



universität
wien

Masterarbeit

Titel der Masterarbeit

A Conceptual Method for Data Integration in Business Analytics

An Open Models Prototype Implementation

Verfasserin

Beatrice Gurell, Bakk.rer.soc.oec.

angestrebter akademischer Grad

Magistra der Sozial- und Wirtschaftswissenschaften (Mag.rer.soc.oec.)

Wien, 2011

Studienkennzahl:	A 066 926
Studienrichtung:	Magisterstudium Wirtschaftsinformatik
Betreuer:	Univ.-Prof. Dr. Wilfried Grossmann

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Masterarbeit ohne Hilfe Dritter und nur mir den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Acknowledgement

I would like to show my gratitude to **Univ.-Prof. Dr. Wilfried Grossmann**, whose research is the basis for my thesis, for his great support. Without his encouragement, guidance and especially his deep belief in my work from the initial to the final; this master thesis would not have been possible.

I would also like to thank **o. Univ.-Prof. Dr. Dimitris Karagiannis** who gave me the chance to conduct this thesis and the possibility to gain insights of topic of modelling and the opportunity to visit the conference Modellierung 2010 in Klagenfurt.

At last I would like to thank my **family and friends** who encouraged me and believed that I would finish my master thesis, especially **Wilfrid Utz** for all his positive energy and great feedback. A special thank goes to my **husband** for his support and patience and **my parents in law** who helped to take care of my daughter during the time I was conducting this thesis.

Curriculum Vitae

PERSONAL DATA:

Address: Penzinger Strasse 50/11, 1140 Vienna, Austria
E-mail: beatricegurell@hotmail.com
Date of birth: 21 August 1975
Nationality: Swedish

PROFESSIONAL INTERESTS:

Data Quality Management

Data Governance (Logical Data Modelling)

Business Intelligence Strategies

Data Integration

WORK EXPERIENCE:

04/2004 – present **bwin.party services (Austria) GmbH, Vienna**

Data Warehouse Developer / BI Engineer (05/2008 – present)

Data quality process management, test coordination of data warehouse integrations and requirement engineering

Database Marketing / Business Analyst (05/2005 – 04/2008)

Churn prediction (data mining) including analytical data set and modelling (decision tree and logistic regression). Customer base analysis and customer segmentation, CRM campaign management including user selections for direct marketing mailings and campaign measurement

Sales Controlling (04/2004 – 04/2005)

Reporting including requirement analysis, implementation and deployment

Translation (02/2001 – 03/2004)

German and English to Swedish

03/1999 – 09/2000 **Sumitomo Corporation Europe Plc, U.K., Stockholm**

Managing import-export accounts, accounting and contract administration

09/1998 – 02/1999 **Gulliver's Resebureau, Stockholm**

Assembling travel products and maintenance of price list database

EDUCATION:

2005 – present	Universität Wien (University of Vienna), Technische Universität Wien (Vienna University of Technology) Master's degree in Business Informatics (Wirtschaftsinformatik). Thesis: A Conceptual Method for Data Integration in Business Analytics - An Open Models Prototype Implementation
2000 – 2005	Universität Wien (University of Vienna), Technische Universität Wien (Vienna University of Technology) Bachelor's degree in Business Informatics (Wirtschaftsinformatik), main focus on business modelling and information management
1997 – 1998	Holmes Colleges, Melbourne Advanced Certificate in Travel Operations
1996	InfoKomp, Stockholm Tourism, marketing and business
1995	Stockholms universitet, (Stockholm University) German language
1991 – 1994	Frans Schartau Gymnasium, Stockholm High school

ADDITIONAL QUALIFICATIONS:

Software:	SPSS Clementine, Business Objects, MS Office
Relational database:	SQL Server and Oracle (SQL Server Management Studio and TOAD)
Trainings and seminars:	SPSS, Clementine training 2005 EC3, Data mining Conference 2005, 2006, 2007 The Data Warehousing Institute, European TDWI Conference 2008
Languages:	Excellent knowledge of German, Swedish (mother tongue) and English language. Oral and reading understanding of Spanish.

Table of Content

Table of Content	1
Zusammenfassung	3
Abstract	5
Table of Abbreviations.....	7
Table of Figures	8
Table of Tables.....	9
1 Introduction	10
1.1 Purpose and Goal of the Thesis.....	10
1.2 Organisation of the Master Thesis	10
2 Theoretical Background	12
2.1 Definition and Terminology.....	12
2.1.1 Definition: Business Intelligence and Business Analytics.....	12
2.1.2 Definition: Data Integration	13
2.2 Data Integration.....	14
2.2.1 Causes of Data Integration.....	14
2.2.2 Distribution, autonomy and heterogeneity.....	16
2.2.3 Materialized and Virtual Integration.....	20
2.2.4 Architectures	21
2.2.5 Approaches.....	25
2.2.5.1 Schema Matching and Schema Mapping	27
2.2.5.2 Multi Database Language	34
2.2.5.3 Ontology.....	35
2.2.6 Challenges'.....	37
3 Meta-Modelling within the Open Model Initiative.....	39
3.1 The Open Model Initiative.....	39
3.1.1 Definition of the Term "Model"	39
3.2 Motivation and Benefit of Modelling.....	40
3.3 Modelling Methods.....	42
3.3.1 Modelling Language.....	43
3.3.2 Modelling Procedures.....	44
3.3.3 Mechanism and Algorithms.....	45
3.3.4 Meta ² Representation: Formal Representation of the Meta-Model.....	45
3.3.5 Meta-Modelling Platform Architecture.....	47
3.4 Roles in Meta-Modelling	49
4 A Conceptual Method for Data Integration	51
4.1 Domain Logic for Business Analytics	52
4.2 Information Logic for Business Analytics	53
4.3 Processing Logic for Business Analytics	54
4.4 Data Integration Process.....	55
5 Prototype Implementation on the ADOxx® Platform	57
5.1 „Conceptualization“-Approach.....	57
5.2 The ADOxx® Meta-Modelling Platform.....	57
5.3 Conceptualization Results.....	59
5.4 Modelling Procedure: Data Integration Methodology Guideline	59
5.4.1 Generic Definition and Dependencies.....	59

5.4.2	Data Integration Methodology Guideline	60
5.5	<i>Modelling Language</i>	61
5.5.1	Generic Definition and Dependencies	61
5.5.2	Modelling Language Conceptualization Results	62
5.5.3	Modelling Language Overview: Data Model	66
5.5.4	Modelling Language Overview: Process Model	67
5.6	<i>Mechanism and algorithms</i>	98
6	Conclusion	107
7	References	108

Zusammenfassung

Viele Unternehmen funktionieren derzeit in einem schnellen, dynamischen und vor allem unbeständigen Umfeld und wettbewerbsintensiven Markt. Daraus folgt, dass schnelle und faktenbasierende Entscheidungen ein wichtiger Erfolgsfaktor sein können. Basis für solche Entscheidungen sind oft Informationen aus Business Intelligence und Business Analytics.

Eine der Herausforderungen bei der Schaffung von hochqualitativer Information für Geschäftsentscheidungen ist die Konsolidierung der Daten, die häufig aus mehrfachen heterogenen Systemen innerhalb eines Unternehmens oder in ein oder mehreren Standorten verteilt sind. ETL-Prozesse (Extraction, Transforming and Loading) sind häufig im Einsatz, um heterogene Daten aus einem oder mehreren Datenquellen in einem Zielsystem zusammenzuführen mit dem Ziel Data Marts oder Data Warehouse zu erstellen. Aufgrund mangelnder allgemeiner Methoden oder Ansätze, um systematisch solche ETL-Prozesse zu bewältigen, und Aufgrund der hohen Komplexität der Integration von Daten aus multiplen Quellen in einer allgemeinen, vereinheitlichten Darstellung, ist es sowohl für Fachleute als auch für die wenige erfahrene Anwender schwierig, Daten erfolgreich zu konsolidieren. Derzeit wird der analytische Prozess oft ohne vordefiniertes Rahmenwerk durchgeführt und basiert eher auf informelles Wissen als auf eine wissenschaftliche Methodik.

Das größte Problem mit kommerzieller Software, die den Datenintegrationsprozess inklusive Visualisierung, Wiederverwendung von analytischen Sequenzen und automatischer Übersetzung der visuellen Beschreibung in einem ausführbaren Code unterstützt, ist, dass Metadaten für die Datenintegration generell nur syntaktisches Wissen darstellt. Semantische Informationen über die Datenstruktur sind typischerweise nur in rudimentärer Form vorhanden und das obwohl sie eine signifikante Rolle bei der Definition des analytischen Modells und der Evaluierung des Ergebnisse spielen.

Vor diesem Hintergrund hat Grossmann¹ das “Conceptual Approach for Data Integration for Business Analytics” formuliert. Es zielt darauf hin, die Komplexität der analytischen Prozesse zu reduzieren und Fachkräfte in ihrer Arbeit zu unterstützen, um somit auch den Prozess für weniger erfahrene Anwender in unterschiedlichen Domänen zugänglich zu machen. Das Konzept ist detailliertes Wissen über Daten in Business Analytics, speziell Information über Semantik, zu berücksichtigen. Der Fokus liegt auf die Einbeziehung der strukturierten Beschreibung der Transformationsprozesse im Business Analytics, wo Informationen über

¹ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009.

Abhängigkeiten und Nebeneffekte von Algorithmen auch inkludiert sind. Darüber hinaus bezieht dieser Ansatz das Meta-Modell² Konzept mit ein: es präsentiert ein Rahmenwerk mit Modellierungskonzepten für Datenintegration für Business Analytics.

Basierend auf Grossmans Ansatz ist das Ziel dieser Masterarbeit die Entwicklung eines Meta-Model Prototyps, der die Datenintegration für Business Analytics unterstützt. Der Fokus liegt auf dem intellektuellen Prozess der Umwandlung einer theoretischen Methode in einem konzeptuellen Model, das auf ein Rahmenwerk von Modellierungsmethoden angewendet werden kann und welches zu den spezifischen Konzepten für eine bestimmte angewandte Meta-Model Plattform passt. Das Ergebnis ist ein Prototyp, der auf einer generischen konzeptuellen Methode basiert, welche unabhängig von der Ausführbarkeit einer Plattform ist. Darüber hinaus gibt es keine vordefinierte Granularitätsebene und die Modellobjekte sind für die unterschiedlichen Phasen der Datenintegration Prozess wiederverwendbar.

Der Prototyp wurde auf der Open Model Plattform eingesetzt. Die Open Model Plattform ist eine Initiative der Universität Wien mit dem Ziel die Verwendung von Modellierungsmethoden zu erweitern und diese durch das Rahmenwerk, welches alle mögliche Modellierungsaktivitäten beinhaltet, für Geschäftsdomäne zur Verfügung zu stellen und nützlich zu machen, um die Zugänglichkeit bei den Anwendern zu steigern.

² Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.80ff.

Abstract

Today many organizations are operating in dynamic and rapid changing environment and highly competitive markets. Consequently fast and accurate fact-based decisions can be an important success factor. The basis for such decisions is usually business information as a result of business intelligence and business analytics in the corporate associations.

One of the challenges of creating high-quality information for business decision is to consolidate the collected data that is spread in multiple heterogeneous systems throughout the organization in one or many different locations. Typically ETL-processes (**E**xtraction, **T**ransforming and **L**oading) are used to merge heterogeneous data from one or more data sources into a target system to form data repositories, data marts, or data warehouses. Due to the lack of a common methods or approaches to systematically manage such ETL processes and the high complexity of the task of integrating data from multiple sources to one common and unified view, it is difficult for both professionals and less experienced users to successfully consolidate data. Currently the analysis process is often performed without any predefined framework and is rather based on informal basis than a scientific methodology.

Hence, for commercial tools that are supporting the data integration process including visualization of the integration, the reuse of analyses sequences and the automatic translation of the visual description to executable code, the major problem is that metadata used for data integration in general is only employed for representation of syntactic knowledge. Semantic information about the data structure is typically only available in a rudimentary form though it plays a significant role in defining the analysis model and the evaluation of the results.

With this background Grossmann developed a “Conceptual Approach for Data Integration for Business Analytics”³. It aims to support professionals by making business analytics easier and consequently more applicable to less experienced user in different domains. The idea is to incorporate detailed knowledge about the data in business analytics, especially information about semantics. It focuses on the inclusion of a more structured description of the transformation process in business analytics in which information about dependencies and side effects of the algorithms are included. Furthermore the approach incorporates the concept

³ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009.

of meta-modelling⁴; it presents a framework including the modelling concepts for data integration for business analytics.

The idea of the thesis at hand is to develop a meta-model prototype that supports Data Integration for Business Analytics based on Grossman's approach. The paper focuses on the intellectual process of transforming the theoretical method into a conceptual model which can be applied to the framework of a modelling methods and which fits to the specific concepts of a meta-model platform used. The result is a prototype based on a generic conceptual method which is execution platform independent, there are no pre-defined granularity levels and the objects of the model are re-usable for the different phases of the data integration process.

The prototype is deployed on the Open Model Platform, an initiative started at the University of Vienna that aims to extend the usage of modelling methods and models and to make it more accessible to users by offering a framework including all kinds of modelling activities useful for business applications.

⁴ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.80ff.

Table of Abbreviations

BA	Business Analytics
BI	Business Intelligence
CRIPS-DM	Cross-Industry Standard Process for Data-Mining
ER	Entity Relationship
ETL	Extraction, Transform, Load
GAV	Global as View
GUI	Graphical User Interface
IS	Information System
IT	Information Technology
LAV	Local as View
LEO	Leonine Editor with Outlines
OLAP	Online Analytical Processing
OMG	Object Management Group
SEMMA	Sample, Explore, Modify, Model, Assess
SQL	Structured Query language

Table of Figures

Figure 1: Original schema A	15
Figure 2: Original schema B	15
Figure 3: Replacing entity „Keyword“ to „Topics“	15
Figure 4: Make Publisher into an entity	15
Figure 5: Integrated schema	16
Figure 6: Creation of a subset relation	16
Figure 7: Dimensions of information integration	17
Figure 8: Example of structural heterogeneity	20
Figure 9: Architecture of systems with materialized and virtual data integration	21
Figure 10: Five-level schema architecture of an FDBS	23
Figure 11: Mediated Architecture	24
Figure 12: Peer Data Management System	25
Figure 13: Classification of schema matching approaches	29
Figure 14: Full vs. partial structural match (example)	30
Figure 15: Interaction between concepts, the real world things and symbols.....	36
Figure 16: Representation of different model types	42
Figure 17: Modelling methods and mechanisms & algorithms	43
Figure 18: Syntax, semantics and notation of the Entity Relationship Modelling Language ..	44
Figure 19: Formalization of the modelling process	47
Figure 20: Generic architecture of meta-modelling platforms	47
Figure 21: Information categories in Business Analytics	54
Figure 22: Architecture Levels and Responsibilities	58
Figure 23: Modelling procedure	59
Figure 24: Methodological Guideline as Modelling Procedure	61
Figure 25: Model hierarchy	62
Figure 26: Class diagram of the data meta-model.....	63
Figure 27: Class diagram of process meta-model	65
Figure 28: Modelling language	66

Table of Tables

Table 1: Data model, modelling language overview.....	67
Table 2: Process model, modelling language overview	69
Table 3: Class Population – modelling language details.....	71
Table 4: Class Observable Units – modelling language details	73
Table 5: Class Data Set – modelling language details	75
Table 6: Class Variables – modelling language details.....	77
Table 7: Class Value Domain – modelling language details.....	79
Table 8: Class Additional Attributes – modelling language details	81
Table 9: Class Overall Goal – modelling language details	83
Table 10: Class Process Goal – modelling language details	85
Table 11: Class Method – modelling language details	87
Table 12: Class Criteria – modelling language details.....	89
Table 13: Class Evaluation – modelling language details.....	91
Table 14: Class Operation – modelling language details	93
Table 15: Class Result – modelling language details.....	95
Table 16: Class Decision – modelling language details.....	97
Table 17: Data model, algorithms and mechanisms overview.....	98
Table 18: Possible algorithms and mechanisms overview	100
Table 19: Algorithm for class Value Domain for Data Model	103
Table 20: Algorithm for class Result for Process Model	106

1 Introduction

1.1 Purpose and Goal of the Thesis

The goal of this master thesis is to create a meta-model prototype based on a pre-defined modelling method that supports data integration for business analytics. The report focuses on the elaboration of the translation of the method on the meta-model platform, in other words how the method was implemented using a meta-modelling platform. Grossman's work *Data Integration for Business Analytics: A Conceptual Approach* ⁵ is the basis for the development of the prototype and it is realised on the ADOxx[®] meta-modelling platform.

The purpose of this master thesis was to gain insight of the intellectual process of turning a pre-defined scientific method into a meta-model on a meta-model platform, also referred to as "conceptualization". A further purpose is to contribute to the Open Model community by the development and deployment of a meta-model prototype for the community platform.

1.2 Organisation of the Master Thesis

This thesis is arranged in two main parts; based on a theoretical background in the first half a presentation of the result of the implementation of a meta-model Prototype follows in the second part.

The theoretical part consists of three chapters; a theoretical background of the topics of business analytics and data integration, an introduction to meta-modelling within the Open Model Initiative and a presentation of the conceptual method for data integration. The theoretical background gives the definitions of business analytics and business integration and thereafter it describes data integration in more detail, including its causes, architectures and approaches. Chapter 3 introduces the reader to the topic of meta-modelling and starts with definitions and continues to present the advantages of meta-modelling and focuses on the approaches for meta-modelling. This chapter concludes with a brief overview on the roles needed to use and administer of a meta-modelling platform. The last chapter of the theoretical part, 4 *A Conceptual Method for Data Integration* describes the ideas behind the conceptual method for data integration for business analytics building the basis for the prototype.

The second part is describing the prototype implementation. Chapter 5 *Prototype Implementation on the ADOxx[®] Platform* starts with "conceptualization", a presentation of the general process of turning a theoretical method into a meta-model and it is followed by the

⁵ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009.

elaboration of the conceptualization of the actual method to support data integration for business analytics. The main focus of this chapter lies on the presentation of the implementation of the three modelling methods parts: modelling procedure, modelling language and the mechanisms and algorithms.

As a conclusion the objectives and results of the master thesis are summarized.

2 Theoretical Background

This chapter describes the theoretical framework to support data integration for business analytics. The first chapter will explain the term business analytics and address different views on this concept in order to give a brief contextual overview. Further, data integration as a prerequisite for data analytics is defined. Thereafter the following chapters provides a more detailed presentation of data integration by illustrating architecture, approaches, state of the art, challenges and current research.

