese Lücke zu schließen. Die präsentierte Software-Architektur integriert sowohl MDA als auch ADM. Bereits existierende Frameworks zum effizienten Umgang mit Modellen werden auf das Problem angewendet, um durch eine prototypische Implementierung die Umsetzbarkeit der Architektur zu zeigen.

1.2 Forschungsmethodik

Der Design Science Ansatz verfolgt die Zielstellung, praktischen und theoretischen Herausforderungen durch die Erstellung und Evaluation von IT-Artefakten zu begegnen. Die so erstellten Artefakte lösen die aufgezeigten organisatorischen Probleme (March und Smith 1995, Hevner, March, Park und Ram 2004, Peffers, Tuunanen, Rothenberger und Chatterjee 2008). Der von (Peffers et al. 2008) vorgestellte Forschungsprozess findet in der vorliegenden Arbeit wie folgt Anwendung:

- *Problemidentifikation und Motivation:* Das Einleitungskapitel legte die Problemstellung immer komplexerer Data Warehouse-Architekturen dar und motivierte die Verwendung von MDA-Konzepten als tragfähigen Lösungsansatz.

- *Definition der Zielstellungen der Lösung:* Die gegebenen Forschungsfragen repräsentieren die Zielstellung: Es ist eine Software-Architektur zu entwickeln, die es erlaubt, MDA und ADM für die Modellierung mehrdimensionaler Datenstrukturen einzusetzen.

- *Design und Entwicklung:* In dieser Phase finden verwandte Arbeiten Anwendung, um die gesetzten Ziele zu erreichen. Die Architektur wird durch einen Prototypen exemplarisch umgesetzt.

- *Demonstration:* Ein Proof-of-Concept zeigt die Eignung des erstellten Prototyps zur Lösung der gestellten Problematik.

- *Evaluation:* Im Rahmen der Evaluation ist der Prototyp auf Realwelt-Probleme anzuwenden. Die hier identifizierten Ergänzungen dienen als Input für einen Kreislauf zur weiteren Verbesserung und Erweiterung.

2 Verwandte Arbeiten und Zielstellungen von CAWE

2.1 Model Driven Architecture (MDA) und Architecture Driven Modernization (ADM)

Die Object Management Group (OMG) vereint als non-profit Industriekonsortium Unternehmungen um den ganzen Globus. Der Zweck seiner Gründung besteht in Beschleunigung sowie Senkung von Komplexität und Kosten in der Softwareentwicklung. Detaillierte Spezifikationen führen zu interoperablen, wiederverwendbaren sowie portablen Softwarekomponenten auf Basis standardisierter Modelle (Object Management Group 2003).

Der Mehrwert der MDA liegt in der Möglichkeit, maschinenlesbare Applikations- und Datenmodelle zu erstellen, die eine langfristige Flexibilität von Implementierung, Integration, Wartung, Test und Simulation sicherstellen. Die bereits genannte strikte Trennung zwischen funktionaler Systemspezifikation und Implementierungsdetails einer konkreten Zielplattform mündet in der Einführung von drei sogenannten Viewpoints auf ein System. Im Kontext der MDA ist ein Viewpoint als Abstraktionsstufe zu verstehen, die für die jeweilige Stufe unnötige Details zur Erlangung eines vereinfachten Modells ausblendet. Jeder Viewpoint wird durch ein korrespondierendes Modell repräsentiert. Das

Computation Independent Model (CIM), auch Domänen-Modell genannt, nutzt als implementierungsunabhängige Sicht das Vokabular der Anwendungsdomäne. Hauptzweck ist es, eine Brücke zwischen Domänenexperten und Entwicklern zu schlagen. Das *Platform Independent Model* (PIM) bleibt in dem Sinne plattformunabhängig, dass es für verschiedene Plattformen ähnlichen Typs zur Anwendung kommt. Die Ergänzung des PIM um zusätzliche Details bezüglich der Umsetzung auf einer konkreten Zielplattform führt zum *Platform Specific Model* (PSM) (Object Management Group 2003). Aufgrund verschieden starker Ausdrucksstärken von Modellierungssprachen ist es durchaus möglich, dass mehrere, zueinander in Beziehung stehende, Modelle zur Beschreibung eines Systems herangezogen werden. Dieser Sachverhalt ist vor allem auf CIM-Ebene von Relevanz. Ergänzend zu Modellen auf verschiedener Abstraktionsebene spielen Modelltransformationen eine zentrale Rolle in der MDA: Sie konvertieren verschiedene Modelle des gleichen Systems ineinander; beispielsweise das CIM in ein PIM und dieses wiederum in ein PSM und anschließend in Code. Abbildung 1 fasst das beschriebene Gesamtkonzept zusammen.