2.1 Definition and Terminology

2.1.1 Definition: Business Intelligence and Business Analytics

This chapter aims to define the term Business Intelligence (BI) and Business Analytics (BA). There are a vast number of explanations, definitions, interpretations and publications referring to BI and BA, this paper will only reflect a few, focusing on showing some different approaches to the topic.

The concept of Business Intelligence was introduced 1989 by Howard Dresner, a Gartner Group Analyst and was describes as *"concepts and methods to improve business decision making by using fact-based support systems."*⁶ Among the various numbers of definitions that have evolved since then, a common view of the primary purpose of BI, supporting decision making in the organization⁷ is found, but the approach to reach such goal can be quite different. It ranges from a predominantly technical view, a more holistic view in which a combination of information systems (IS) as well as processes and methods are taken into consideration to definitions focusing only on the decision making not including any technical perspectives.

An example of a definition which reflects a merely technical point of view is Moss. *"BI is neither a product nor a system. It is an architecture and a collection of integrated operational as well as decision-support applications and databases that provide the business community easy access to business data."*⁸ Examples of BI definitions which only focus on the decision making but lacking the technical view are found by Williams: *"...business information and business analysis within the context of key business processes that lead to decisions and*

⁶ Power, D.J.: A Brief History of Decision Support Systems, 2007, online in WWW at <http://dssresources.com> (<http://goo.gl/dj9N1>), accessed on 20.05.2010.

⁷ Isik, O.: Business Intelligence Success: An Empirical Evaluation of the Role of BI Capabilities the Decision Environment, 2010, online in WWW at <http://digital.library.unt.edu/> (<http://goo.gl/Qm2bt>), accessed on 20.11.2010, p.11ff.

⁸ Moss T. L., Atre S.: Business Intelligence Roadmap: the complete project lifecycle for decision-support applications, Addison-Wesly, 2003, p.4.

*actions that result in improved business performance.”*⁹ Also Wells focus on the decision making without consideration of the technical perspective; *“Business intelligence is the ability of an organization or business to reason, plan, predict, solve problems, think abstractly, comprehend, innovate and learn in ways that increase organizational knowledge, inform decision processes, enable effective actions and help to establish and achieve business goals.”*¹⁰

In Gartner’s Glossary of Information Technology Acronyms and Terms from 2004, the definition of Business Intelligence falls into the category of a holistic approach. It is referred to as *“An interactive process for exploring and analyzing structured, domain-specific information (often stored in data warehouses) to discern business trends or patterns, thereby deriving insights and drawing conclusions. The business intelligence process includes communicating findings and effecting change. Domains include customers, suppliers, products, services and competitors”*¹¹

Loshin’s definition is a combination of processes and IS and it is described as *“... the processes, technologies and tools needed to turn data into information, information into knowledge and knowledge into plans that drive profitable business actions. Business Intelligence encompasses data warehousing, business analytic tools and content and knowledge management.”*¹²

2.1.2 Definition: Data Integration

Regardless of the approach of the terms Business Intelligence and Business Analytics, it is undisputable that without data either BI or BA would exist. In almost every corporate organisation data is collected and generated in multiple information systems which are often indispensable. In many business areas, for example business analytics and customer relationship, data from more than one application are more or less inevitable and integrated data with one single access point is very beneficial.

⁹ Williams N., Williams S.: The Profit Impact of Business Intelligence, Elsevier Inc., 2007, p.2.

¹⁰ Wells D.: Business Analytics – Getting the Point, 2010, online in WWW at <http://www.b-eye-network.com/> (<http://goo.gl/F9Yup>), accessed on 22.05.2010.

¹¹ Gartner: The Gartner Glossary of Information Technology Acronyms and Terms, 2004, online in WWW at <http://www.gartner.com> (<http://goo.gl/3JOxh>), accessed on 21.08.2010, p.48.

¹² Loshin D.: Business Intelligence, The Savy Manager’s guide, Getting Onboard with Emerging IT, Morgan Kaufmann, 2003, p.6.

Data integration refers to the problem of combining data from heterogeneous sources, and the provision of a homogenous unified view to the user¹³. Furthermore data integration also aims to deliver a non-redundant view or at least a low grade of redundant data.

2.2 Data Integration

The previous chapter illustrated the definitions of the terms business analytics and data integration. In this chapter the area of data integration is examined in more detail. As an introduction, an example illustrating the problem of data integration and its causes is presented. Thereafter two different types of integration, the materialized and the virtual integration are explained and the examination of the core problem with data integration is continued. In the last sections different integration architectures and an overview of different approaches to the data integration problem are presented, concluding with an outlook of the challenges with data integration and current research.

2.2.1 Causes of Data Integration

In order to give the reader an impression of the problem of data integration an example based on the work of Batini and Lenzerini¹⁴ is given. Two schemas as depicted in Figure 1 and Figure 2 should be merged into one unified schema. In this example the words “Topics” and “Keyword” belong to the same concept and since the two schemas should be merged, the names need to be unified. In order to prepare for the merger of the schemas the name “Keyword” in schema B is replaced by the word “Topic” as reflected in Figure 3.

The element “Publisher” represents another difference between the two source schemas, in the first schema it is an entity and in the second schema it is an attribute. In order to merge schema A and B these two representations need to be conformed. A transformation of the attribute Publisher to an entity with a new attribute Name is performed, seen in Figure 4.

¹³ Dittrich K., Ziegler, R.: Three Decades of Data Integration - All Problems Solved?, 2004, online in WWW at <http://www.uzh.ch> (<http://goo.gl/PfFCk>), accessed on 05.10.2010, p.3.

¹⁴ Batini C., Lenzerini M.: A Comparative Analysis of Methodologies for Database Schema Integration, 1986, online in WWW at <http://www.ubc.ca> (<http://goo.gl/45OIa>), accessed on 21.10.2010, p.329ff.

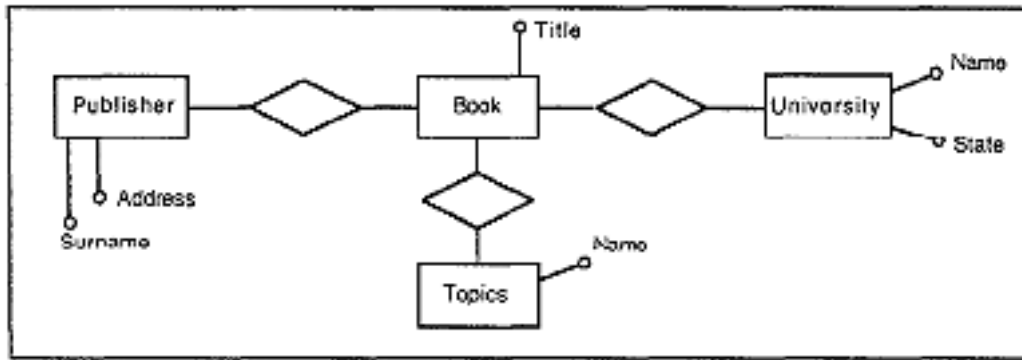


Figure 1: Original schema A

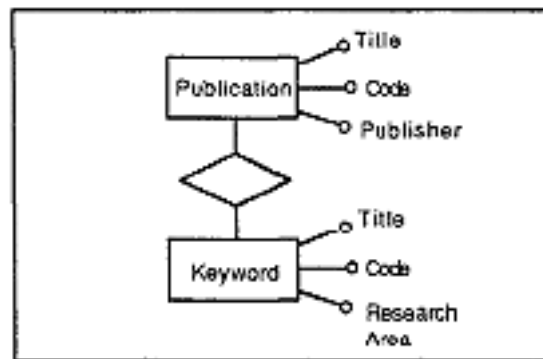


Figure 2: Original schema B

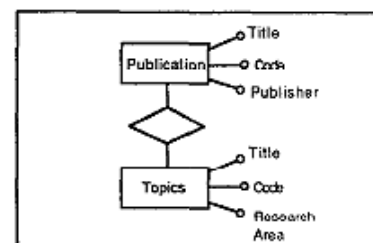
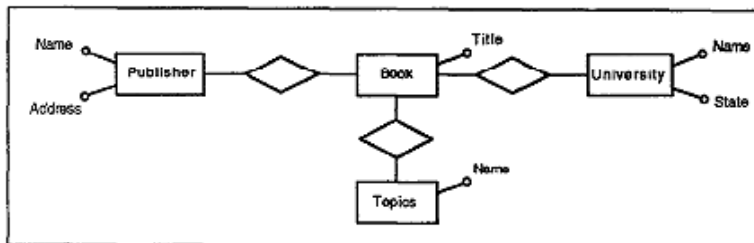


Figure 3: Replacing entity „Keyword“ to „Topics“

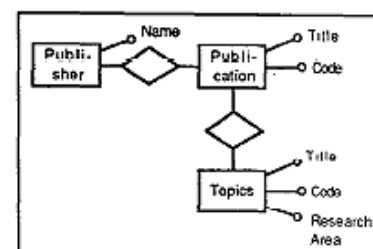
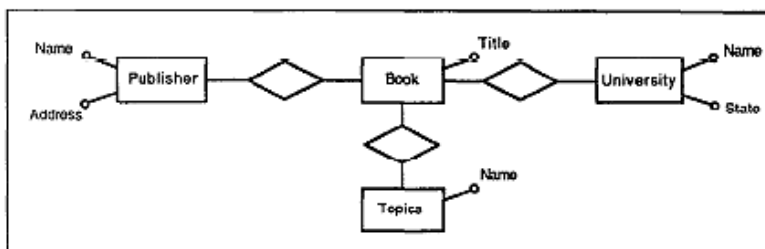


Figure 4: Make Publisher into an entity

After these adaptations it is possible to merge the two schemas and the result is depicted in Figure 5.

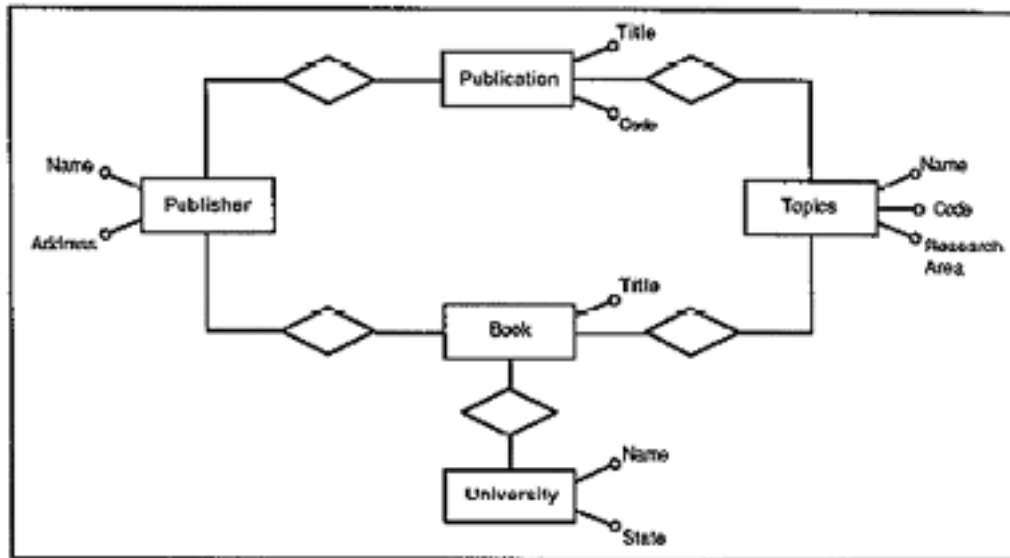


Figure 5: Integrated schema

Still the integration is not yet fulfilled; properties that relate concepts belonging to different schemas must be identified. The subset relationship between the concepts “Book” and “Publication” is such a case and this is added to the schema illustrated in Figure 6.

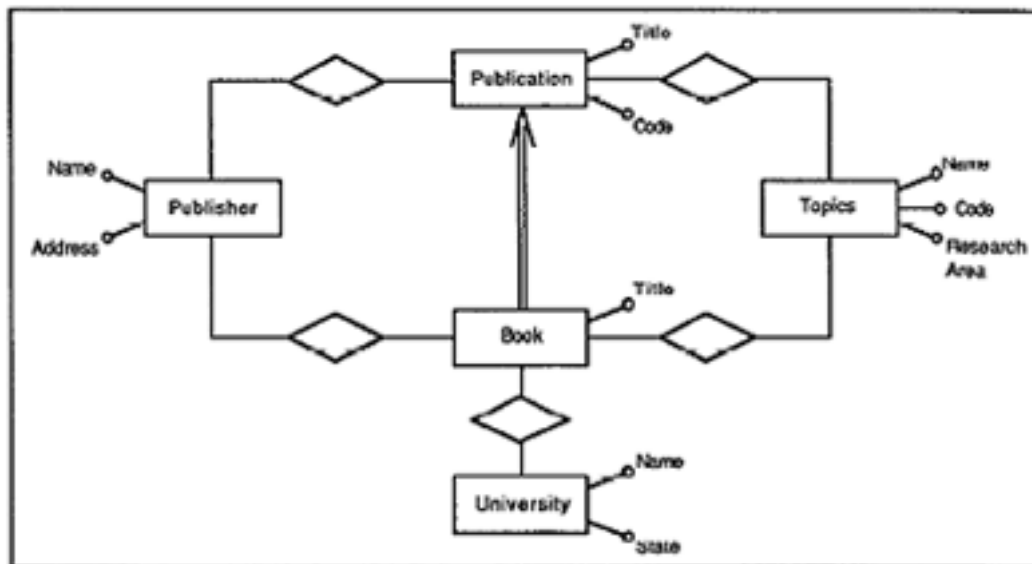


Figure 6: Creation of a subset relation

This very much simplified example highlights some of the basic problems; semantic and structural heterogeneity, involved with data integration.

2.2.2 Distribution, autonomy and heterogeneity

The cornerstone of the data integration problem is heterogeneity in the source systems and in many cases this is caused by the physical distribution of the data and the autonomy of the

systems holding the data. The problem with heterogeneity, distribution and autonomy is referred to as orthogonal dimensions¹⁵. Each problem can occur independently but usually the dimensions are strongly connected to each other.

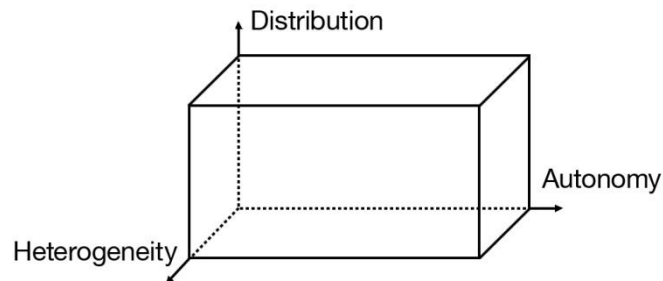


Figure 7: Dimensions of information integration¹⁶

Data Distribution

The problem with data distribution means that the data is physically distributed over multiple systems. Data can be stored on one single computer or spread over different geographical distributed computers which are connected to each other within a network. The reason for distributing data over different locations is increased availability and reliability as well as enhanced access times¹⁷.

Physical distributed data may impose problems when performing data integration. When data is spread over multiple computers, difficulties identifying the physical locations, such as computers, server and ports, might occur. A problem with physical distribution is also that it always implies different schemata and most traditional query languages do not have the ability to handle this¹⁸.

The data can also be distributed logically and as opposed to the physical distribution, where the relations are kept intact, the logical distribution implies that the tables are divided in either vertical, horizontal or hybrid partitions¹⁹. A table that is divided by the rows reflects a

¹⁵ Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.50.

¹⁶ cf. Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.50.

¹⁷ Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990, p.185.

¹⁸ Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.17.

¹⁹ Rababaah H.: Distributed databases, fundamentals and research, 2005, online in WWW at <http://www.cs.iusb.edu> (<http://goo.gl/USU3E>), accessed on 26.09.2010, p.7ff.

horizontal fragmentation and a table which divided by its columns is a vertical fragmentation. A hybrid partition is a combination of a vertical and horizontal fragmentation.

Autonomy

In the context of data integration autonomy represents the independent and separate administration and access control of the data sources. Usually this is found when systems from independent companies should be merged but this can also be an issue for various systems within the same organization. According to Leser and Neuman there are four main categories of autonomy; design autonomy, interface autonomy, access autonomy and legal autonomy²⁰.

Design autonomy refers to the freedom how to manage the data. Decisions about format, data model, schema, syntactical design, and the usage of keys and conceptualization or semantic interpretation can be made independently by the data source system. Design autonomy causes the most difficult types of data integration; structural and semantically heterogeneity.

Interface autonomy addresses the topic about the freedom for each data source deciding about how the data technically can be accessed, such as the type of protocol and which query language should be used. The interface autonomy is closely connected to the design autonomy because the type of the data representation also determines or can restrict the technical access.

Access autonomy reflects the freedom to decide who can access the system and handles authentication and authorization topics and allocates read and write rights for specific data or data domains.

Legal autonomy denotes the right for a data source to prohibit integration of the data, for example due to copyrights.

Further kind of autonomy mentioned in the literature is communication, execution and association autonomy²¹. A system with *communication autonomy* decides if and when to communicate with other system. This is a temporary extension of the access autonomy. The *execution autonomy* refers to the ability for the system to control when and in which order to execute operations from external systems. *Associated autonomy* means that the systems can make decisions about sharing its functionality, such as which operations it supports, and

²⁰ Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.54ff.

²¹ Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990, p.188-189.

resources, for example the data it manages. Associated autonomy can be regarded as part of the design autonomy or as an autonomy in its own right.

Heterogeneity

A heterogeneous system reflects a system in which methods, models and structure for data access are not identical. And for data integration heterogeneous data sources are a core problem. As mentioned in the previous sections, distributed systems and autonomy can cause major data integration problems. Distributed systems are in many cases administrated and further developed by different persons and this usually leads to heterogeneous systems. Heterogeneity is also often strongly connected to autonomy systems and a higher grade of autonomy normally causes a higher grade of heterogeneity²².

A various number of different approaches to classify heterogeneity appear in the literature^{23, 24, 25, 26}. Typical categories or subcategories often mentioned are semantic, structural, schematic as well as data and data model heterogeneity, but there is no common understanding or classification of these terms in the context of heterogeneity for systems holding data. Apart from differences regarding technical issues such as query possibilities, query languages and communication protocols, basically the problem with heterogeneity arises by differences in the underlying data model or representation of the data and the type of conflicts that occurs can be at the structural and/or semantic levels²⁷.