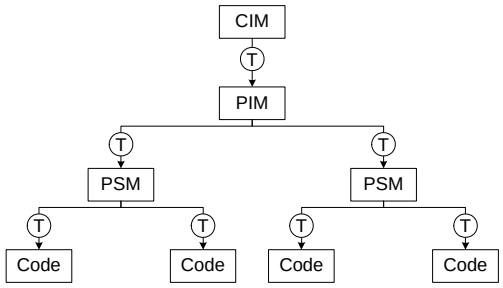

Abbildung 1: Grundkonzept der MDA (Mazón und Trujillo 2008)

Die Meta Object Facility (MOF) stellt eine vierstufige Metamodellierungsarchitektur als Rahmen für MDA bereit. Jede Ebene definiert die jeweils darunter liegende: Die unterste Ebene enthält die zu beschreibenden Daten. Auf zweiter Ebene finden sich Modelle zur Beschreibung der Daten – sogenannte Metadaten. In der dritten Ebene sind Metamodelle anzusiedeln, die wiederum die Struktur der Metadaten definieren. Das Meta-Metamodell auf vierter Ebene beinhaltet die Beschreibung von Struktur und Semantik der Metamodelle. Es stellt also die abstrakte Sprache zur Beschreibung verschiedener Arten von Metamodellen bereit (Object Management Group 2002, Object Management Group 2006).

Die Architecture Driven Modernization (ADM)-Task Force der OMG setzt sich eine Spezifikation zur Modernisierung bereits existierender Applikationen zum Ziel (Object Management Group 2009). Den Kern des Ansatzes bildet das in Abbildung 2 dargestellte Hufeisenmodell (Kazman, Woods und Carrière 1998, Khusidman und Ulrich 2007, Khusidman 2008). Es teilt sich horizontal in drei Ebenen: *Business Architecture*, *Applica-*

tion and Data Architecture sowie *Technical Architecture*. Vertikal ist zwischen IST- und SOLL- bzw. Quell- und Zielsystem zu unterscheiden. Das Hufeisen beschreibt verschiedene Pfade von der ursprünglichen Lösung zur neuen Zielarchitektur. Der kürzeste Weg verbleibt dabei innerhalb einer Ebene. Veränderungen innerhalb der technischen Architektur, z.B. neue Hardware, beeinflusst lediglich eine Ebene. Der längste Transformationspfad, der in Abbildung 2 gezeigt ist, verläuft durch alle drei Ebenen. Unabhängig von dessen Länge ist jeder Transformationspfad durch drei Schritte gekennzeichnet: Strukturextraktion aus bestehenden Systemen, Definition der Zielarchitektur sowie die Transformation des IST-Systems in das spezifizierte SOLL-System.

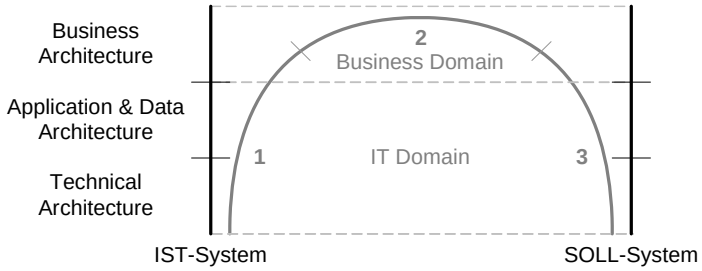

Abbildung 2: Prinzip der Architecture Driven Modernization (Khusidman und Ulrich 2007)

2.2 Data Warehousing und MDA/ADM

Im Kontext des Data Warehousings existieren verschiedene CIMs, beispielsweise ADAPT (Bulos und Forsman 2006), DFM (Golfarelli, Maio und Rizzi 1998) oder ME/R (Sapia, Blaschka, Hofling und Dinter 1998). Allerdings konnte sich bislang keiner dieser Ansätze als allgemein akzeptierte multidimensionale Modellierungstechnik durchsetzen. Insgesamt erweist sich die Definition eines Modellierungskonzeptes, welches alle mehrdimensionalen Besonderheiten abdeckt, als schwierig. Auf PIM/PSM-Ebene konkurrieren drei Ansätze gegeneinander: Das auf relationalen Datenbanken basierende ROLAP, auf multidimensionalen Datenbanken basierendes MOLAP sowie eine hybride Kombination beider Herangehensweisen, HOLAP (Chaudhuri und Dayal 1997). Der Unterschied zwischen PIM und PSM für ein ROLAP-System besteht beispielsweise darin, dass ein PIM weder Datentypen noch konkrete Indexierungsverfahren beschreibt; ein PSM hingegen auf die speziellen Gegebenheiten eines konkreten Systems eingeht.

In diesem Punkt unterscheidet sich die Auffassung der Autoren gegenüber der von (Mazón und Trujillo 2008). Grafische Notationen wie DFM oder ME/R sind in der vorliegenden Arbeit den CIMs zugerechnet. Gemäß Definition sind diese Modelle zur Kommunikation zwischen Endanwendern und Entwicklern gedacht (Object Management Group 2003, S. 2-5). PIMs sind für eine Reihe verschiedener Plattformen ähnlichen Typs

konzipiert (Object Management Group 2003, S. 2-6). Ein relationales Star-Schema beispielsweise ist für verschiedene konkrete relationale Datenbanksysteme geeignet. Das PSM reichert ein PIM mit Spezifika einer konkreten Plattform (Object Management Group 2003, S. 2-6) dahingehend an, dass, am Beispiel des Star-Schemas, spezifische Datentypen, Indexierungsmechanismen, Tabellenpartitionierungen und weitere Merkmale eines konkreten Datenbanksystems angegeben sind.

Das Framework von (Mazón und Trujillo 2008) deckt verschiedene Ebenen des Data Warehousings ab: Datenquellen, ETL-Prozesse, mehrdimensionale Datenmodellierung eines relationalen Data Warehouses, mehrdimensionale Würfel sowie Applikationsmodelle, beispielsweise für das Reporting, die freie Datenanalyse oder das Data Mining. Im Gegensatz zur genannten Arbeit fassen die Autoren die Modellierung eines relationalen Warehouses und mehrdimensionaler Würfel in eine Ebene der mehrdimensionalen Datenmodellierung zusammen. Diese Zusammenfassung resultiert aus der Unterscheidung zwischen ROLAP, MOLAP und HOLAP. Jede Realisierungsform erfordert verschiedene Beschreibungen. Im Falle von HOLAP existieren beide Beschreibungsformen und bedingen einander. Jede Schicht einer Data Warehouse-Anwendung wird von allen drei Viewpoints (CIM, PIM, PSM) aus modelliert und anschließend in Code transformiert. Die CIM-Ebene stellt dabei den wesentlichen Input für mehrdimensionale Daten- und Applikationsmodelle auf PIM-Ebene dar. Abbildung 3 fasst das Rahmenkonzept zusammen.

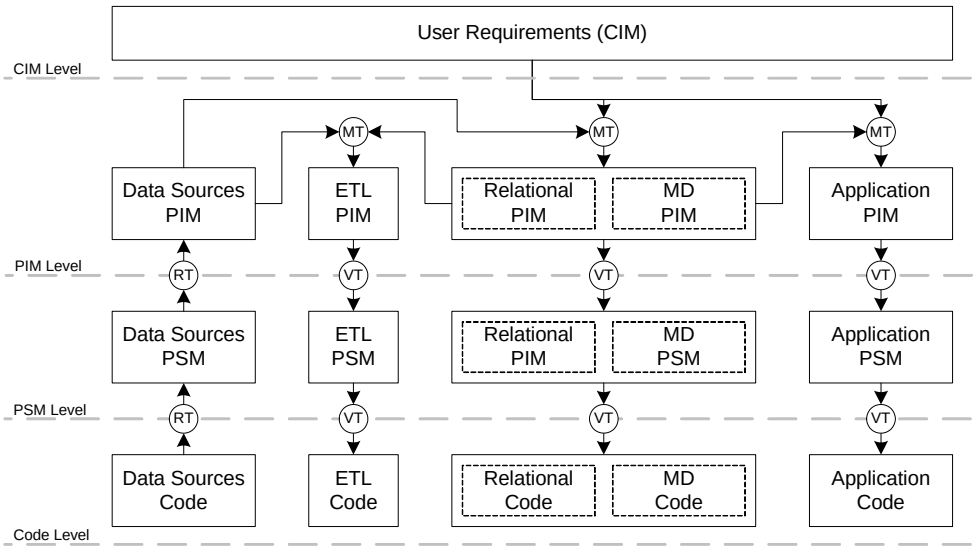

Abbildung 3: MDA-Framework für die Entwicklung von Data Warehouses (Mazón und Trujillo 2008)

Die Überführung von Modellen ineinander erfordert eine Reihe von Transformationen innerhalb des Frameworks. Zur Verwendung in einem modellgetriebenen Vorgehen ist es notwendig, zunächst sämtliche Quellsysteme durch Modelle zu beschreiben. In den meisten Fällen existieren diese nicht und sind, ausgehend von Quelltexten bzw. operationalen Datenbanksystemen, zu entwickeln. Diese Transformationen sind als RT (Reverse Transformation) in Abbildung 3 dargestellt. Transformationen, die mehrere Modelle zu einem Zielmodell zusammenführen, sogenannte Merging Transformations, sind als MT abgebildet. Ein ETL-Modell auf PIM-Ebene vereint beispielsweise Modelle der Datenquellen mit Modellen der mehrdimensionalen Zielsysteme. Vertikale Transformationen (VT) transformieren Modelle in darunterliegende Abstraktionsebenen in Richtung Code.