Different data models can be applied to describe different views of the data, for example one model can describe the conceptual view of the data and another model can represent the physical view of the data. Systems using different data models for the same data view reflect heterogeneity in the data model, and this result in different schemata for the models which causes problem for data integration. For example an object-oriented model can handle concepts like generalization and specification but relational model can't²⁸. There are various

²² Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.58-59.

²³ Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.58ff.

²⁴ Koch C.: Data Integration against Multiple Evolving Autonomous Schemata, 2001, online in WWW at <http://www.csd.uoc.gr> (<http://goo.gl/afmsi>), accessed on 11.10.2010, p.40-41.

²⁵ Härder T., Sauter G., Thomas J.: The Intrinsic Problems of Structural Heterogeneity and an Approach to their Solution, in the VLDB Journal 8:1, pp. 25-43, 1999, p.26-27.

²⁶ Dou D., Liu H.: An Exploration of Understanding Heterogeneity through Data Mining, in Proceedings of KDD'08 Workshop on Mining Multiple Information Sources (MMIS 2008), pp. 18-25, 2008, p.1ff.

²⁷ Hammer J., Pluempitiwiriyawej C.: A Classification Scheme for Semantic and Schematic Heterogeneities in XML Data Sources, 2000, online in WWW at <http://www.cise.ufl.edu> (<http://goo.gl/2xgAr>), accessed on 08.10.2010, p.7.

²⁸ Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.66.

difficulties translating one data model to another, for example a relational model can easily be transformed into an object-orientated model but the other way around is much harder²⁹.

Leser and Naumann³⁰ describe the structural heterogeneity as extract from the real world that can modeled in the same model but in different ways. For instance an element of the data model can be reflected as a relation, an attribute or as a value of an attribute. The following examples should demonstrate the mentioned possibilities.

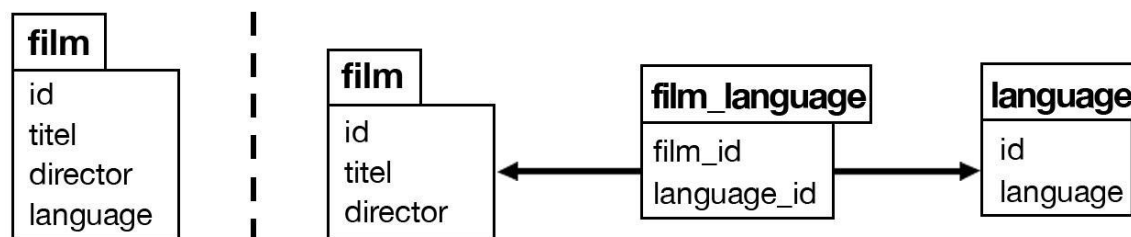


Figure 8: Example of structural heterogeneity³¹

The understanding of classification of semantic heterogeneity is still evolving and there are many interpretations found in the database community. According to Larsson and Seth semantic heterogeneity occurs when “... *there is a disagreement about the meaning, interpretation, or intended use of the same or related data.*”³² The main reason for semantic conflicts is because people think differently and therefore a lack of common domain specific knowledge arises.

2.2.3 Materialized and Virtual Integration

In the context of data integration it is common to distinguish between materialized and virtual integration. Materialized data integration implies that data from the sources are extracted and saved physically in the integration system, in one central location. The data remains in the sources but once available in the integration system the source data is not addressed anymore. An example of materialized data integration is data warehouses.

²⁹ Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Dataquellen, dpunkt.verlag, 2007, p.66.

³⁰ Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Dataquellen, dpunkt.verlag, 2007, p.67.

³¹ cf. Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Dataquellen, dpunkt.verlag, 2007, p.70.

³² Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990, p.187.

Data used in a virtual integration is only physically stored in the original source and is used on the fly when needed. The integrated data only exists virtually for the limited time. Mediated integration systems are an example for virtual integration.

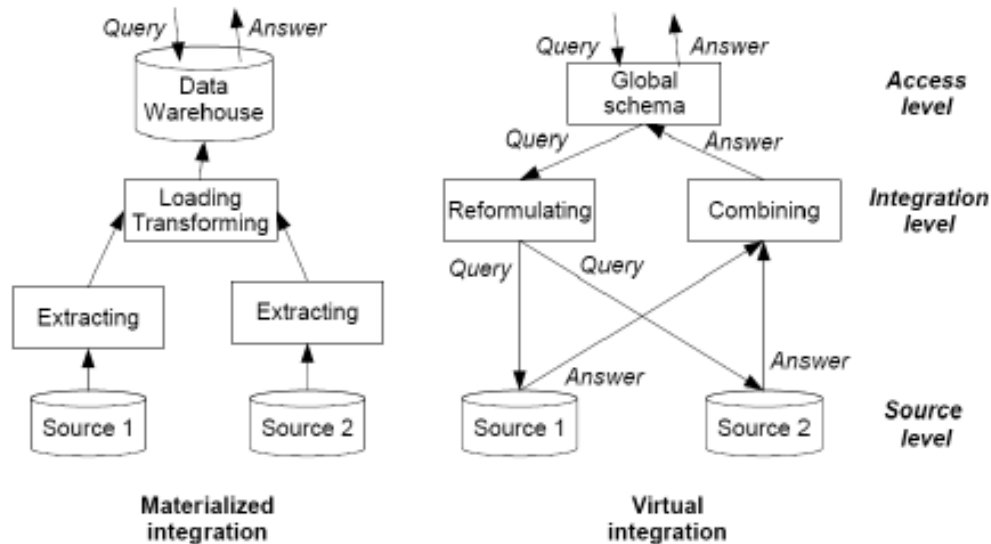


Figure 9: Architecture of systems with materialized and virtual data integration³³

2.2.4 Architectures

With the goal of integrating one or more source systems into a non-redundant unified view of data, a number of different integration system architectures have been developed over the years. As mentioned in the previous chapter distribution, autonomy and heterogeneity have an impact on integration architecture and based on this different implementation alternatives are applied, such as federated databases, mediators, peer systems and ontology-based system. These architectures will be briefly presented in the following sections.

Federated Databases

This section is based on the work of Larsson and Seth³⁴ as well as Leser and Naumann³⁵. The term “Federated Database System” describes a system which consists of autonomous database components participating in a federation with each other. Federated database systems can be considered as compromise between total lack of integration, where users explicitly have to access data from multiple sources, and total integration, in which the data sources have no

³³ Gawinecki M.: How schema mapping can help in data integration?, 2009, online in WWW at <http://www.agentgroup.unimore.it> (<http://goo.gl/EkR6w>), accessed on 20.10.2010, p.1.

³⁴ Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990, p.183ff.

³⁵ Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.94ff.

autonomy and the user can access the data from one single interface but have no possibility of local access to the individual sources. The members of a federated system have control over their own data that they manage and only partial controlled data is shared in the federation. This implies that there is no centralized control and the components rule over the access to their data.

Federated database systems provide two types of operations in order to enable controlled sharing but at the same time keep the components' autonomy. Global operations reflect the access of data that may be managed by multiple members of the federation. Local operations means that a user can access the local database directly but only data of the local component will be available. Federated database systems can be categorized in two different types; *tightly* and *loosely coupled*. Loosely coupled systems are sometimes also referred to as interoperable systems or multi-database systems. These systems do not have a central administration the control and maintenance of the federation are enforced by its local users. If the system is created and maintained by the federation and its administrator, it is referred to as a tightly coupled system. The data access controls of each member components are also controlled by the federation and its administrators.

The major attribute of a tightly coupled federated database system is the use of a global conceptual schema, also referred to as federated schema. This schema acts as a solid reference for all external schemas and its applications. There are two possibilities creating a federated schema, either consolidating the existing export schemas or creating it independently from the local schemas.

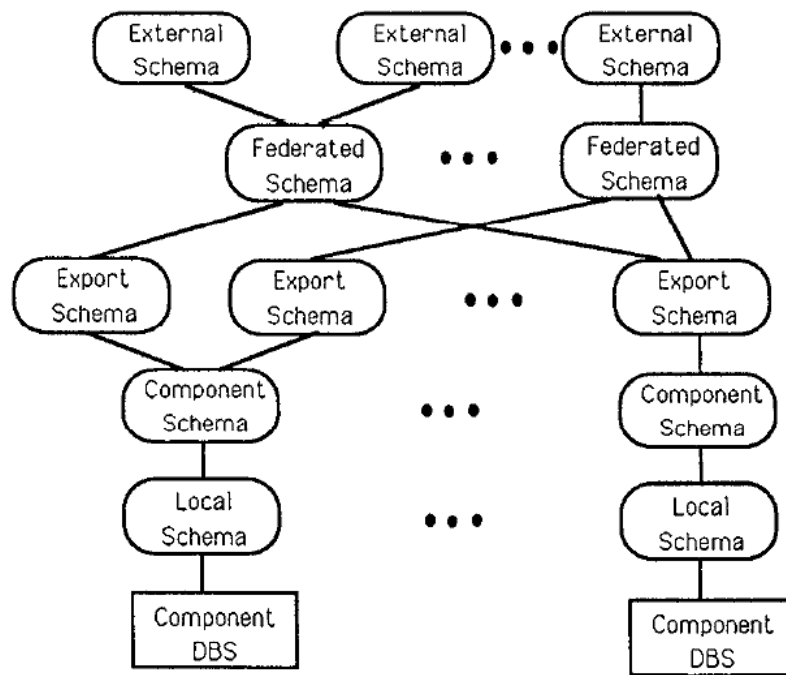


Figure 10: Five-level schema architecture of an FDBS³⁶

Multi Database System

The overview of multi database systems is based on Leser and Naumann³⁷. The multi database system is a collection of several autonomous databases that are loosely coupled. The involved databases provide external views from which the data can be accessed via a multi database language. Such language allows the user to access the distributed databases in one query. Significant for a multi database is the absence of a global conceptual schema. Each individual database has its own export schema, which defines part of the local schema that is available in the export schema. In most cases the databases included in the multi database system use the same data model. If data model heterogeneity exists then the local data source or the multi database query language must offer a translation to the global schema.

Mediators

Mediators are a generalized description of the federated systems and are characterized by two prominent roles; wrappers and mediators. The wrapper is an interface between mediator and data source. It translates queries from the mediator into executable queries for the data sources and it transforms data from the data sources into a representation in the data model of the

³⁶ Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990, p.199.

³⁷ Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.93.

mediator³⁸. The wrappers deal with technical interoperability problems and heterogeneity in the interface, data model and the schema. The essential task of the mediator is to transform the end user's queries to queries that can be executed by the wrappers. This implies dealing with structural and semantic differences.

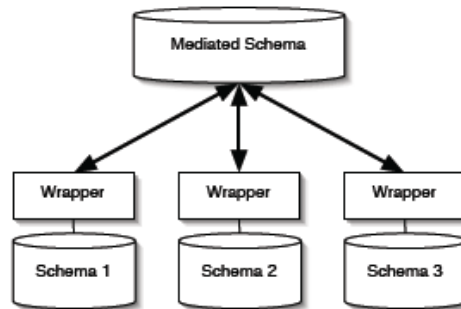


Figure 11: Mediated Architecture³⁹

Peer Data Management Systems

As opposed to the previous presented data integration system architectures, a peer data management system does not reside in a central logical schema or a centralized management⁴⁰. The term is based on the ideas of the Peer-to-Peer (P2P)-systems⁴¹ in which tasks or workloads are distributed between peers. The PDMS is a collection of autonomous data sources which are linked to other data sources or so called nodes or peers. The peers offer data, schema and mappings for sharing with other peers. The peers define their own schema and if it is a new schema, the peer must provide a mapping to the schemas of the other peers that already exist in the system⁴². The motivation for using a PDMS is increased scalability and flexibility and the possibility to react on dynamic changes in the system.

³⁸ Leser U.: Query Planning in Mediator Based Information Systems, 2000, online in WWW at <http://www2.informatik.hu-berlin.de> (<http://goo.gl/HyLpv>), accessed on 18.10.2010, p.52.

³⁹ Cudré-Mauroux P.: Peer Data Management system, 2008, online in WWW at <http://www.csail.mit.edu> (<http://goo.gl/f7Gmo>), accessed on 18.10.2010, p.1.

⁴⁰ Cudré-Mauroux P.: Peer Data Management system, 2008, online in WWW at <http://www.csail.mit.edu> (<http://goo.gl/f7Gmo>), accessed on 18.10.2010, p.1.

⁴¹ Leser U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.101.

⁴² Cudré-Mauroux P.: Peer Data Management system, 2008, online in WWW at <http://www.csail.mit.edu> (<http://goo.gl/f7Gmo>), accessed on 18.10.2010, p.1.

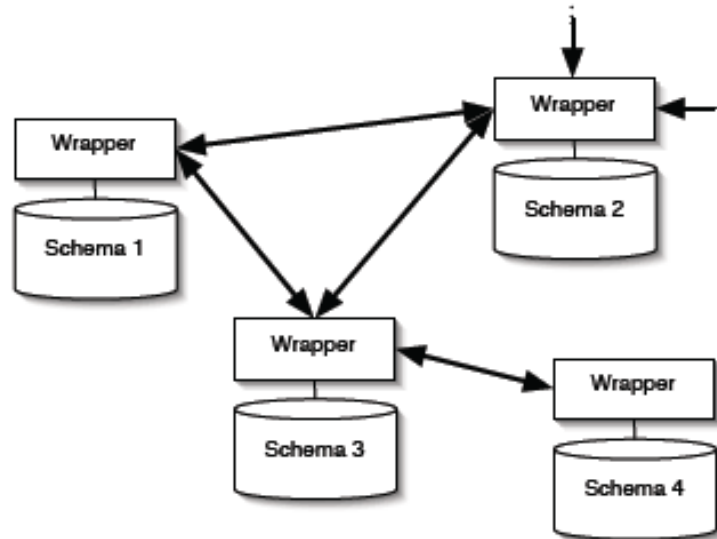


Figure 12: Peer Data Management System⁴³

2.2.5 Approaches

This section provides an introduction to the major techniques used for data integration. A detailed presentation of the extensive research available in this field would go beyond the scope of this thesis, so this chapter will focus on outlining the major features of these approaches. The process of schema integration is considered as part of the data integration process and has been a major subject of research for over 20 years. This chapter starts with a formal definition of the components of a data integration system and then proceeds with a description the different phases of the schema integration process. It then focuses on the most prominent phase of this process; finding correspondence between schema elements, known as schema matching. Additional to schema matching approaches, multi database languages and ontology are briefly presented.

Schema integration refers to the process of integrating multiple schemas into one global unified conceptual schema and is considered as part of the data integration process. The main components of a data integration system using a mediated schema are the global schema, the sources and the mappings. Based on Lenzerini's⁴⁴ work it is formalized as follows: I is representing the data integration system existing of the triple $\{G, S, M\}$.

⁴³ Cudré-Mauroux P.: Peer Data Management system, 2008, online in WWW at <http://www.csail.mit.edu> (<http://goo.gl/f7Gmo>), accessed on 18.10.2010, p1.

⁴⁴ Lenzerini M.: Data Integration: A Theoretical Perspective, 2002, online in WWW at <http://www.cs.ubc.ca> (<http://goo.gl/WJfRv>), accessed on 26.10.2010, p.2.

- \mathcal{G} reflects the global schema and is expressed in a language $\mathcal{L}_{\mathcal{G}}$ over an alphabet $\mathcal{A}_{\mathcal{G}}$. Each symbol G is represented in the alphabet (for example relations if G is relational and objects if G is object-orientated).
- \mathcal{S} reflects the sources and is expressed in a language $\mathcal{L}_{\mathcal{S}}$ over an alphabet $\mathcal{A}_{\mathcal{S}}$. The alphabet $\mathcal{A}_{\mathcal{S}}$ comprises a symbol for each element of the sources.
- \mathcal{M} is the mapping between \mathcal{G} and \mathcal{S} holding a set of assertions of the forms

$$q_{\mathcal{S}} \rightsquigarrow q_{\mathcal{G}}$$

$$q_{\mathcal{G}} \rightsquigarrow q_{\mathcal{S}}$$

where $q_{\mathcal{S}}$ and $q_{\mathcal{G}}$ represent two queries respectively over the source schema \mathcal{S} and the global schema \mathcal{G} . Queries over the source schema are represented in the a query language $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ over the alphabet $\mathcal{A}_{\mathcal{S}}$ and the queries $q_{\mathcal{G}}$ over the global schema is represented by the query language $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ over the alphabet $\mathcal{A}_{\mathcal{G}}$. $q_{\mathcal{S}} \rightsquigarrow q_{\mathcal{G}}$ is an assertion which specifies the concept of the queries $q_{\mathcal{S}}$ over the sources correspond to the concept of the global schema expressed by the queries $q_{\mathcal{G}}$.

The source schema illustrates the structure of the sources where the real data is kept. The global schema is a virtual view of the integrated and adjusted sources. The correspondence between the elements of the source schema and the global schema is reflected in the assertion in the mapping.

The process of data integration is divided into four steps according to Batini and Lenzerin⁴⁵ and will be described below:

Pre-Integration

It starts with the pre-integration that includes an analysis of the schemas before the integration. This is done in order to decide on an integration strategy including decisions on which schemas to be used, order of integration and if the entire schema should be integrated or only parts of it. Further decisions in this phase, if applicable, are the number of designers active in the integration work and the number of schemas that should be integrated at one time. Additional information such as the type of implementation or constraints is also considered to be part of this phase.

⁴⁵ Batini C., Lenzerini M.: A Comparative Analysis of Methodologies for Database Schema Integration, 1986, online in WWW at <http://www.ubc.ca> (<http://goo.gl/45OIa>), accessed on 21.10.2010, p336ff.

Comparison of Schemas

This step deals with comparing and analyzing schemas in order to find correspondence among concepts and to identify possible conflicts.

Conforming Schemas

If conflicts are detected in the previous step an effort to resolve these conflicts are done in the third step. This is not an easy endeavor and must usually be done manually in cooperation between designers and domain users.

Merging and Reconstructing

In the fourth and final phase of the schema integration the source schemas are integrated to some intermediate schema. The result is analyzed and depending on if the quality criteria is met or not the schema might need to be reconstructed. The quality criteria are;

- Completeness and Correctness:

All concepts available in any of the component schemas must be existent in the global schema. The concepts in the integrated schema must have the same meaning as in the local schema and no semantic conflicts are allowed.

- Minimality:

A semantic concept that is represented in multiple source schemas must only occur once in the integrated target schema.

- Understandability:

Designers and end users must easily understand the integrated schema.

2.2.5.1 Schema Matching and Schema Mapping

Comparison of schemas can also be referred to as schema matching which is the process of identifying semantic matches between elements of two or more schemas⁴⁶. The purpose of schema matching is to integrate heterogeneous sources into one global unified conceptual schema⁴⁷. It can be used for merging databases, establishing a materialized view or enabling queries to be executed on a virtual view of multiple, heterogeneous sources. A high degree of implicit semantic domain knowledge of the database schemas to be matched is required and

⁴⁶ Blake R.: A Survey of Schema Matching Research, 2007, online in WWW at <http://www.umb.edu> (<http://goo.gl/GT0ue>), accessed on 30.10.2010, p.2.

⁴⁷ Gal A., Trombetta A.: A Model for schema Integration in heterogeneous Databases, 2003, online in WWW at <http://www1.technion.ac.il> (<http://goo.gl/IGjsN>), accessed on 21.10.2010, p.1.

therefore schema matching is generally performed manually. This can be an error-prone and time consuming and usually an expensive process⁴⁸. When correspondences between elements of the source and target schemas are established by applying schema matching the next step is usually to revolve the result of the matching into mapping queries. This is usually referred to as schema mapping and is together with schema matching a common task within the data integration process.⁴⁹ This chapter outlines an overview of different schema matching approaches followed by an introduction to schema mapping.

Bernstein and Rahm⁵⁰ have classified the major approaches for schema matching approaches and also named a few sample approaches shown in Figure 13. Depending on the application domain and the schema types the applicable matching algorithms are selected, either individual match algorithms that calculate a mapping based on a single criterion or a combination of multiple match algorithms can be used. When using multiple matchers either multiple matching criteria can be used within an integrated hybrid matcher or multiple matching results which are generated by different match algorithms can be combined within a so called composite matcher. For individual matchers the following - largely - orthogonal classification criteria is given:

- *Instance vs. Schema*: matching approaches can consider instance data (i.e., data contents) or only schema-level information.
- *Element vs. Structure Matching*: matching can be performed for individual schema elements, such as attributes, or for combinations of elements, such as complex schema structures.
- *Language vs. Constraint*: a matcher can use a linguistic based approach (e.g., based on names and textual descriptions of schema elements) or a constraint-based approach (e.g., based on keys and relationships).
- *Matching Cardinality*: the overall match result may relate one or more elements of one schema to one or more elements of the other, yielding four cases: 1:1, 1:n, n:1, n:m. In addition, each mapping element may interrelate one or more elements of the two schemas. Furthermore, there may be different match cardinalities at instance level.

⁴⁸ Manakanatas D., Plexousakis D.: A Tool for Semi-Automated Semantic Schema Mapping: Design and Implementation, 2006, in WWW at <http://www.ics.forth.gr> (<http://goo.gl/t9eVG>), accessed on 21.10.2010, p.1

⁴⁹ Bohannon p., Elnahrawy E., et al: Putting Context into Schema Matching, 2006, online in WWW at <http://www.vldb.org/> (<http://goo.gl/G5bPY>), accessed on 09.05.2011, p307.

⁵⁰ Bernstein P. A., Rahm E.: A survey of approaches to automatic schema matching, in the VLDB Journal 10: 334–350, 2001, p.33ff.

- *Auxiliary Information*: most matchers rely not only on input schemas S1 and S2 but also on auxiliary information, such as dictionaries, global schema, previous matching decisions, and user input.

The classification does not consider the different types of schemas (relational, XML, object-orientated, etc.) and their internal representation because match algorithms deal mostly with the output information of the schema not the representation.

Schema-Level Matchers

A schema-level matcher only handles information about the schema, not about the instance data. This information comprises facts about the schema elements such as name, description, data type, relationship type (part-of, is-a), constraints and schema structure. The general procedure is that the algorithm applied finds multiple match candidates and for each candidate the degree of similarity is estimated in order find the best match.

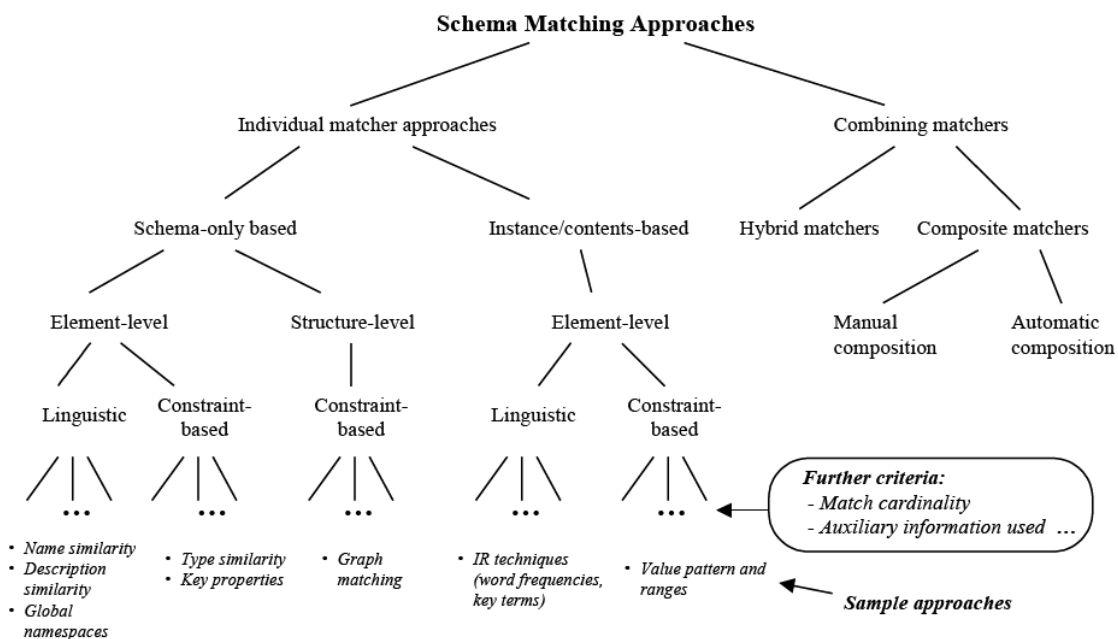


Figure 13: Classification of schema matching approaches⁵¹

For the granularity of the match, there are two main alternatives for schema-level matching; element-level and structure level matching. In the first alternative, each element of the first schema determines the matching element of the second schema. For the simplest case, this means that only elements of the finest granularity, also referred to as the atomic level, are

⁵¹ Bernstein P. A., Rahm E.: A survey of approaches to automatic schema matching, in the VLDB Journal 10: 334–350, 2001, p.338.

considered. For example in a XML schema the finest level is the attributes and in a relational schema it would be the columns. An example of atomic matches is demonstrated in Figure 14, `Address.ZIP` of schema 1 match `CustomerAddress.PostalCode` of schema 2. The element matching can also be applied to higher-level granularity but requires that elements of higher level are examined in isolation, ignoring its substructures and components. The latter alternative, structure level matching, combines elements that occur together in a structure, shown in Figure 14. The optimal case would be that all elements of the structure match but all kind of cases is possible depending on required preciseness and completeness of the match.

Linguistic approaches are using names and text (i.e. words and sentences) to find a match with semantically similar schema elements. Bernstein and Rahm distinguish between two types of language based schema level approaches: name matching and description matching. Name matching detects matches of schema elements with similar names. There are a number of different techniques on how identifying name based matches such as equality of names, synonyms, hyponyms and similarity of names based on common substrings, edit distance, pronunciation as well as user-provided name matches. Description matching refers to the use of comments (in natural language) about the semantic of the schema in order to find similarity between schema elements. Two schemas with different element name; S1: emp // employee name and S2: name // name of employee could be matched using linguistic analysis of the description of the elements.

S1 elements	S2 elements	
Address	CustomerAddress	full structural match of Address and CustomerAddress
Street	Street	
City	City	
State	USState	
ZIP	PostalCode	
AccountOwner	Customer	partial structural match of AccountOwner and Customer
Name	Cname	
Address	CAddress	
Birthdate	CPhone	
TaxExempt		

Figure 14: Full vs. partial structural match (example)⁵²

Constraint-based approaches consider schema constraints; data types, value range, uniqueness, relationship types and cardinalities, to name a few example. If the two input schemas are appointed with such constraints, match algorithms can determine a match, based

⁵² Bernstein P. A., Rahm E.: A survey of approaches to automatic schema matching, in the VLDB Journal 10: 334–350, 2001, p.337.

on the provided information. In general the use of constraints only, leads to poor results since there may be several elements with corresponding constraints but this approach can give additional help to restrict the number of match candidates and may be combined with other match approaches.

Match cardinality is concerned with the number of match result for schema elements to be matched. An element of one schema can participate zero, one or many times in mappings to one or more elements in a second schema. This expresses the usual relationship cardinality of 1:1, 1:n, n:1 and n:m for different mapping elements (global cardinality) or for an individual mapping element (local cardinality). In general most existing approaches apply matching of elements with highest similarity with the effect of merely local 1:1 matches and global 1:n matches.

The reuse of common schema components and previously determined mappings is a way to improve the effectiveness of schema matching. Common schema information, which must be agreed among participating parties such as an enterprise, its trading partners, relevant standard bodies or similar organizations, is defined and maintained in a schema library in order to reduce schema variability. Schema editors should consult these libraries in order to reuse predefined schema fragments and defined terms. The reuse of schema elements has been proven to be especially appropriate for frequently used entities such as address, customer, employee, purchase, order and invoice. The reuse of existing mappings is another generic approach including for example the reuse of previously element-level matches by adding them to a thesaurus or reusing entire schema structure fragments which is applicable when matching different but similar schemas to the same destination. The latter may occur when new sources should be added into a data warehouse.

Instance-Level Matchers

The result of evaluating instances is a precise description of the actual content of the schema elements and this information can be used as an enhancement to schema-level matching. For example information such as value ranges and character patterns can help a constraint-based matcher to determine the data types more accurately and thereby improve its effectiveness. The insight of the content and meaning of the schema elements is also especially useful when schema information is missing or is restricted or as in the worst case when there is no schema available. In such cases the information of the instance-level data can be used as an input when constructing a new schema or adding information to an existing schema.

Schema-level matching can also be used on its own. In a first step the instances of schema 1 is evaluated in order to specify the data of the elements. Thereafter, the instances of the second schema are matched one-by-one against the specified elements of the first schema. The result of each instance match must be merged and abstracted to the schema level in order to establish a ranked list of match candidates in the first schema for each (schema-level) element in the second schema. Another approach of instance matching is to use auxiliary information, for example previous mappings retrieved from matching different schemas. This type of instance matching can be especially rewarding when matching text elements. The match candidates are making use of key words and this will be illustrated by an example. An analysis done previously may have indicated that the term “Microsoft” occurs frequently for the schema elements “CompanyName”, “Manufacturer”, etc. If an S2 element X frequently contains the word “Microsoft” this can be used to construct “CompanyName” in S1 as a match candidate for X, even though “Microsoft” does not occur in the instance of S1 many times.

The two mentioned approaches of instance matching are in general used for identifying element-level matches. It is not appropriate to use if for sets of schema elements or structures since it would require characterizing the content of these sets. The main problem with instance matching is the explosion of number of combinations schema elements for which instances would have to be evaluated.

Combination of Different Matchers

Matchers using multiple matching approaches are more likely to achieve better results compared to matchers only using one single technique. There are two ways of combining matchers; hybrid and composite matchers.

The hybrid matcher identifies the match candidates based on multiple criteria from several matching approaches. The advantage of using hybrid matchers is the provision of better match candidates, better performance compared to separate execution multiple matchers and also the effectiveness can be improved because poor match candidates matching only one of many criteria can be filtered out in an early stage and complex matches which require joint consideration of multiple criteria can be solved. The hybrid matcher is in general uses a hard-wired combination of particular matching approaches that are executed simultaneously or in a fixed order.

A composite matcher combines the result of several independently executed matchers. As opposed to the hybrid matcher, the composite matcher is more flexible since it is possible to

use any modular matchers based, for example, on application domain or schema languages (e.g., different approaches can be used for structured vs. semi-structured schemas). It also allows a more flexible order of the matchers, implying that they could either be executed simultaneously or sequentially.

Schema Mapping

Mapping between different data representation is an essential part of information integration tasks. According to Fagin et al. schema mapping explains how the data from a source format can be transformed into a target format or in other words it can be referred to as a high-level specification in which the relationship between two database schemas is described⁵³.

In the literature^{54, 55, 56, 57} two common approaches to the schema mapping is found; local-as-view (LAV) and global-as-view (GAV). Based on the work of Lenzerini⁵⁸ the LAV and GAV will be briefly presented below.

Local-As-View (LAV)

In a data integration on system $I = \{G, S, M\}$, described earlier in this section, the LAV approach the mapping M associates to each element s of the source schema S a query q_G over G . This means that the query language $\mathcal{L}_{M,S}$ language only allows expression consisting of one symbol of the alphabet \mathcal{A}_S . Consequently the LAV mapping is a set of assertions, one for each element s of S in the following form:

$$s \sim > q_G$$

The intention of the LAV approach is that the content of each source should be associated in terms of a view over the global schema. The approach is of advantage when the data integration system is based on a stable and well established global schema. It is well known that query processing in LAV is strenuous, the only available knowledge about the data in the global schema exists in the views representing the sources and such views only have partial

⁵³ Fagin R., Kolaitis P., Nash A.: Towards a Theory of Schema-Mapping Optimization, 2008, online in WWW at <http://www.almaden.ibm.com/> (<http://goo.gl/fmcW8>), accessed on 09.05.2011, p.33.

⁵⁴ Calvanese D., Giacomo G., Vardi Y. M.: Simplifying Schema Mappings, 2011, online in WWW at <http://www.edbt.org/index.php> (<http://goo.gl/OFFTw>), accessed on 10.05.2011, p.1.

⁵⁵ Lenzerini M.: Data Integration: A Theoretical Perspective, 2002, online in WWW at <http://www.cs.ubc.ca> (<http://goo.gl/WJfRv>), accessed on 10.05.2011, p.3ff.

⁵⁶ Cali A.: Reasoning in Data Integration Systems: why LAV and GAV are Siblings, 2003, online in WWW at <http://www.csd.uoc.gr/> (<http://goo.gl/xQBcQ>), accessed on 10.05.2011, p.1.

⁵⁷ Leser, U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.187.

⁵⁸ Lenzerini M.: Data Integration: A Theoretical Perspective, 2002, online in WWW at <http://www.cs.ubc.ca> (<http://goo.gl/WJfRv>), accessed on 16.05.2011, p.3ff.

information about the data. The mapping associates to each source a view over the global schema and therefore it is not obvious how to use the sources in order to answer the queries expressed over the global schema.

Global-as-view (GAV)

As opposed to the LAV approach, for GAV the mapping \mathcal{M} associates to each element g in \mathcal{G} a query q_s over \mathcal{S} . This means that the query language $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ language only allows expression consisting of one symbol of the alphabet $\mathcal{A}_{\mathcal{S}}$. Consequently the LAV mapping is a set of assertions, one for each element g of \mathcal{G} in the following form:

$$g \sim > q_s$$

The idea of the GAV approach is that the content of each element g of the global schema should be characterized in terms of a view q_s over the sources. This approach works best for data integration system for which the sources are stable. Compared to LAV, query processing in a global-as-view approach looks easier because the mapping directly specifies which sources that corresponds to the elements of the global schema.

2.2.5.2 Multi Database Language

As opposed to the schema matching approach the multi database language approach does not require a global integrated schema. In general it takes a considerably time to develop a global schema and it requires a high level of maintenance and this advocate the use of a multi database language. A multi database language has the ability to query multiple databases and offers different functions to combine the data that supports the user performing data integration. Multi database languages were originally proposed⁵⁹ in the context of purely relational component databases and it was based on standard SQL capabilities but significantly extended in its functionality⁶⁰. One of the tasks of the multi database languages is the handling of manipulation of data representations⁶¹ including transforming the source information into a representation that is useful for the user. Additionally a multi database language should provide the possibility to display the information available from the various sources, but it is assumed that the user has an idea where the information is residing otherwise

⁵⁹ Fahl G.: Object Views of Relational Data in Multidatabase Systems, 1994, online in WWW at <http://www.uu.se> (<http://goo.gl/dLqZt>), accessed on 27.11.2010, p.9.

⁶⁰ Bright M. W., Pakzad S. H.: A Taxonomy and Current Issues in Multidatabase Systems, 1992, online in WWW at <http://www.mst.edu> (<http://goo.gl/psh36>), accessed on 27.11.2010, p.55.

⁶¹ Bright M. W., Pakzad S. H.: A Taxonomy and Current Issues in Multidatabase Systems, 1992, online in WWW at <http://www.mst.edu> (<http://goo.gl/psh36>), accessed on 27.11.2010, p.55.

the search of necessary data can be an overwhelming task⁶². A further desirable property is that functions that limits the scope of the query to the corresponding component databases. Though multi database languages were developed based on relational databases, more recently discussions of multi database languages for object-oriented models are available and there are possibilities to extend the language in order to also handle data model heterogeneity⁶³.

2.2.5.3 Ontology

Ontologies are a well-established concept for resolving semantic heterogeneity. It has its roots in artificially intelligence and addresses the subject of domain knowledge representation. Cruz and Xiao describe an ontology as “... a formal, explicit specification of a shared conceptualization”⁶⁴. The “conceptualization” denotes an abstract model of certain domain knowledge in the world, represented by the domain’s concept. Further the word “shared” in the definition, illustrates that the ontology is agreed in a particular community using the ontology. “Explicit” refers to the explicit definition of the concepts of the ontology and the constraints of these concepts. At last, “formal” indicates that the ontology must have the ability to be executed by a machine.

Based on the work of Leser and Naumann⁶⁵ the two main concepts, *conceptualization* and *formal specification* of the above definition are addressed in more detail. Additionally the aspect of communication of ontologies is highlighted.

One of the key characteristics of an ontology is the establishment of all relevant concepts of a particular domain. The interaction between concepts, the real world things and symbols is depicted in Figure 15.

In this context the term “Concept” refers to a particular “real-world” thing or a group of things. These concepts are only imaginably and individual for each person and in order to communicate a concept symbols, words or a string of characters are used. For each person these concepts are distinct but in general it is difficult to communicate this in an explicit way. Concepts as such can only be represented by symbols and the goal of an ontology is

⁶² Bright M. W., Pakzad S. H.: A Taxonomy and Current Issues in Multidatabase Systems, 1992, online in WWW at <http://www.mst.edu> (<http://goo.gl/psh36>), accessed on 27.11.2010, p.55.

⁶³ Fahl G.: Object Views of Relational Data in Multidatabase Systems, 1994, online in WWW at <http://www.uu.se> (<http://goo.gl/dLqZt>), accessed on 27.11.2010, p.9.