Das beschriebene Framework beinhaltet ADM-Mechanismen lediglich in der Quellsystem-Ebene bei der Erstellung von PIMs aus operativen Systemen heraus. Aus Sicht der Autoren ist jedoch ein formaler und systematischer Weg zur Unterstützung des Verstehens, Anpassens und Weiterentwickelns von gesamten Data Warehouse-Lösungen nötig. Diese Argumentation führt zur Postulation eines ADM-Ansatzes auf allen Ebenen des Data Warehousings: ETL, mehrdimensionale Daten und Applikationen. Ein derartiges Vorgehen ist auch für die Einführung eines modellgetriebenen Entwicklungsvorgehens für bereits im Einsatz befindliche Data Warehouse-Systeme unerlässlich: Im ersten Schritt sind für alle Systemelemente geeignete Modelle zu erstellen. Aufgrund der semantischen Lücke zwischen CIM und PIM stellt sich diese Aufgabe als schwierig heraus. Um Data Warehouse-Entwickler bestmöglich zu unterstützen, ist ein automatisiertes Verfahren für diesen Arbeitsschritt zu entwickeln.

Die Ausführungen zeigen eine weitere Anforderung an die Architektur auf: Sie muss komplexe Algorithmen für das Entdecken von mehrdimensionalen Elementen innerhalb von PIMs und deren Transformation in Elemente eines CIMs verwalten und ausführen können. Da den einzelnen Elementen zusätzliche Semantik hinzuzufügen ist, sind die Ergebnisse verschiedener Algorithmen zu evaluieren und in ein resultierendes CIM zu integrieren.

2.3 Fusioniertes CIM

Das von uns vorgeschlagene fusionierte, implementierungsunabhängige Modell basiert auf der Analyse verschiedener Quellen zur konzeptionellen multidimensionalen Modellierung und fußt auf dem UML-Profil von (Luján-Mora, Trujillo und Song 2006). Die darin enthaltenen Abstraktionsstufen sind, speziell für sehr große Modelle, von essenzieller Bedeutung (Kurze und Gluchowski 2009). Die aktuelle Implementierung lässt die Abstrahierung jedoch zur Reduktion der Komplexität vorerst außen vor. Das von (Pedersen, Jensen und Dyreson 2001) aufgestellte konzeptionelle mehrdimensionale Modell gilt

als eines der umfangreichsten Modelle auf implementierungsunabhängiger Ebene (Mansmann und Scholl 2007, S. 36). Die darin aufgestellten elf Regeln, die ein mehrdimensionales Modell zu befolgen hat, werden zur Erweiterung des Ausgangsmodells herangezogen. Eine formale Beschreibung stellen (Mansmann und Scholl 2007) vor. Weiterhin finden Klassifikationen von Hierarchien (Malinowski und Zimányi 2004, Malinowski und Zimányi 2006) sowie Kennzahlen-Dimensions-Beziehungen (Mazón, Lechtenbörger und Trujillo 2008, Mazón, Lechtenbörger und Trujillo 2009) Anwendung. Hauptziel ist das Abdecken von nahezu jeder möglichen Besonderheit der mehrdimensionalen Datenmodellierung. In realen Anwendungen finden idealtypische Schemata aufgrund komplizierter Anforderungen nur selten Anwendung.

In der ersten Iteration der Metamodell-Entwicklung liegt der Fokus auf der granularsten Ebene von (Luján-Mora et al. 2006). Die Konzepte degenerierter Fakten und Dimensionen finden im Ansatz des vorliegenden Papers keine Betrachtung, da es sich hierbei aus Sicht der Autoren um logische Konstrukte handelt, die auf CIM-Ebene noch nicht relevant sind.

Folgende Anforderungen von (Pedersen et al. 2001) sind in der ersten Iteration nicht integriert worden: Unterstützung der Aggregationssemantik von Kennzahlen bezüglich der Dimensionen, symmetrische Behandlung von Kennzahlen und Dimensionen (die beispielsweise die Verwendung von Kundenalter als Kennzahl zur Durchschnittsbildung, aber auch zum Filtern einzelner Kunden erlaubt), temporale Aspekte wie beispielsweise die Veränderung von Dimensionsinstanzen, Umgang mit verschiedenen Granularitäten sowie das Handling von unpräzisen Daten.