⁶⁴ Cruz I. F., Xiao H.: The Role of Ontologies in Data Integration, 2005, online in WWW at <http://www.uic.edu/uic> (<http://goo.gl/0yRrR>), accessed on 29.11.2010, p.2.

⁶⁵ Leser, U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.272ff.

consensual interpretation of the symbols of all members of a certain community and this illustrates the main difficulty of ontologies.

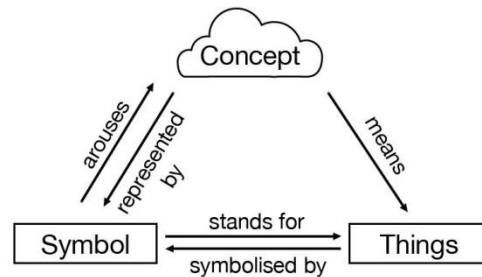


Figure 15: Interaction between concepts, the real world things and symbols⁶⁶

A detailed model capturing the properties of the concept, such as attributes, constraints and relations between the concepts is called a specification of the ontology. In order for a computer program to understand and execute an ontology a formal specification is necessary. Ontologies using a formal specification are referred to as a formal ontology in contrast to informal ontologies including for example unstructured lists of terms or thesaurus. The main difference between an ontology and data model is that the ontology does not make a strict separation between schema and data.

A further important property of an ontology is the exchange of data which aims to support the communication process and an explicit and common understanding of the concepts used for certain domains and its applications. This is a major advantage when different groups of people are participating in developing process or using the application.

The goal of using ontologies is not to define a global, explicit and robust definition of terms for all communities and to stipulate them to have the same interpretation; it is rather to offer communities to communicate based on their own defined ontology⁶⁷.

To provide an impression how ontologies can be used for data integration tasks, an overview on different ontology applications for data integration is presented below⁶⁸:

- **Metadata Representation**

⁶⁶ cf. Leser, U., Naumann F. : Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen, dpunkt.verlag, 2007, p.273.

⁶⁷ Hakimpour F.: Using Ontologies to Resolve Semantic Heterogeneity for Integrating Spatial Database Schemata, 2003, online in WWW at <http://www.uzh.ch> (<http://goo.gl/5gMyr>), accessed on 30.11.2010, p.2ff.

⁶⁸ Cruz I. F., Xiao H.: The Role of Ontologies in Data Integration, 2005, online in WWW at <http://www.uic.edu/uic> (<http://goo.gl/0yRrR>), accessed on 29.11.2010, p.6.

A local ontology can be used to explicitly represent metadata (i.e. source schemas) in each data source by using a single language.

- **Global Conceptualization**

A conceptual view over schematically-heterogeneous source schemas can be provided by a global ontology.

- **Support for High-Level Queries**

Given a high-level view of the sources, as provided by a global ontology, the user can formulate a query without specific knowledge of the different data sources. The query is then rewritten into queries over the sources, based on the semantic mappings between the global and local ontologies.

- **Declarative Mediation**

A global ontology can be used for query processing in a hybrid peer-to-peer system as a declarative mediator for query rewriting between peers.

- **Mapping Support**

A thesaurus that is formalized in terms of ontology can be used in order to support the mapping process to facilitate its automation.

2.2.6 Challenges'

Data integration has been studied actively for more than 20 years and it has become a prominent part of the field of database research. The key challenge is to resolve semantic heterogeneity and this chapter addresses the question of why semantic heterogeneity is regarded a core issue.

The main cause for semantic heterogeneity is because the data structure is developed independently and this may lead different ways of representing the same or overlapping concepts. Even though people developing data structures who are confronted with the same modelling goal, the outcome will look differently because humans think differently. To summarize the problematic, heterogeneity will always exists as long as it is possible to structure data differently and as long as there are different designers involved in the data base developing process⁶⁹.

⁶⁹ Halevy A. Y.,: Why Your Data Won't Mix Semantic Heterogeneity, 2005, online in WWW at <http://www.washington.edu> (<http://goo.gl/peCY2>), accessed on 02.12.2010, p.3.

Doan and Halevy⁷⁰ discuss the reasons why it can be such a complex endeavor to resolve semantic heterogeneity; information and documentation about the data is in general very difficult to get hold of from the creators, who may no longer be available or not possible to ask. If documentation is available it is not unusual that it is sketchy, incorrect or outdated. Element names, types, data values, schema structures, and integrity constraints in the schema or data are clues typically used for matching schema elements. Such clues are not always possible to trust, for example an element name can refer to different real-world entities and two elements with different names can refer to the same real-world concept. A further issue when using clues for matching is the problem with incomplete information, for instance an element stating “contact information” does not necessarily provide sufficient information about what kind of contact information, such as address or phone number, etc. that is included and therefore makes it difficult to find an matching element. Schema matching is also related with challenges when it comes to finding the best match that increases the complexity and the costs. Due to the problems for computer applications to understand the entire semantic meaning or intent of schema or data, it is customary that people are involved in the matching process, this of course is time-consuming and a very labor intensive task causing major cost but also imply problems that the input of single parties may be considered too subjective. The challenges concerning semantics are the reason for many still open research issues in the data integration field⁷¹.

⁷⁰ Doan A., Halevy, A. Y.: Semantic-Integration, Research in the Database Community, A Brief Survey, 2005, online in WWW at <http://www.wisc.edu> (<http://goo.gl/392LN>), accessed on 05.12.2010, p.2.

⁷¹ Dittrich K., Ziegler, R.: Three Decades of Data Integration - All Problems Solved?, 2004, online in WWW at <http://www.uzh.ch> (<http://goo.gl/PfFCK>), accessed on 10.12.2010, p.8.

3 Meta-Modelling within the Open Model Initiative

This chapter presents a conceptual approach to support data integration for business analytics based on a modelling framework. But firstly, the topic of modelling, including a presentation of the *Open Model Initiative*⁷², available user roles when working with models, methods for modelling and motivation and benefit of using a model are introduced. The last part of this chapter will demonstrate the ideas behind the “conceptualization”, the process of mapping a method to a meta-model platform.

3.1 The Open Model Initiative

The founders of the *Open Model Initiative* are persuaded that in intermediate-term the importance of communicating by the means of models will increase significantly and will become a considerably factor of production for knowledge intensive business especially⁷³. The idea is to provide models for everyone with the goal formulated as: “... *establishment of a community that deals with the creation, maintenance, modification, distribution, and analysis of models*”⁷⁴. The initiative does not only invite modelling experts to participate but also encourages people with little or no knowledge about models to join in order to enable them to work with models. In principal there should not be any restriction to the term model and the framework should serve as a platform for all kind of model content that is considered useful for any kind of user group.

3.1.1 Definition of the Term “Model”

To fulfill the goal of providing *models for everyone* the *Open Model Initiative* aims to keep an open approach to the term model without any restrictions to the definition. Hence, when looking up model in a common dictionary, the term is used to describe a social role, archetype and job in various domains like architecture, design, fashion, cars, statistics, and even zoology⁷⁵. This extremely broad definition does obviously not serve the purpose of the *Open Model Initiative* and must be narrowed down. The definition for the model used in the context of such modelling platform is as formulated by the authors of the *Open Model Initiative* as “*Models are a representation of either reality or vision that is created for some certain*

⁷² Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010.

⁷³ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.8.

⁷⁴ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.8.

⁷⁵ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.15.

purpose with an intended goal in mind.”⁷⁶ The vision or the reality represented by a model is sometimes also referred to as a “system under study” (SUS) and such SUS can be distinguished in three different types; a physical observable element or phenomena, a digital system available in computer memories or an abstract system that eventually only exists in human brains⁷⁷. Depending on the type of SUS, different kinds of models are distinguished; a model that refers to an already existing SUS is described to as descriptive model and a SUS does not yet exists but to be created is referred to as a specification model.

3.2 Motivation and Benefit of Modelling

This chapter introduces the benefits of modelling in general and provides a presentation of the goal and motivation for using the *Open Model Initiative* as well as an idea how to realize the project outlined in this thesis.

The founders of the *Open Model Initiative* point out that there is barely any scientific work available dealing with reasons of why one should model at all. Their approach to this topic is that models are *knowledge that can be operationalized*⁷⁸ and they have defined four types of benefits that a model can provide:⁷⁹

- **Specification**

Perspective models are models that describe something that does not yet exist and these models can be used as a specification. The benefit of using a model as a specification is reduced complexity and due to the fact that common understanding is created this enables a structured approach to the actual implementation of the model.

- **Implementation**

For semi-automatic implementation of target states a model can be a useable input. Model Driven Architecture (MDA) and Model Driven Engineering (MDE) are two examples of how implementation can be supported by a model.

- **Documentation**

Since models are explicit specification they serve very well as documentation and it

⁷⁶ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.15.

⁷⁷ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.15.

⁷⁸ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.16

⁷⁹ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.16ff.

aids easier and more precise understanding. The documentation, with the ability to preserve knowledge, can also be utilized to support knowledge management tasks.

- **Evaluation**

As already pointed out, models are clear specifications of target states and thus a created object can always be evaluated against its model. This enables a verification of completeness of functionality and other non-functional quality attributes.

The *Open Model Initiative* aims to support people that work with any model according to the definition in chapter 3.1.1, and it considers to have the duty to act as an enabler in this context and consequently to fulfill the obligation of providing active support to potential users. Based on the perception of that it is better to begin small with good support quality, the founders of the framework of open models have deduced a hypothesis stating that in order to establish an *Open Model Initiative* the interpretation of the term “model” has to be reduced in an early stage and can be broadened again gradually after the successful implementation of some projects⁸⁰. Hence, to begin with the platform will only include a few selected types of models. To facilitate the selection of the models, the Open Model Initiative classifies the models according to their representation. Two types of models are distinguished, linguistic and non-linguistic⁸¹. The latter is also referred to as iconic models and exists of signs and symbols that are showing an obvious similarity to the object it is representing. Linguistic models consist of basic primitives such as signs, characters or numbers that do not have an obvious relationship to the object it is representing. Based on the language type used, the linguistic model can be divided further into two subclasses; textual languages and graphical or diagrammatic languages. The first refers to the use of the alphabet, for example the Latin, Cyrillic or Arabic alphabet. Graphical or diagrammatic languages are made up of pictorial signs that represent the objects they visually depict. It should be mentioned that this reference is not based on the obvious similarity as it is for the iconic model. Example of diagrammatic models is the Entity Relationship Modelling (ERM) technique and most parts of the Unified Modelling Language (UML). Figure 16 shows the classification of the different types of models and its subclasses.

To put the vision of the *Open Model Initiative* into practice the founders have formulated three main topics that need to be considered; Open Model Community, Open Model

⁸⁰ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.20.

⁸¹ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.22ff.

Foundations and Open Model Projects. For the community organizational structures, processes and motivation are aspects that need to be set up and managed. The modelling foundations such as modelling language, modelling procedures, mechanism and algorithms as well as IT-based modelling environments for the creation, maintenance and processing of models must also be provided. Finally, a framework for projects that allows the creation of (reference) models and the development of modelling foundations is required.

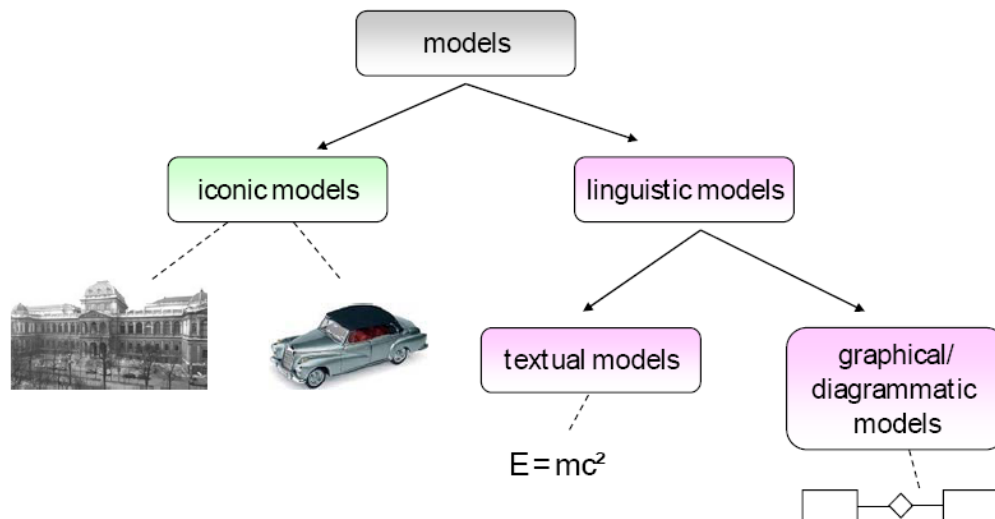


Figure 16: Representation of different model types⁸²

The platform for the Open Model Community provides a central point of contact for its members from different domains, both from the academic and the industrial sector and allows them to interact and exchange and combine their know-how regarding models. The initiative provides the potential users the possibility to create a large repository of high quality models that can be used for sharing knowledge base on a free basis. The last benefit to be mentioned is that by providing the foundations of modelling in terms of modelling methods and IT-based modelling environment the *Open Model Initiative* acts as an important “enabler” that support its users to select the correct basics for the creation of their models.

3.3 Modelling Methods

Based on Karagiannis et al.⁸³ the ideas of modelling methods, the concept of meta-model and the meta² view in the model hierarchy are presented in this section. Modelling methods are

⁸² Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.22.

⁸³ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p64ff.

the theoretical basis for modelling and a precondition for creating models. The modelling methods together with the modelling environment form the technical foundation for hosting a platform for the *Open Model Initiative*.

The modelling methods consist of modelling techniques and mechanisms and algorithms where as the modelling techniques can be further divided into modelling language and modelling procedures. Before starting to elaborate the details of modelling methods the Figure 17 provides an impression of how the different modelling methods are associated and their relations.

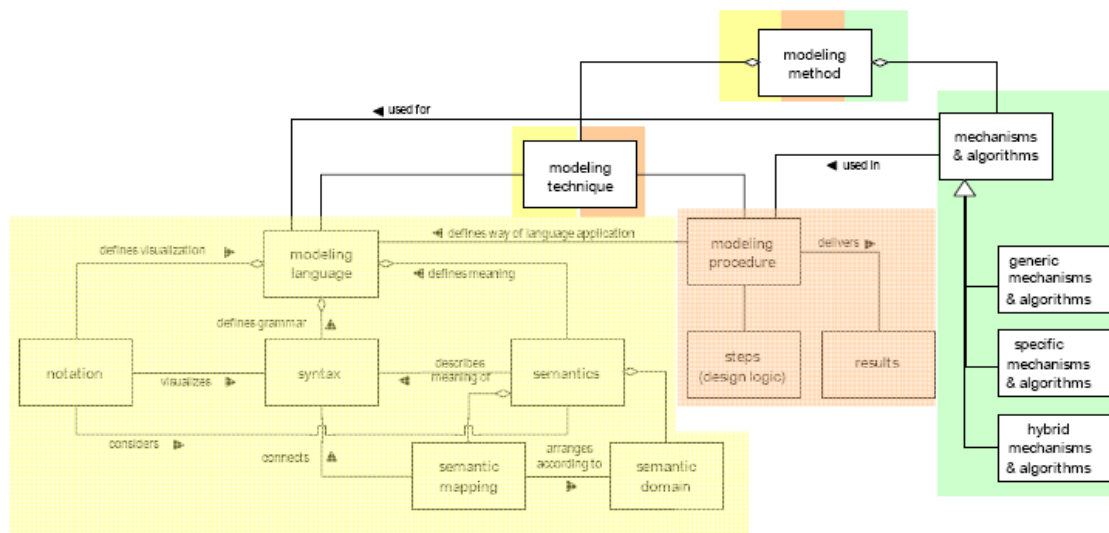


Figure 17: Modelling methods and mechanisms & algorithms⁸⁴

3.3.1 Modelling Language

A modelling language can be described according to three perspectives: syntax, semantic and notation. The syntax defines the available modelling elements and rules determining the valid combinations of these elements. The meaning of the modelling language elements is reflected by the semantic. The last aspect of a modelling language is the notation that denotes the visual appearance of the syntactical elements. To make these elements more tangible for the reader, an example based on the Entity Relationship modelling language is provided in Figure 18. The syntax, for a simplified version of this language, describes that the modelling language is made up of rectangles, rhombuses and lines and it is only allowed to connect two rectangles directly with each other. The semantic of these syntactical elements is describes as follows;

⁸⁴ cf. Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.65.

the rectangle represents a real world entity, a rhombus indicates the relationship and the last element line shows which entities that are involved in the relationship. By using the names of the graphical shapes for the syntax elements the notation is already implicitly illustrated. Though it is important to point out that in principal syntax and notations is separated. Figure 18 depicts the three different aspects, syntax, semantics and notation of a modelling language.




<i>syntax</i>	<i>semantics</i>	<i>notation</i>
syntax element 1	entities of the real world	
syntax element 2	relationships	
syntax element 3	connectors between entities and relationships	

Figure 18: Syntax, semantics and notation of the Entity Relationship Modelling Language⁸⁵

3.3.2 Modelling Procedures

Modelling procedures addresses the topic of how to combine the basic elements of a modelling language and the steps that have to be carried out when creating a model and the corresponding results. The modelling procedure should completely depict all necessary aspects (completeness) and at the same time it should guarantee syntactical correctness. A further requirement of the modelling procedure is that it must be adequate to achieve the modelling objectives. Such objectives can include modelling of complex problems at different levels of abstraction.

The modelling objectives have an impact of the modelling techniques, for instance the amount of syntactical rules and steps in the design logic of a modelling technique are determined by the degree of complexity in terms of desired automatic processing. A model that is representing a “system under study” with the purpose of supporting people discussing this object may allow some syntactical mistakes but a model for which information should be processed by a computer on the other hand, does not tolerate any errors.

⁸⁵ cf. Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.66.

3.3.3 Mechanism and Algorithms

Mechanisms and algorithms support models that have an objective to automate the processing. This part of the modelling methods provides the possibility to execute structural analysis (for example calculations of predefined criteria like costs, delivery times) and simulation of models (for example prediction of cycle times or staff requirements). Three types of mechanisms and algorithms are distinguished: generic, specific and hybrid.

In this context, the generic type denotes that the semantics of the processed model instances is barely considered. This kind of mechanisms and algorithms can be applied to models that have defined syntactical rules created in order to meet the defined modelling steps.

In contrast to generic mechanisms & algorithms, the specific ones must not only comply with defined syntactical rules and the specified modelling steps, but in particular respect the semantics. In other words, for a correct processing of the model, the meaning of the real world must be considered. The last type, hybrid mechanisms & algorithms, is an approach in which the syntactical processing techniques and semantic issues are combined.

3.3.4 Meta² Representation: Formal Representation of the Meta-Model

As mentioned in the previous section a meta-model determines the set of available modelling language objects that can be used for creating a model instance. This section will look into the formal representation of a meta-model.

Based on an empirical analysis of different aspects of business model, the authors of the feasibility study of the Open Model Initiative summarize the requirements of a formal representation⁸⁶. The four main objectives (design, implementation, documentation and evaluation), also listed in detail in chapter 3.3, must be supported by the formal representation. In order to fulfill these requirements the study presents the idea of the meta²-model, which is an enlargement by one level of the standard meta-model hierarchy of the OMG⁸⁷. The basic idea behind this extra level is the definition of a conceptual framework for the development of the meta-model; this can also be referred to as the meta-model of the meta-model. Such a model enables the definition of a meta-model and consists of three different modelling logics⁸⁸;

⁸⁶ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.81.

⁸⁷ Object Management Group, online in WWW at <http://www.omg.org>, accessed on 05.11.2010.

⁸⁸ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.82.

- **Information Logic:**

This logic defines the syntactical part of the meta model and it describes the basic information constructs available in the domain as well as the relationships between the information constructs.

- **Domain Logic:**

The Domain Logic handles the semantic of the modelling language. The most prominent aspect of this logic is the definition of the constructs of interest in the domain as well as the relationship between these constructs

- **Processing Logic:**

This logic describes the procedure of the model and the mechanisms and algorithms. All different kind of activities for the information construct which is necessary for obtaining knowledge and information about the domain is are found here within.

The three logics are not destined to any specific modelling tool and depending of the purpose of the meta-model it is possible with three different modelling languages used for the three different logics. Usually the information logic is represented by traditional UML constructs, however in some modelling domains the information logic is required to be much more specific than general structures like inheritance, inclusions relations and different types of aggregations. Grossmann gives further examples of how the logics can be represented⁸⁹; for the domain logic basic terminologies or ontologies can be used, but also ER-diagrams can be useful. The processing logic is often based on algorithms derived from the area of computer science and mathematics

Figure 19 illustrates the formalization of the modelling process where the lowest level is the original upon which a model is built. The highest level in the four-layer architecture represents the concepts for building meta-models.

⁸⁹ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009, p.124.

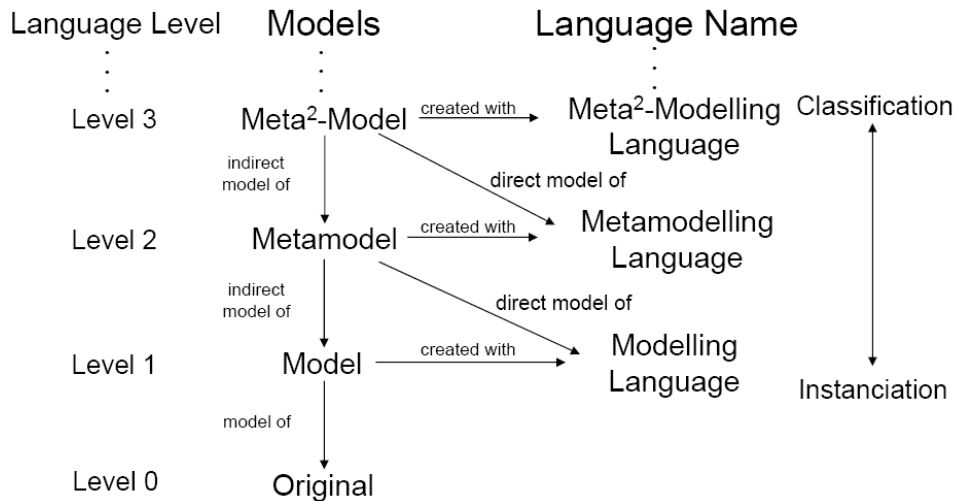


Figure 19: Formalization of the modelling process⁹⁰

3.3.5 Meta-Modelling Platform Architecture

In order to facilitate the topic of meta-modelling mentioned in chapter 3.3 a component-based, distributable and scalable architecture is proposed by Kühn⁹¹.

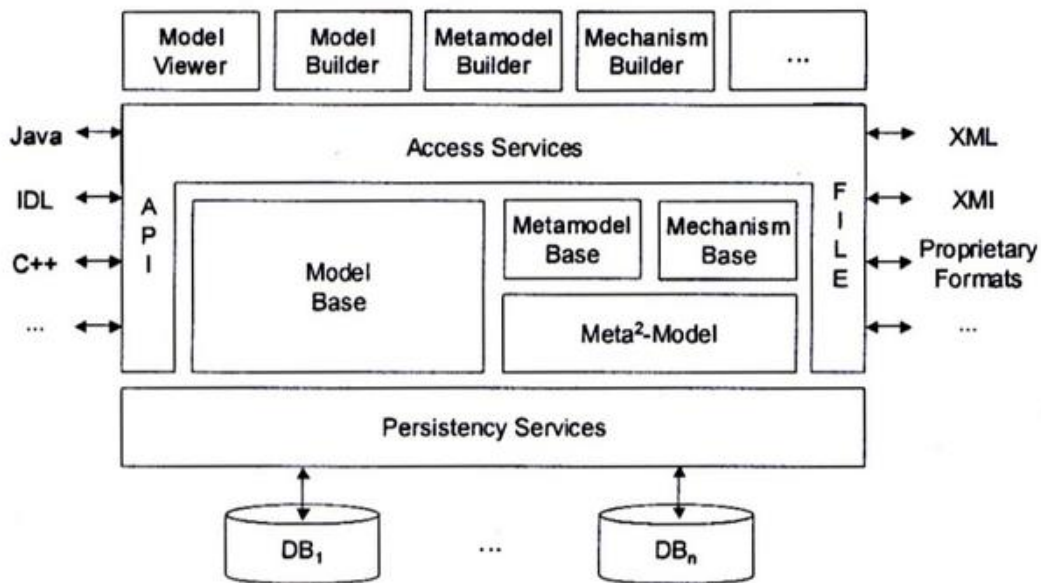


Figure 20: Generic architecture of meta-modelling platforms⁹²

⁹⁰ Karagiannis, D., Kuehn H.: Metamodelling Platforms. IN: Bauknecht, K. Min Tjoa A, Quirmayer G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin Heidelberg, p.3.

⁹¹ Karagiannis, D., Kuehn H.: Metamodelling Platforms. IN: Bauknecht, K. Min Tjoa A, Quirmayer G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin Heidelberg, p.5-6.

As shown in Figure 20 the generic architecture consists of the following elements:

Persistency Services

This component manages the storage of all models and the meta-model information. It holds information about concrete storage types such as specific databases, files systems etc. Additionally the persistency service is responsible for the distribution of parts of models and meta-models.

Meta²-Model

Within this part the creation of the meta-models and mechanisms, e.g. “classes”, “relations”, “attributes”, “model types”, and “scripts” takes place. It is a central part of the meta-model platform architecture is because it provides the conceptual foundation of the platform and it is connected with all other parts.

Meta-Model Base

The meta-model base controls all information about the meta-models which are managed by the modelling platform. Changes done on the meta-model base is forwarded to the model bases accordingly to keep consistency between the models and their meta-models.

Mechanism Base

Information about functionalities applied to the model and meta-models are kept in the meta-model base. There are two possibilities of storing such functionalities, either in the directly in the mechanism base or outside the meta-modelling platform. In the latter case, the mechanisms base only store information where to find the appropriate mechanisms.

Model Base

All models that are based on the meta-models are stored in the model base. This component communicates with the meta-model base in order to track meta-model changes and to forward them to the affected models.

Access Service

⁹² Karagiannis, D., Kuehn H.: Metamodelling Platforms. IN: Bauknecht, K. Min Tjoa A, Quirchmayer G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin Heidelberg, p.5.

File based and online interfaces to the different types of bases are provided by the access service. Information from the bases can be queried or changes depending on the given access rights.

Viewer and Builder Components

On top of the mentioned components, this element supports the usage and maintenance of the meta-modelling platform, e.g. model builder, meta-model builder, and mechanism builder.

3.4 Roles in Meta-Modelling

For using and administrating a meta-model platform different roles are distinguished according to Karagiannis and Kühn⁹³.

- **Method Engineer**

The method engineer ensures a consistent and an accurate defined modelling method. The characteristic of this role is her/his technical skills and in addition this profile often has professional skills in an application domain. The application domains can be divided into verticals skills such as financial services, telecommunications, public administration, and manufacturing and horizontal skills such as business process modelling, application development, workflow management, and knowledge management.

- **Language Engineer**

The language engineer defines the modelling language that includes an adequate definition of the syntax, semantics and the notation.

- **Process Engineer**

The definition of the modelling procedure lies in the responsibility of the process engineer. In these role experts in applying modelling languages and persons with considerable experience in project management and project execution is often found.

- **Tool Engineer**

The tool engineer is responsible for the configuration of the mechanisms of the meta-modelling platform for specific meta-models. Possible new additional mechanism that need to the meta-model is also within this role's responsibility.

⁹³ Karagiannis, D., Kuehn H.: Metamodeling Platforms. IN: Bauknecht, K. Min Tjoa A, Quirchmayer G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin Heidelberg, p.4-5.

- **Infrastructure Engineer**

The infrastructure engineer ensures that the IT infrastructure which is necessary to run a meta-modelling platform is provided and is responsible for integrating the platform to existing infrastructure.

Method User

A user applying the method by using the platform is referred to a method user and is the one creating models by employing the modelling language, following the modelling procedure and using the available mechanisms.

4 A Conceptual Method for Data Integration

This chapter gives a description of the theoretical method upon which the prototype is developed. Based on the work of Grossmann⁹⁴ this chapter begins with an introduction of the current situation of the problem of data integration and then continues with a presentation of the suggested approaches how to manage these challenges. As part of the meta²-level modelling upon the conceptual model is based, the topic of information logics applied for business analytics is demonstrated. The last part describes Grossmann's actual conceptual model.

Nowadays many commercial and open source tools supporting the data integration process are available; either offering models for data integration, for example Pentaho-Kettle⁹⁵, or tools which focus on ETL activities. CRISP-DM⁹⁶, as foundation for the data mining software PASW (Predictive Analysis SoftWare), formerly Clementine and SEMMA⁹⁷, foundation of the SAS Enterpriser miner software, are theoretical frameworks also including data integration. Grossmann points out that the main benefit of such instruments is the visualization of the integration activities that supports the user to understand data processing, the reuse of analyses sequences and it enables, to some extent, the automatic translation of the visual description to executable code⁹⁸. Such data integration tools and theoretical frameworks offer many advantages but there are still open issues. The major problem according to Grossmann is that meta-data used for data integration in general is only employed for representation of syntactic knowledge. Semantic information about the data structure is typically only available in an embryonic form though it plays a significant role in defining the analysis model and in the evaluation of the results. Hence it is customary that data integration work is often based on informal knowledge and the experience of the business analyst rather than a scientifically method. Only analyst experts know about possible pitfalls and side effects of procedures that restrain less experienced users.

⁹⁴ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009.

⁹⁵ Pentaho-Kettle, online in WWW at <http://kettle.pentaho.com>, accessed on 04.12.2010.

⁹⁶ PASW, Chapman P., Clinton J., Kerber R., Khabaza T., Reinartz T., Shearer C., Wirth R.: CRISP-DM 1.0 - Step-by-step data mining guide, 2000, online in WWW at <http://www.crisp-dm.org> (<http://goo.gl/kGvnA>), accessed on 04.12.2010.

⁹⁷ SAS, online in WWW at <http://www.sas.com> (<http://goo.gl/rINUI>), accessed on 04.12.2010.

⁹⁸ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009, p.122.

With the aim to make data integration for business analytics easier for professionals and also more applicable for different user extensions Grossman presents two approaches⁹⁹;

- Include detailed knowledge about the semantics, not only schema information. This information about the data should support the user with correct interpretation and evaluation of the results.
- Enhance the data transformation process by developing a more structured description of the transformation. This description should not only serve as a specification of the application of the basic algorithmic blocks but also include information about dependencies and side effects of the algorithms. For example it is usual that transformations trigger other transformations and need a check about the feasibility.

With respect to the aspects of the two above approaches, Grossmann proposes a conceptual model for data integration based on a combination of statistical meta-data and workflow management. To capture semantic knowledge about data, statistical meta-data, normally used in the field of statistic research, is suggested for this model since business analytics can be considered as certain kind of statistical analysis. The transformation process is suggested to be based on a workflow management framework in which business analytic activities can be proposed as actions and states obtained by the actions.

The conceptual model is based on the concept of meta²-level modelling hierarchy (described in detail in chapter 3.3.4), and the employment of the information logics for the application class business analytics will be described in the below sections.

4.1 Domain Logic for Business Analytics

The domain logic for business analytics includes an exact specification of the objects of interests and the attributes of these objects. Typical objects in commercial or industrial applications are “Customer”, “Enterprises”, “Employees”, “Sales products” to name a few. Common attributes that are describing these objects are “Name”, “Location”, “Size”, “Number of employees”, “Revenues” and “Prices”, etc.. There are different possibilities to capture the knowledge of the domain; a terminology model that allows the definition of relations between the terms is one example. By using broader terms, narrower terms and related terms a hierarchy is established and such conception of generalization and restriction can also be the basis for more sophisticated classification systems as well as ontologies.

⁹⁹ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009, p.123.

Besides terminology orientated models other ways of expressing a domain model can be a classical ER-model.

4.2 Information Logic for Business Analytics

The goal of the information logic is to specify the formal structure of the model. In order to apply a model correctly the domain logic must be mapped onto the information logic. Knowledge about the mapping process is essential and consequently a transparent representation of the features of the domain logic in the information logic is a significant aspect to be considered. Grossmann points out that the information logic designed for statistical processing is a natural choice due to the fact that business analytics is primarily based on empirical information and on methods developed in the statistics. A general feature of such information logic is that for analysis itself the structure is very simple, it can be represented in a flat file or a summary table. The challenge lies in the mapping of the domain logic to the information logic for statistics. Regardless of how elaborated or complex the data model is defined in the domain logic, it needs to be represented in such rather simple data model. Based on ideas of statistical meta-data models Grossmann proposes the information logic for business analytics including six concepts.

- **Observable Unit**

The observable units describe the elements of which information is sought. Typical examples for business intelligence in commercial areas are “Enterprises”, “Customers“ or “Products”.

- **Population**

The population is a collection of the observable units. In general it is not possible to observe a population completely, e.g. one may only have information about the behavior of registered customers during a certain time frame, but is interested to model the behavior of all potential customers in the future. The difference between the entire population and the observed parts of the population is essential when predicting behavior of the all members of the population as well as knowledge about the relation between population and observed units from the population.

- **Variable**

The variables, usually also referred to as attributes, hold the empirical information about the population. The properties of the observed units are formalized in the variables.

- **Value Domain**

The value domain defines the allowed measurement units and possible relationships between possible measurements units for each variable. Conversion formulas or relations defined by a hierarchical classification schema are typical examples of relationship between measurement units.

- **Dataset**

The dataset includes all data collected for the observable units and is the basic information entity for all different types of activities in business analytics.

- **Additional Attribute**

This concept comprises additional information that might be necessary, for example the proportion of the dataset of the population.

The above mentioned six entities are the basic information entities for business intelligence and can all together be referred to as information categories for its application. Figure 21 demonstrates the relationship between these information entities.

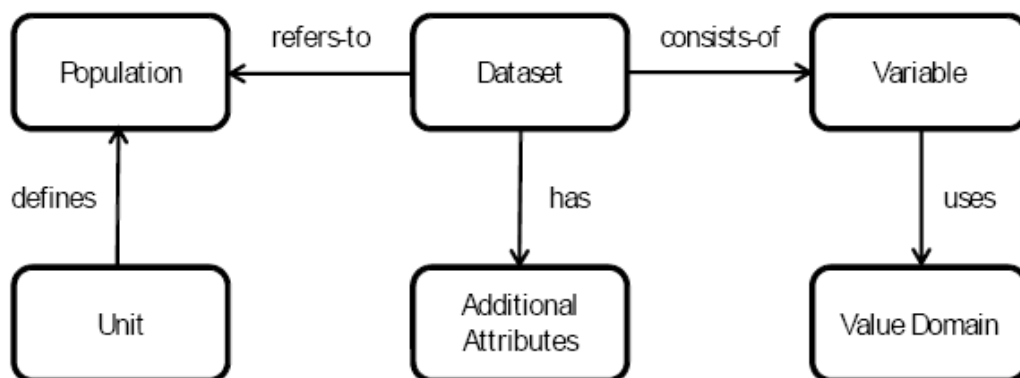


Figure 21: Information categories in Business Analytics¹⁰⁰

4.3 Processing Logic for Business Analytics

The processing logic is based on the the CRISP¹⁰¹ framework for data mining which includes five processing steps; “Business Understanding”, “Data Understanding”, “Data Preparation”, “Modelling” and “Model Evaluation”, though, as pointed out by Grossmann, it involves a

¹⁰⁰ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009, p.128.

¹⁰¹ PASW, Chapman P., Clinton J., Kerber R., Khabaza T., Reinartz T., Shearer C., Wirth R.: CRISP-DM 1.0 - Step-by-step data mining guide, 2000, online in WWW at <http://www.crisp-dm.org> (<http://goo.gl/kGvnA>), accessed on 04.12.2010, p.14.

more formal approach where the processing activities are referred to as basic operations. A basic operation illustrates a transformation where a number of instances of categories is used as input and is mapped into a number of new categories instances as output. Formally the transformation is represented as depicted below.

$$(C_1, C_2, \dots, C_p) \rightarrow (T_1(C_1, C_2, \dots, C_p), \dots; T_k(C_1, C_2, \dots, C_p))$$

Typically one main category is subject to the transformation, but a transformation can have side effects also affecting other categories. To give the reader an impression on how such side effects arises and appears two examples can be mentioned. “Data Selection” (a sub-task to the data preparation processing step) implies the creation of a new dataset based on an existing dataset. This can be done either by applying a horizontal or vertical selection whereas the horizontal selection means that a number of variables are dropped and the vertical selection reflects the selection of a specific number of observable units. Notably side effects are the restriction of the variables when applying the horizontal selection and when the vertical selection is used the description of the population has to be adjusted. The second example shows how side effects can arise when data is constructed. The simplest method to create data is to employ mathematical formulas and such methods are commonly offered by commercial data manipulation tools and programming environments in the field of statistic and data mining. The necessary modification of the value domain category when applying mathematical formulas to create data is the essential side effect.