3 Architektur und Implementierung von CAWE

3.1 Konzeptionelle Architektur

Aus den zugrunde gelegten Aspekten der Metamodellierung ergeben sich die wesentlichen Anforderungen an CAWE-Werkzeuge. Ausgangspunkt der in diesem Kapitel vorgestellten Architektur bildet deshalb ein generischer Ansatz zur Anordnung von Systemkomponenten von (Karagiannis und Kühn 2002). Wie *Abbildung* zeigt, stellt das *Meta²-Modell* die Kernkonzepte zur Erzeugung von Metamodellen und Mechanismen bereit. Es steht im Zentrum der Architektur und ist mit allen weiteren Komponenten verbunden. Modelle und Metamodelle sind im *Modell-* bzw. *Metamodell-Repository* abgelegt. Beide Repositories stehen zueinander in Beziehung, um Veränderungen an Metamodellen an die entsprechenden Modelle zu propagieren, d.h. sie synchron zu halten. Auf Modelle und Metamodelle anwendbare Funktionen sind im *Mechanismen-Repository* gespeichert. Ihre Ablage erfolgt entweder komplett oder als externe Referenz im Repository. Im zweiten

Fall befindet sich zusätzlich eine Schnittstellenbeschreibung zur Interaktion mit dem Baustein im Repository. Der *Persistenzierungsdienst* ist aus den einzelnen Datenspeichern ausgegliedert: Er verwaltet die persistente Ablage sämtlicher (Meta-) Modelle und Mechanismen. Der Zugriff erfolgt transparent und unabhängig von der gewählten physischen Speichermethode wie beispielsweise Datenbanksystemen oder dem Dateisystem. Zusätzlich ermöglicht der Dienst die Verteilung von (Meta-) Modellen, d.h. sehr große Modelle können in kleineren Teilmodellen persistenziert werden. Zugriff auf die einzelnen Komponenten erfolgt über die *Zugriffsdienste* per API oder Dateiaustausch. *Viewer und Builder-Komponenten* an der Spitze der Architektur erlauben Nutzung und Wartung der gesamten Plattform.

Die gestellten Anforderungen der Abschnitte 2.1 und 2.2 sind durch die vorgestellte Architektur erfüllt. Sie dient als Basis für die Entwicklung von Metamodellierungswerkzeugen und deckt somit die benötigten Metamodellierungsaspekte ab. Modelltransformationen sind im Mechanismen-Repository abgelegt und lassen sich auf (Meta-) Modelle anwenden. ADM-Spezifika sind ebenfalls abbildbar: Modell-Discovery aus bestehenden Systemen erweist sich als Mechanismus; die Integration komplexer Algorithmen für diese Aufgabe erfolgt aus externer Ablage. Transformationen von PSMs in PIMs sind ebenfalls im Mechanismen-Repository gespeichert. Ein fusioniertes CIM, wie in Abschnitt 2.3 vorgestellt, ist im Metamodell-Repository bzw. dessen Instanz im Modell-Repository abzulegen.

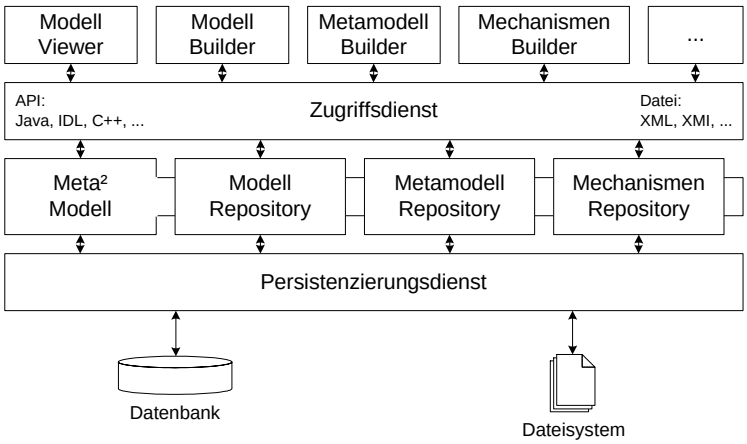

Abbildung 4: Konzeptionelle CAWE-Architektur (Karagiannis und Kühn 2002)

3.2 Implementierung der Komponenten

Die gewählte Implementierung basiert auf dem Eclipse Modeling Project (Gronback 2009). Es handelt sich dabei um ein Open Source-Projekt mit einer großen Gemeinde an Entwicklern, die permanent an dessen Evolution arbeiten. Als Meta²-Modell findet das

Ecore-Metamodell des Eclipse Modeling Frameworks (EMF) Anwendung (Steinberg, Budinsky, Paternostro und Merks 2009). Die Modell- und Metamodell-Repositories sind derzeit durch einen dateibasierten Persistenzierungsdienst unterstützt. Das Mechanismen-Repository dient hauptsächlich zur Ablage von Modelltransformationen. Komplexere Algorithmen, insb. zum Modell-Discovery, sind in Java geschrieben. Sämtliche Metamodelle können in Form von Eclipse-Plugins bereitgestellt werden und unterstützen somit einen transparenten Zugriff per API im Sinne der Zugriffsdienste; ein Datei-Export ist ebenso möglich. Viewer- und Builder-Komponenten sind im Rahmen von EMF ebenfalls vorhanden. Integrierte, baumbasierte Editoren lassen sich automatisch aus Metamodellen generieren (Abbildung 7 zeigt Beispiele); ein baumbasierter wie auch ein grafischer Editor für Metamodelle stehen zur Verfügung.