A process model that is based on the idea of basic operations according to the concepts of workflow analysis has to preconceive some specific features related to business analytics;

- The role of computation and algorithms plays a much more important role compared to usual workflow management
- More data is included
- More activities are in general involved

Also the control of the entire process, which is determined by a rather complex evaluation functions based on various quality criteria's is an important feature.

4.4 Data Integration Process

When defining a process model for business analytics not only the processes for one category has to be defined also the side effects on other categories must be considered. Also the evaluation of the result of each basic operation has to be taken into account in order to define further processing activities. The data integration process is described as a process that is

combining different datasets and that regards the side effects of the category objects related to the datasets as well as evaluation functions. In order to manage the process basic operations must be combined and there are four different processing activities for the data integration process;

- Determine the observation units,
- Align the procedures for the pre-processing of the data sets to be integrated
- Performing the data integration operations
- Align the procedures for the post-processing of the integrated data set.

The main activities in the data integration process are found in the third step and all other steps are dependent on the activities in this step.

5 Prototype Implementation on the ADOxx[®] Platform

This chapter starts by presenting the term conceptualization in the area of meta-modelling. It shows the principal process of turning a theoretical method into a modelling method on a meta-modelling platform. The next section provides a brief introduction to the platform used for the implementation because knowledge of the concepts of the modelling platform is essential for the conceptualization of the meta-model. The major ideas for developing the prototype as a result of the intellectual process of matching the data integration method to the ADOxx[®] platform are described in 5.3 Conceptualization . A detailed presentation of the elaboration of the translation of the method on the platform, how the method was implemented with the ADOxx[®] platform, concludes the chapter.

5.1 „Conceptualization“-Approach

This section will look into the general process of turning an available method into a modelling method on a meta-model platform.

A prerequisite for the method engineer to develop a meta-model is the knowledge of the available modelling methods (cf. 3.3 Modelling Methods) and the available concepts used and provided by the meta-model platform. Based on this knowledge the task of the model engineer is now to convey the method into a conceptual model which in turn can be applied to the framework of a modelling method and which fits to the specific concepts of the meta-modelling platform. At this point it is important for the model engineer to derive elements like syntax and semantic which are adequate for the method and to elaborate suitable notations. Furthermore the potential mechanisms and algorithms for the modelling procedures must be considered. The result of this phase of the conceptualization work is a semi-formalised structure including the specification for the modelling method¹⁰².

5.2 The ADOxx[®] Meta-Modelling Platform

The software used for the implementation of the method is the ADOxx[®] meta-modelling platform as a meta-modelling-based development and configuration environment which supports the creation of domain specific modelling tools. The platform is realized on a component-based architecture and offers different levels and responsibilities to different kind of users.

¹⁰² Bergmayr, A., Karagiannis, D., Schwab, M.: i* on ADOxx[®]: A Case Study, in Fourth International i* Workshop (iStar'10) at CAiSE'10, 22nd International Conference on Advanced Information Systems Engineering (07-08 June 2010), pp. 92-97, p.2.

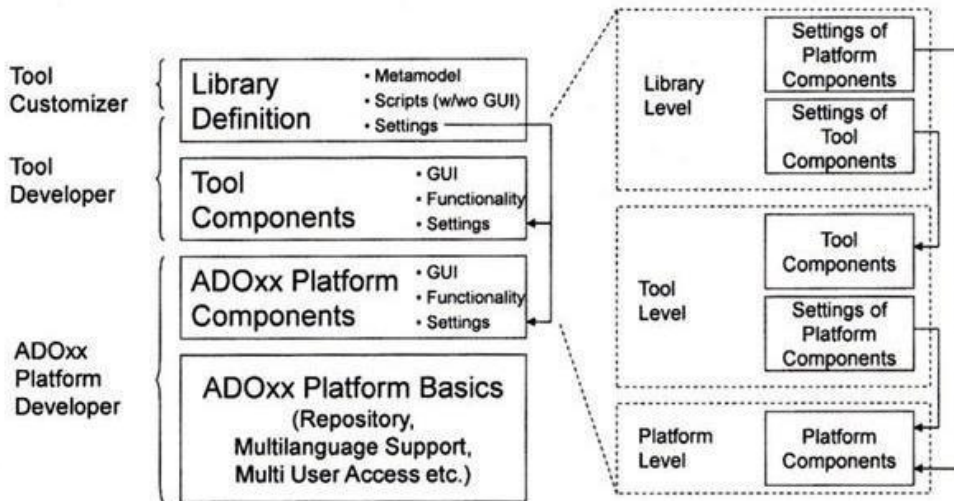


Figure 22: Architecture Levels and Responsibilities¹⁰³

The ADOxx[®] platform consists of three different workspaces¹⁰⁴; Product Workspace, Administration Workspace and Modelling Workspace. The product development environment offers features like creation of new modelling products, configuration of pre-defined components and definition of new functionality. The second workspace is the environment for configuration and administration. It gives the users the possibility to create their own modelling methods (as meta-models and DSLs) and to make customer specific extension via scripting. In this workspace the administration, such as user rights and model rights, is also managed. Furthermore the import and export of modelling methods, repositories, models, objects and users is handled. The third workspace is the modelling environment as the workspace where actual models are built, in either rich or web client deployment.

The administration workspace has been used to develop the modelling method of the prototype. By configuring and customizing the features of the ADOxx[®] platform such as the class hierarchy, class attributes including graphical representation and the processing logic the prototype was developed. In the modelling workspace the result of the implementation of the meta-models are evident and available to the users.

¹⁰³ Kühn, H.: The ADOxx[®] Metamodelling Platform, Workshop “Methods as Plug-Ins for Meta-Modelling” in conjunction with “Modellierung 2010”, Klagenfurt (24-26 March 2010), p2.

¹⁰⁴ Kühn, H.: The ADOxx[®] Metamodelling Platform, Workshop “Methods as Plug-Ins for Meta-Modelling” in conjunction with “Modellierung 2010”, Klagenfurt (24-26 March 2010), p.1.

5.3 Conceptualization Results

This section describes the major ideas as a result of the conceptualization of the meta-model prototype based on the conceptual model for data integration method (cf. 4 A Conceptual Method for Data Integration).

The conceptualisation results are presented in accordance with Kühn's classification of modelling methods¹⁰⁵:

- Modelling Procedure as the starting point of the analysis of the modelling method, a general methodological guideline has been derived
- Modelling Language identified the modelling and relation classes needed to support the procedure, distinguishing in a data meta-model for domain and information logic modelling and a process meta-model to describe the processing logic.
- Mechanisms and Algorithms supporting the enactment of the models created using the language and providing results as defined in the modelling procedure.

5.4 Modelling Procedure: Data Integration Methodology Guideline

5.4.1 Generic Definition and Dependencies

The modelling procedure is responsible for the design logic, how to combine the objects of the model in order to create a valid model. It defines the steps that must be executed when creating a model and the corresponding result.

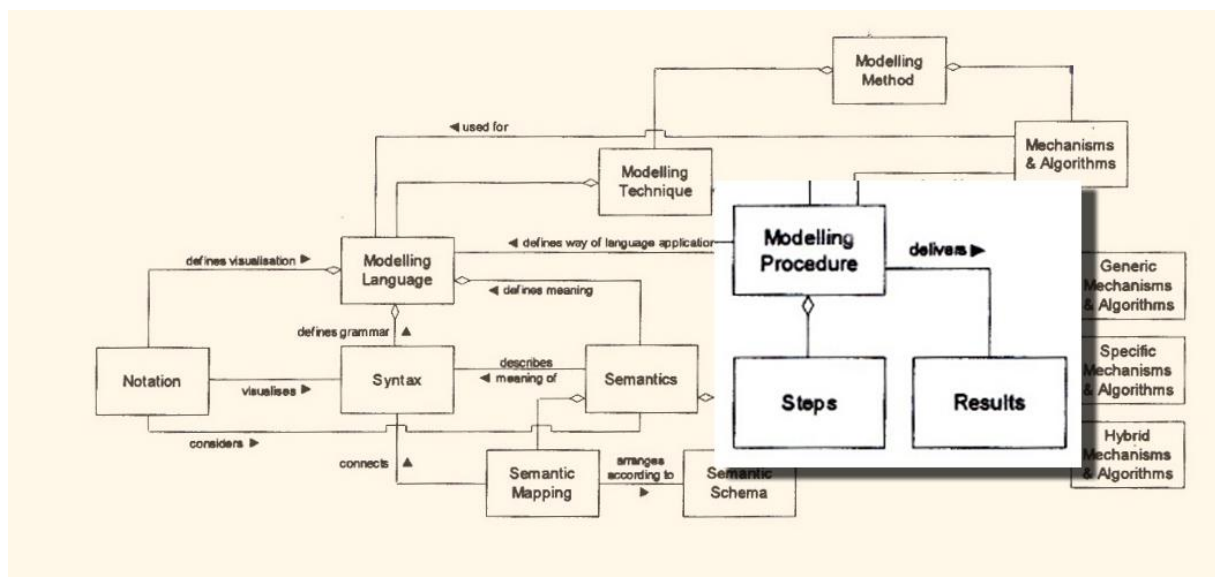


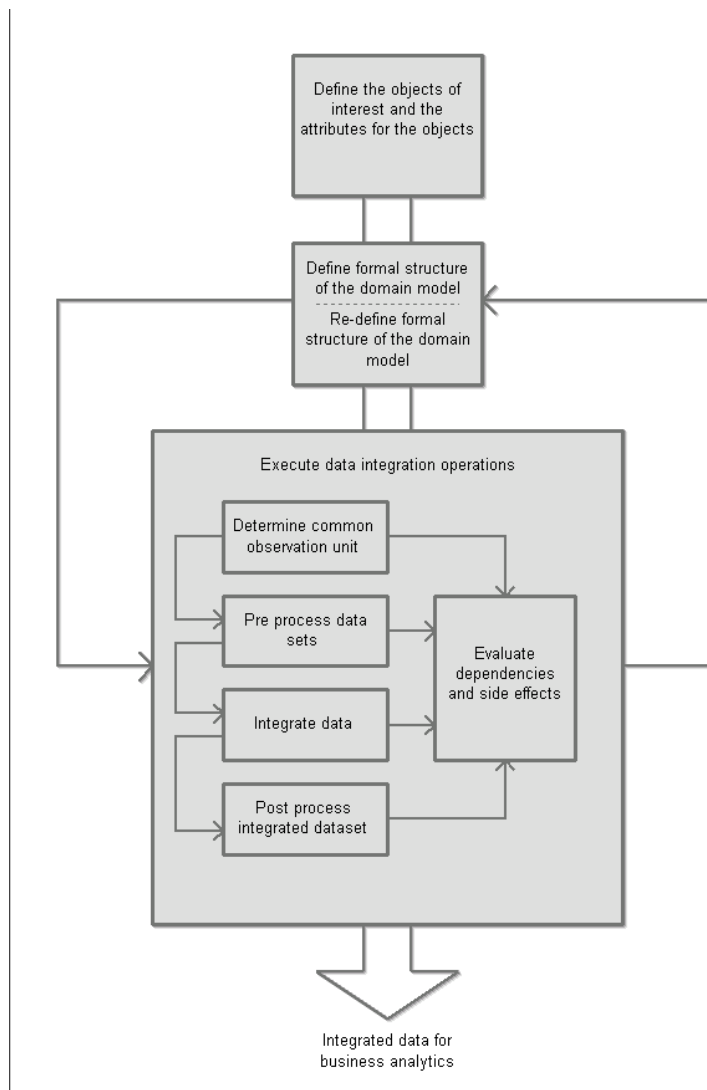
Figure 23: Modelling procedure¹⁰⁶

¹⁰⁵ Kühn H.: Methodenintegration im Business Engineering, PHD Thesis, University of Vienna 2004

The modelling procedure should enable that when the model is created that all necessary aspects are depicted and that it is syntactical correct¹⁰⁷. The following sections give the reader a presentation of the modelling procedure for the prototype.

5.4.2 Data Integration Methodology Guideline

A framework describing the major steps and tasks is presented in the Data Integration Methodology Guideline. It aims to provide the modeller with an overview of the major steps how to approach data integration problems, to explain the interaction between the data model and the process model and to present the iterative process in which the dependencies between the data and the process and the side effects of the process on the data are described.



¹⁰⁶ Schwab M., Utz W.: ADOxx® Customizing-Schulung, Methodenumsetzung, Erfahrungsberichte, Universität Wien, Department Knowledge Engineering, 2009, p5.

¹⁰⁷ Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010, p.66.

Figure 24: Methodological Guideline as Modelling Procedure

The process starts with the definition of the objects of interests and its attributes that will be included in intended data integration. The output of this phase can be a domain model in the form of an ER-diagram¹⁰⁸. In the second step a formal structure of the domain model is defined in an information logic model, referred to as the data model on platform-level.

Thereafter, in the step of execution of data integration, the actual work of integrating data begins. This step holds 4 sub tasks and an evaluation sub task in which dependencies and side effects are evaluated immediately after each sub task. This implies an iterative process and depending on the outcome of the evaluation some sub task might have to be executed more often.

The purpose of the first sub task is to determine a common observation unit that enables an integration of two or more data sets. The next sub task is responsible for pre-processing the data sets that might be necessary. Examples of possible pre-processing tasks could be adjustments of data formats and elimination outliers. If any side effects or dependencies are found in the evaluation then the logical models of the data sets need to be reviewed and possibly redefined depending on the impacts of the pre-processing. In the third sub task the actual data integration is executed and the side effect of this sub task is the development of one merged logical model based from the previous two or more data sets. The last sub task is the post processing of the integrated data set, and an example could be the replacement of missing values with estimated values. On platform level these steps are processed in a model that is referred to as the process model.

5.5 Modelling Language

5.5.1 Generic Definition and Dependencies

Having defined the modelling procedure the next step is to conceptualize the modelling language. To start with the model engineer must define the meta-models. On ADOxx® platform level the concept referred to as **Library**, found in the administration workspace, is the virtual place where all formalisms and structures of an instance of modelling language are assigned to. The platform offers a default Library including standard formalisms and structures that are inherited to the meta-model instances created by the method engineer. The inherence concept of the platform is demonstrated in Figure 25. The name of the instance library is arbitrary and can be freely chosen. The name of the prototype is referred to as “Data Integration 1.0”.

¹⁰⁸ Step one is not included in the implemented prototype.

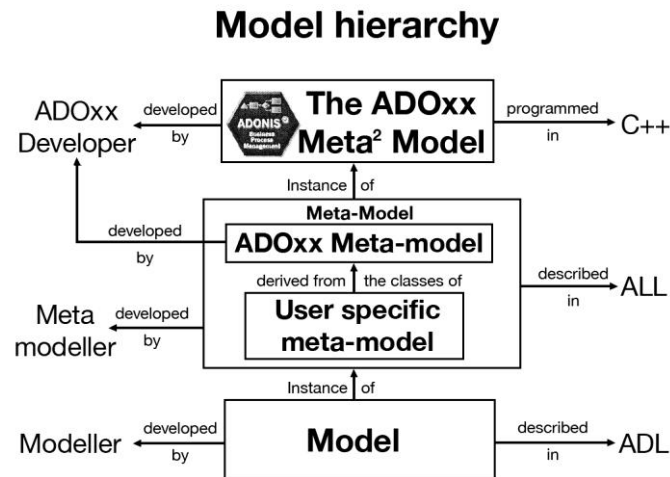


Figure 25: Model hierarchy¹⁰⁹

In the next step the instance libraries are assigned to model types. The idea of model types is the possibility to create different modelling components within the application library that corresponds to different modelling concepts of the method and to further describe the purpose of the model. Two model types that were derived for the method under discussion: *data model* and *process model*.

5.5.2 Modelling Language Conceptualization Results

The theoretical method is aiming for a modelling language that provides the modeller the possibility to include detailed knowledge about the data and to give support for the data transformation process by including dependencies and side effects. To fulfil these requirements the author of the method suggests the information logic and the process logic approaches (cf. 4 A Conceptual Method for Data Integration) and on platform-level these concepts were matched to a data meta-model and a process meta-model.

Data Meta-Model

The idea behind the data model is to give the modeller the possibility to describe and manage the formal structure of the domain data. It comprises six objects; population, data set, additional attributes observable units, variables and value domain and defined relation classes.

¹⁰⁹ Hintringer S., Karagiannis D. Schwab M., Specht G., Utz W.: ADOxx® - Customizing Einführung, Universität Wien, Department Knowledge Engineering, 2009, p.5.

For data integration problems automated routines are hardly possible but what can be achieved is a methodological guideline, a framework that breaks down the work of modelling in a stepwise manner. It aims to support and make the work easier for the modeller when building new models or adjust existing models.

Relation classes describe how the objects relate to each other. Within each object there is a possibility to add information, for example for the entity *Variables* the modeller can add the name of the attributes and within the object *Value Domain* one can add the measurement units for the attributes.

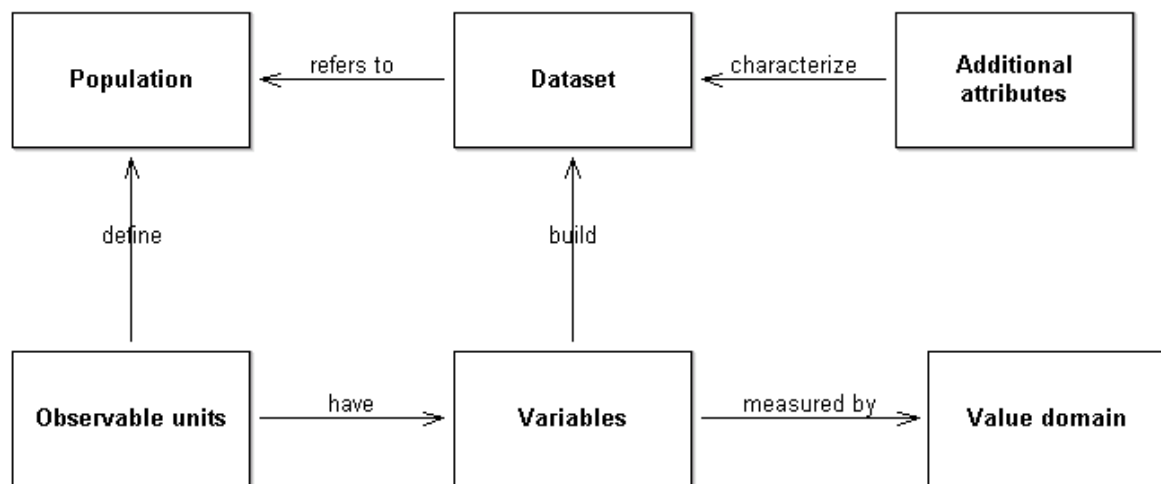


Figure 26: Class diagram of the data meta-model

Process Meta-Model

The purpose of the process meta-model is to fulfil the goal of the conceptual data integration approach: the development of a structured description of the transformation process. The objects and connectors of the process model correspond to the idea of creating building blocks of the transformation process and hence formalizing it. All objects included in the process meta-model offer the modeller to create a general documentation of each operation besides the possibility to add specific information to each object.