Abbildung 5 zeigt das Prinzip der Modell-Discovery: Das Schema relationaler Datenbanksysteme ist in Form vom Modellen – korrespondierend zu deren Metamodellen – abzubilden. Der Discoverer liest das Datenbankschema und legt ein Modell entsprechend der gefundenen Strukturen an. Die Implementierung ist der MoDisco-Toolbox des Eclipse Modeling Projects entnommen und entsprechend angepasst.
Das erstellte Schemamodell, das in relationalen Datenbanksystemen aufgrund deren ausgeprägten Metadaten-Repositories stets erstellt werden kann, erlaubt die Anwendung von Regeln zum Aufdecken der typischen Snowflake-Struktur.

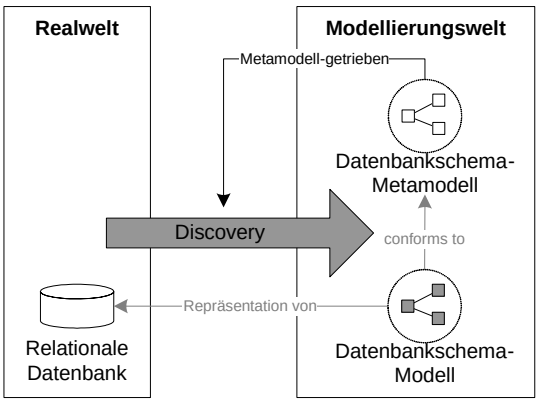

Abbildung 5: Prinzip von Model-Discovery für relationale Datenbanken

4 Anwendung einer Fallstudie

Zur Evaluation der vorgestellten Lösung kommt die Fallstudie eines Unternehmens zum Einsatz, das seine Absatzzahlen (Absatzmenge, Preis und Umsatz) hinsichtlich verschiedener Produkte, Organisationseinheiten und Kunden über die Zeit hinweg analysieren möchte. Dieses Beispiel lässt sich mithilfe eines Würfels (Sales) mit vier Dimensionen

(Organisational Unit, Customer, Product und Time) modellieren. Die gesamte Kette verschiedener (Meta-) Modelle ist in Abbildung 6 aufgeführt. Metamodelle sind durch Kreise mit weißen Quadraten, Modelle mit grauen Quadraten dargestellt. Die Ausführung einer Transformation repräsentiert eine Raute mit grauen Quadraten, wobei MM eine Modell zu Modell-Transformation, MT eine Modell zu Text-Transformation und MS Modell-Discovery (im Sinne einer System zu Modell-Transformation) kennzeichnet.

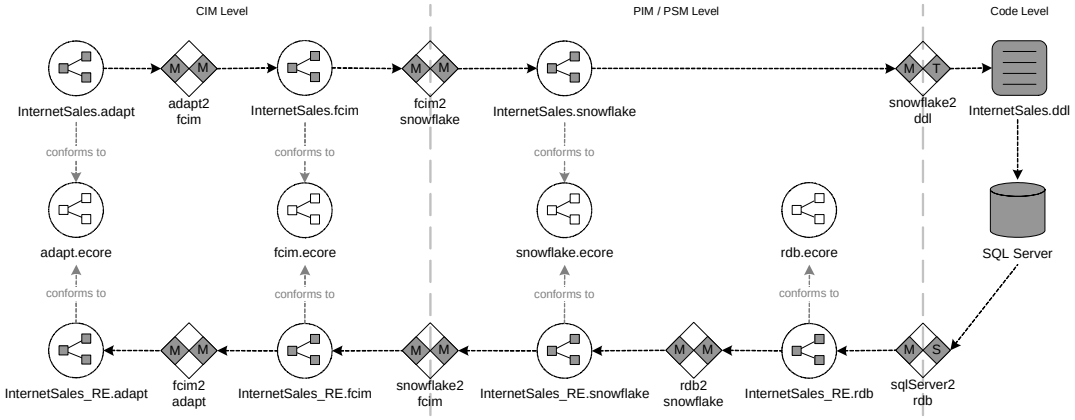

Abbildung 6: Kette von (Meta-) Modellen der Fallstudie

Das ADAPT-Modell ist zur Erhöhung der Wiederverwendbarkeit, gemäß den Ausführungen in Kapitel 2.3, zunächst in ein übergreifendes CIM zu transformieren. Aus diesem generalistischen Modell entsteht durch Transformation ein Snowflake-Schema-Modell. Das zugehörige Metamodell ist eine Erweiterung des relationalen Modells um eine zusätzliche Klassifikation der Tabellen, d.h. Fakten- bzw. Dimensionstabellen, und entstammt maßgeblich aus dem Relational Package des CWM. Im nächsten Schritt sind den einzelnen Attributen der Relationen Datentypen zuzuweisen. Dieser Schritt erfolgt derzeit manuell. Aufgrund systemspezifischer Datentypen stellt dieser Schritt die Transformation des PIM in ein PSM dar; das InternetSales.snowflake-Modell ist deshalb in den Ebenen PIM und PSM zugeordnet. Ein Java Emitter Template (JET) generiert DDL-Code zur Implementierung in Microsoft SQL Server. Es erstellt Tabellen, Spalten und Fremdschlüsselbeziehungen zwischen den verschiedenen Tabellen. In der Reverse-Richtung ist das Ergebnis des Modell-Discoverers eine relationale Datenbankbeschreibung, in der durch eine zusätzliche Transformation einzelne Tabellen entsprechend ihrer Funktion (Fakt oder Dimension) klassifiziert sind.

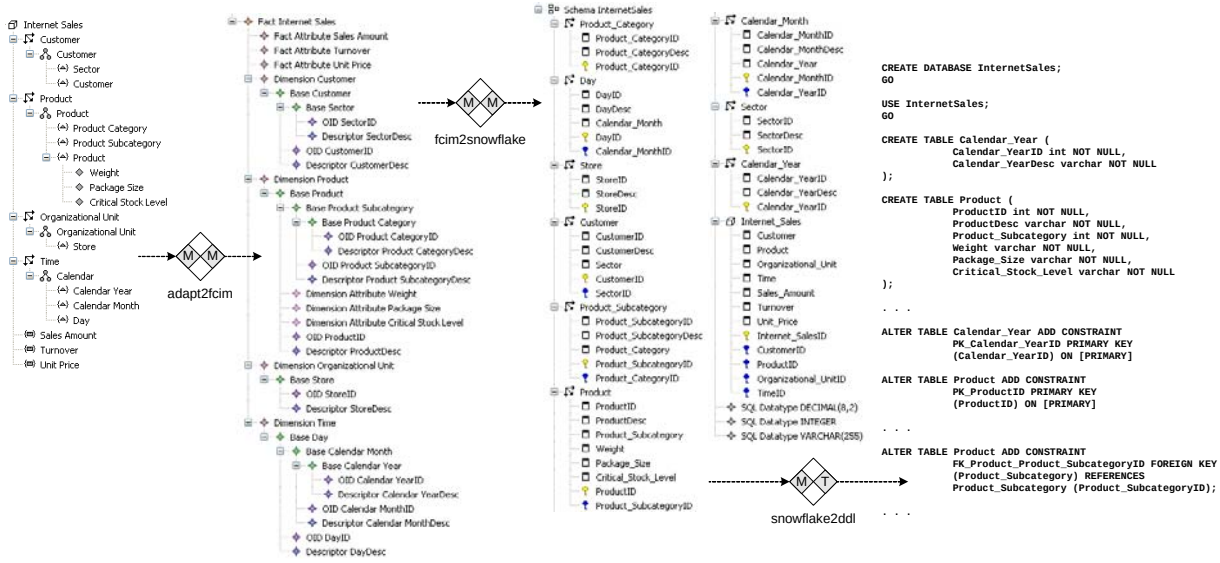

Abbildung 7: Forward-Engineering der Fallstudie

5 Kritische Würdigung und weitere Arbeitsschritte

Das vorliegende Paper stellt Forschungsergebnisse dar, die in weiteren Arbeiten zu vertiefen sind. Dabei spielen vor allem die Erweiterung des fusionierten CIM sowie die Verbesserung der Transformation von Modellen relationaler Datenbanken in mehrdimensionale Modelle eine entscheidende Rolle. Obwohl die vorliegende Arbeit nur beschränkt generalisierbar ist, zeigt sie das Potenzial von CAWE sowie dessen Umsetzbarkeit. Weitere Arbeitsschritte müssen folgen, um die prototypische Anwendung in einem breiten Rahmen einsetzen zu können.

6 Literatur

Bulos, D. und Forsman, S. (2006) Getting Started with ADAPT. Whitepaper, http://symcorp.com/downloads/ADAPT_white_paper.pdf. Letzter Abruf am 13. September 2009.

Chaudhuri, S. und Dayal, U. (1997) An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26(1), 65-74.

Golfarelli, M., Maio, D. und Rizzi, S. (1998) The Dimensional Fact Model: A Conceptual Model for Data Warehouses. International Journal of Cooperative Information Systems, 7(2-3), 215-247.

Gronback, R. C. (2009). Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit. Addison-Wesley.

Hevner, A. R., March, S. T., Park, J. und Ram, S. (2004) Design Science in Information Systems Research. MIS Quarterly, 28(1), 75-105.

Inmon, W. H. (2005) Building the Data Warehouse, 4. Aufl. Wiley.

Karagiannis, D., & Kühn, H. (2002). Metamodelling Platforms. In E-Commerce and Web Technologies, Springer, 451-464.

Kazman, R., Woods, S. G. und Carrière, S. J. (1998) Requirements for Integrating Software Architecture and Reengineering Models: CORUM II. In Proceedings of Working Conference on Reverse Engineering, 154-163.

Khusidman, V. (2008) ADM Transformation. Whitepaper, http://www.omg.org/cgi-bin/doc?admtf/2008-06-10. Letzter Abruf am 13. September 2009.

Khusidman, V. und Ulrich, W. (2007) Architecture-Driven Modernization: Transforming the Enterprise. Whitepaper, http://www.omg.org/cgi-bin/doc?admtf/2007-12-01. Letzter Abruf am 13. September 2009.

Kimball, R., Ross, M., Thornthwaite, W., Mundy, J. und Becker, B. (2008) The Data Warehouse Lifecycle Toolkit, 2. Aufl. Wiley.

Kurze, C. und Gluchowski, P. (2009) Towards Principles for Managing and Structuring Very Large Semantic Multidimensional Data Models. In Proceedings of the 15th Americas Conference on Information Systems.

Luján-Mora, S., Trujillo, J. und Song, I. (2006) A UML profile for multidimensional modeling in data warehouses. Data & Knowledge Engineering, 59(3), 725-769.

Malinowski, E., & Zimányi, E. (2004). OLAP Hierarchies: A Conceptual Perspective. In Advanced Information Systems Engineering, Springer, 477-491.

Malinowski, E. und Zimányi, E. (2006) Hierarchies in a multidimensional model: from conceptual modeling to logical representation. Data & Knowledge Engineering, 59(2), 348-377.

Mansmann, S. und Scholl, M. H. (2007) Empowering the OLAP Technoloy to Support Complex Dimension Hierarchies. International Journal of Data Warehousing and Mining, 3(4), 31-50.

March, S. T. und Smith, G. F. (1995) Design and natural science research on information technology. Decision Support Systems, 15(4), 251-266.

Mazón, J., Lechtenbörger, J. und Trujillo, J. (2008) Solving summarizability problems in fact-dimension relationships for multidimensional models. In Proceeding of the ACM 11th international workshop on Data warehousing and OLAP, 57-64.

Mazón, J., Lechtenbörger, J. und Trujillo, J. (2009) A survey on summarizability issues in multidimensional modeling. Data & Knowledge Engineering, In Press, doi: 10.1016/j.datak.2009.07.010.

Mazón, J. und Trujillo, J. (2008) An MDA approach for the development of data warehouses. Decision Support Systems, 45(1), 41-58.

Object Management Group. (2003) MDA Guide V1.0.1. http://www.omg.org/cgi-bin/doc?omg/03-06-01. Letzter Abruf am 13. September 2009.

Object Management Group. (2002) Meta Object Facility (MOF) Specification Version 1.4. http://www.omg.org/spec/MOF/1.4/PDF/. Letzer Abruf am 19. Juli 2009.

Object Management Group. (2006) Meta Object Facility (MOF) Core Specification Version 2.0. http://www.omg.org/docs/formal/06-01-01.pdf. Letzer Abruf am 19. Juli 2009.

Object Management Group. (2009) Architecture-Driven Modernization Task Force. http://omg.org/adm/. Letzer Abruf am 19. Juli 2009.

Pedersen, T. B., Jensen, C. S. und Dyreson, C. E. (2001) A foundation for capturing and querying complex multidimensional data. Information Systems, 26(5), 383-423.

Peffers, K., Tuunanen, T., Rothenberger, M. und Chatterjee, S. (2008) A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45-77.

Sapia, C., Blaschka, M., Hofling, G. und Dinter, B. (1998) Extending the E/R Model for the Multidimensional Paradigm. In Proceedings of the Workshops on Data Warehousing and Data Mining: Advances in Database Technologies, Springer, 105-116.

Steinberg, D., Budinsky, F., Paternostro, M. und Merks, E. (2009) EMF: Eclipse Modeling Framework, 2. Aufl. Addison-Wesley