All phases of the data integration process including the pre-processing procedures, data integration operations and the post-processing phase are supported by the process model. By applying a generic approach for the building blocks the same type of objects can be used for all phases. For example the modeller can start by defining the method and criteria and then continue to describe the necessary evaluations, operations, decisions and results for the pre-

processing. The same type of objects can be used for the data integration and post-processing phase. Depending on the granularity and the degree of details that the user chose to include, each model can represent a complete phase or separate parts of the main phases. The different process models are connected to each other by the object “*Process*” and the connection to the data set models are enabled via the object “*Data*”.

A further property of the process meta-model is to ensure the inclusion of all necessary steps that have to be performed before within the phases. The connectors enforce the dependencies between these objects and oblige the user to process the objects in a certain chronological and/or logical order. For example an object of class “*Method*” must always be followed by the objects of class “*Criteria*” which in turn must be followed by an object of class “*Evaluation*”.

The conceptual data integration approach aims to consider side effects of the transformations used for data integration. Such transformations can be described in the objects of the process model and their side effects on the data set model are managed in the process model but the result of these side effects are visualised in the data set model. The connector “*affects*” is directed from the object “*Result*”, this object holds information about the results of the algorithms of transformation process, and is appointed to the object “*Data*” which is a gateway to the data set model including the objects that are effect by the results. The result of the translation of the ideas of the method to the process model is shown in Figure 27.

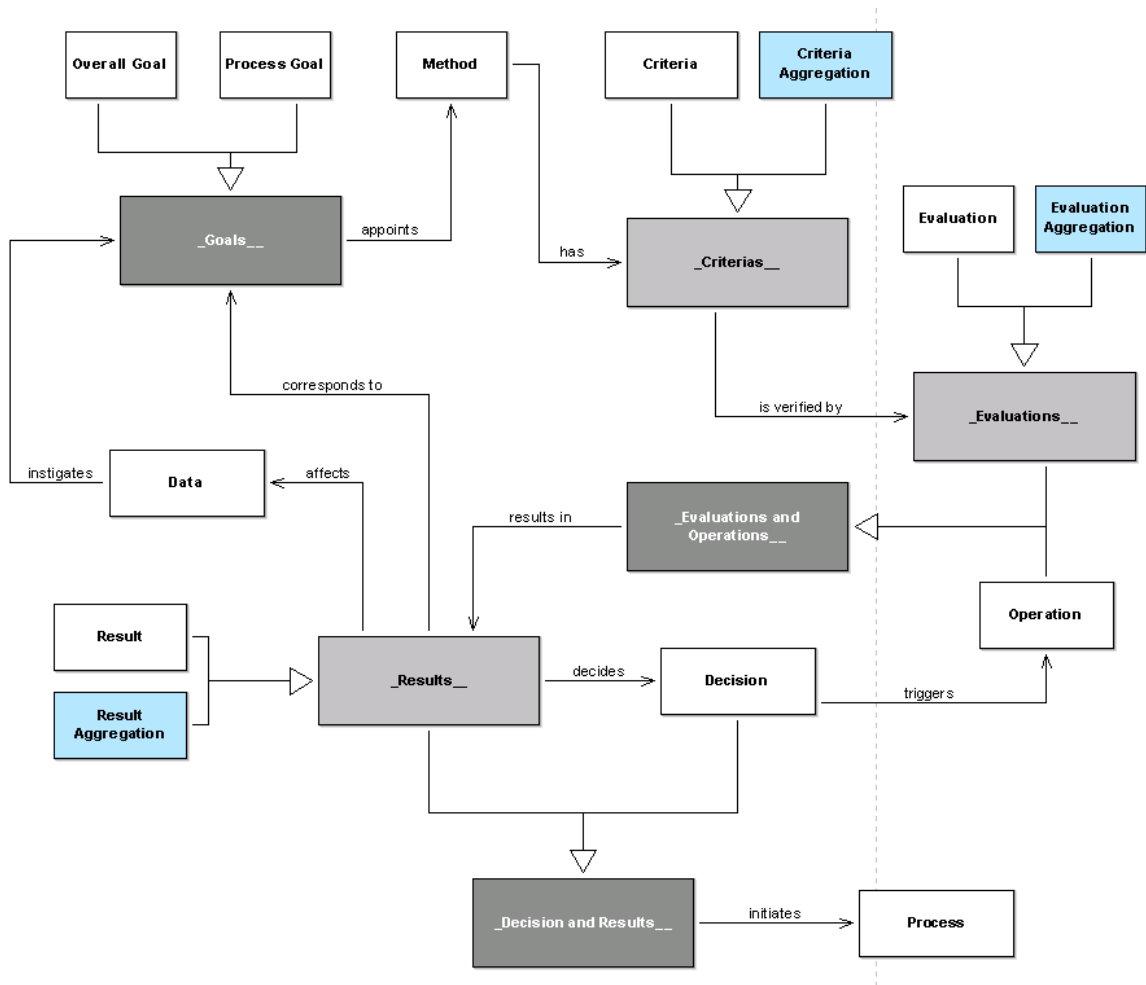


Figure 27: Class diagram of process meta-model

A structured documentation of the ideas of the syntax and semantic for the meta-models can be an indispensable help to fulfil the detailed specification of the modelling methods as well as provide details for the implementation on the platform. Such kind of blueprint (cf. Figure 26: Class diagram of the data meta-model and Figure 27: Class diagram of process meta-model) was the output of the first iteration on conceptualization and gave the method engineer a clear picture of the classes and its dependencies (semantic and the syntax) for the prototype. In a next step the details for each modelling construct as defined in the class diagrams in provided in accordance with the definition of a modelling language within the common framework introduced earlier (see Figure 28).

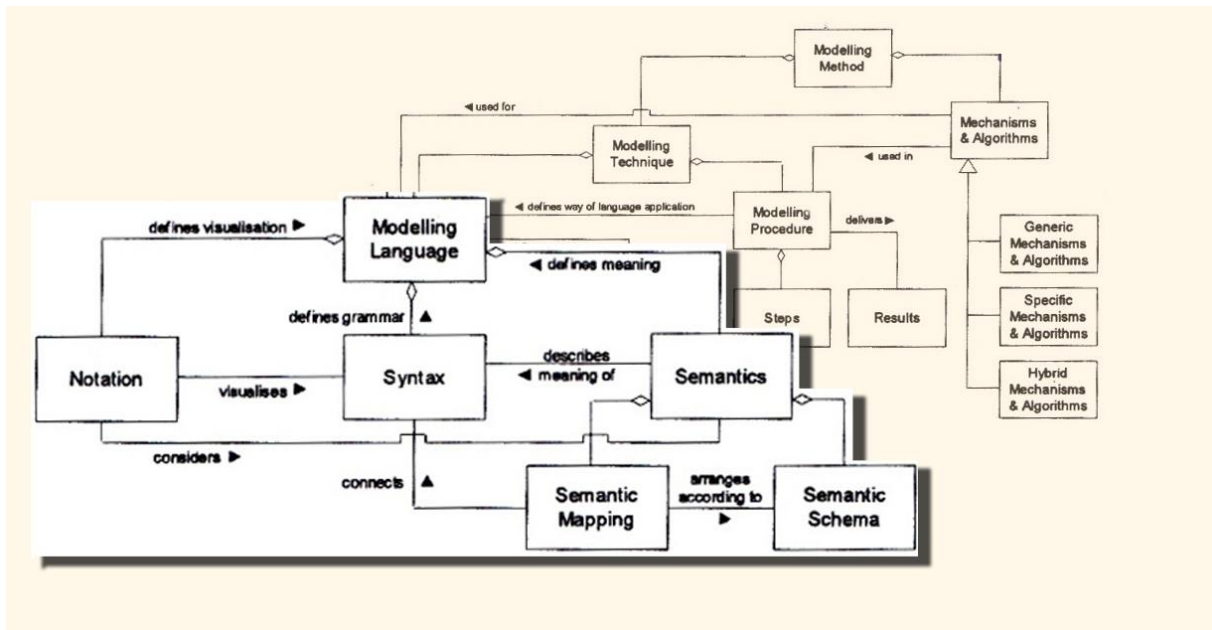


Figure 28: Modelling language¹¹⁰

5.5.3 Modelling Language Overview: Data Model

In the following table an overview of the elements of the data model is provided (see Table 1) followed by a detailed view on the conceptualization as well as implementation results. Each modelling construct is described using a common template that shows the syntax used, the attributes assigned as well as graphical and notebook notation.

¹¹⁰ cf. Schwab M., Utz W.: ADOxx® Customizing-Schulung, Methodenumsetzung, Erfahrungsberichte, Universität Wien, Department Knowledge Engineering, 2009, p.5.

Name / Table reference	Method relevance
Population	This class enables documentation about the population.
Reference: Table 3	
Observable Units	In the class Observable Units the modeller has the possibility to make a general description of the elements which are relevant for the data integration problem.
Reference: Table 4	
Data Set	Information such as data collection, number of records and attributes as well as extraction script can be added to the Data Set class.
Reference: Table 5	
Variables	In the class Variables the modeller can specify the attributes of the observable units.
Reference: Table 6	
Value Domain	The Value Domain class documents information such as measurement unit, value range and invalid values for the variables.
Reference: Table 7	
Additional Attributes	Any additional relevant information related to the data can be added to the class Additional Attributes.
Reference: Table 8	

Table 1: Data model, modelling language overview

5.5.4 Modelling Language Overview: Process Model


In the following table an overview of the elements of the data model is provided (see Table 2) followed by a detailed view on the conceptualization as well as implementation results. Each modelling construct is described using a common template that shows the syntax used, the attributes assigned as well as graphical and notebook notation.

Name / Table reference	Method relevance
Goals	A meta-class representing the classes Overall Goal and Process Goal in order to use a single relation class when connecting to other classes.
Not applicable	
Overall Goal	This class enables the modeller to define the overall goal of the data integration problem. A well-defined overall goal in a structured documentation supports the work of defining the user the intermediate objectives, the process
Reference: Table 9	

	goals.
Process Goal	The Process Goal gives the modeller the possibility to break down the overall goal into intermediate goals, in other words one goal for each sub process. Each goal is defined and documented the in this class.
Reference: Table 10	
Method	The method chosen to achieve the defined Process Goal can be described in the method class. Further the modeller can outline the necessary criteria in order to execute the method.
Reference: Table 11	
_Criteria__	A meta-class representing the classes Criteria and Criteria Aggregation in order to use a single relation class when connecting to other classes.
Not applicable	
Criteria	In the class Criteria the modeller can define the criteria required by the method. The criteria can be applied to specifically to each variable of the data but also a general applicable criterion can be described. If the defined method requires multiple criteria; multiple classes can be used within the same process.
Reference: Table 12	
Criteria Aggregation	The function of this class is a visual support. If multiple criteria are required, these criteria can be combined within the class Criteria Aggregation.
Not applicable	
_Evaluations and Operations__	A meta-class representing the classes _Evaluations__ and _Operations__ in order to use a single relation class when connecting to other classes.
Not applicable	
_Evaluations__	A meta-class representing the classes Evaluation and Evaluation Aggregation in order to use a single relation class when connecting to other classes.
Not applicable	
Evaluation	The evaluation class is used for describing the evaluation of the criteria. It is possible to make a general description about the evaluation and further specific information about the evaluation method and execution as well as the
Reference: Table 13	

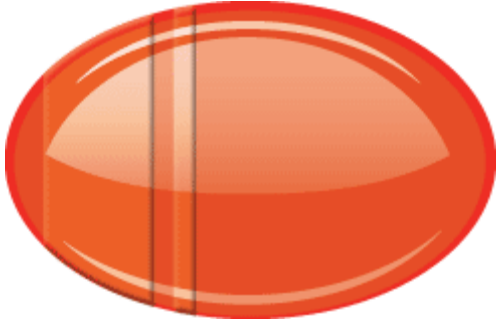
	tools used can be added.
Evaluation Aggregation	The function of this class is a visual support. If multiple evaluations are required these evaluations can be combined within the class Evaluation Aggregation.
Not applicable	
Operation	This class enable the user to make a general description of the actual operation(s) to execute the method. The operation method, execution code and information about any tool used can also be added here.
Reference: Table 14	
Decisions and Results	A meta-class representing the classes _Decisions__ and _Results__ in order to use a single relation class when connecting to other classes.
Not applicable	
_Results__	A meta-class representing the classes Result and Result Aggregation in order to use a single relation class when connecting to other classes.
Not applicable	
Result	The result of the evaluation of the criteria is documented in this class. For each variable the modeller can note if the result was acceptable or not. The class is also used to describe the result of operation class.
Reference: Table 15	
Result Aggregation	The function of this class is a visual support. If multiple results are required, these results can be combined within the class Result Aggregation.
Not applicable	
Decision	The Decision class documents the decision of the evaluation results. Additional to the possibility to register a positive or negative decision the modeller can add the motivation about the decision.
Reference: Table 16	

Table 2: Process model, modelling language overview

POPULATION	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSRING) Designate a description to the class.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\oval_03_244x159px.gif" BITMAP "db:\\oval_03_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold colour: black ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words

Notebook	
Population-80431 (Population)	
Name:	<input type="text" value="Population-80431"/>
Description:	<div><div></div><div></div></div>
Description	

Table 3: Class Population – modelling language details

OBSERVABLE UNIT	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSRING) Designate a description to the class.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\oval_01_244x159px.gif" BITMAP "db:\\oval_01_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words

Notebook


Observable units-81038 (Observable units)

Name:
Observable units-81038

Description:

Description

Table 4: Class Observable Units – modelling language details

DATA SET	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSRING) Designate a description to the class.	
Number of records (INTEGER) Designate the number of records of the data set class.	
Number of attributes (INTEGER) Designate the number of attributes of the data set class.	
Data collection (INTEGER) Designate a description of how data was collected.	Graph Rep:
Extract Script (LONGSTRING) Designate extract script to the data set class. The extract script defines how the data was extracted from its source.	<pre> GRAPHREP BITMAPINFO "db:\\oval_04_244x159px.gif" BITMAP "db:\\oval_04_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold color: black </pre>
Extracted from (STRING) Designate the data source where the extract script can be executed.	<pre> ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words </pre>

Notebook

Dataset-81032 (Dataset)

Name:

Dataset-81032

Description:

Number of records:

0

Number of attributes:

0

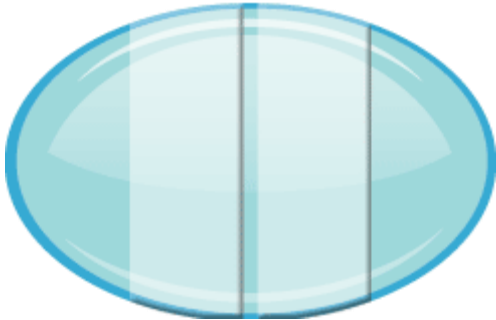
Data collection:

Extract script:

Extracted from:

Description

Table 5: Class Data Set – modelling language details

VARIABLES	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Attributes (RECORD) Designate the attributes of the variables.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\oval_09_244x159px.gif" BITMAP "db:\\oval_09_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words

Notebook

Variables-81050 (Variables)

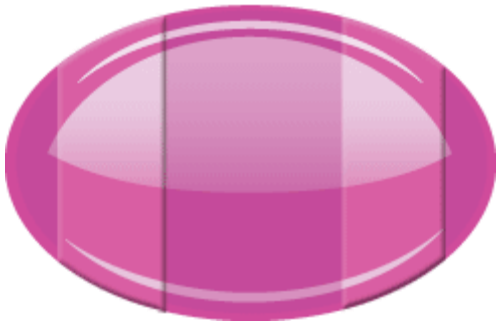
Name:

Attributes:

	Attribute name
1	<input type="text"/>
2	<input type="text"/>
3	<input type="text"/>

Description

Table 6: Class Variables – modelling language details

VALUE DOMAIN	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Value domains (RECORD) Designate the value domains of the attributes.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\oval_02_244x159px.gif" BITMAP "db:\\oval_02_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words

Notebook

Value domain-81041 (Value domain)

Name:

Value domain-81041

Description:

Value domains:


Attribute	Measurement unit	Value range	NULL	Invalid values	Anomaly values	Comment

Import variables

Import variables

Description

Table 7: Class Value Domain – modelling language details

ADDITIONAL ATTRIBUTES	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Data set proportion (DOUBLE) Designate the proportion between population and dataset.	
	Graph Rep: GRAPHREP BITMAPINFO "db:\\oval_07_244x159px.gif" BITMAP "db:\\oval_07_244x159px.gif" x:-1.5cm y:-1cm w:(244 / 96 * 1.2cm) h:(159 / 96 * 1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:2cm h:c:1.2cm line-break: words

Notebook

Additional attributes-81047 (Additional attributes)


Name:
Additional attributes-81047

Description:

Dataset proportion (%):
100,000000

Description

Table 8: Class Additional Attributes – modelling language details

OVERALL GOAL	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Purpose (LONGSTRINGS) Designate a purpose to the class	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\rect_01_300x159pxx.gif" BITMAP "db:\\rect_01_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

Notebook

Goal overall-81093 (Goal overall)

Name: Goal overall-81093

Description:

Purpose:


Description

Table 9: Class Overall Goal – modelling language details

PROCESS GOAL	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Purpose (LONGSTRINGS) Designate a purpose to the class.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\rect_02_300x159px.gif" BITMAP "db:\\rect_02_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words


Notebook	
Goal-81090 (Goal)	
Name:	Goal-81090
Description:	
Purpose:	

Table 10: Class Process Goal – modelling language details

Method	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Method criteria (LONGSTRINGS) Designate a description of the motivation why the selected method was chosen.	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\rect_03_300x159pxx.gif"
	BITMAP "db:\\rect_03_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm)
	FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

Notebook	
Method-81096 (Method)	
Name:	<input type="text" value="Method-81096"/>
Description:	<div><div></div><div></div><div></div></div>
Method criteria:	<div><div></div><div></div><div></div></div>

Table 11: Class Method – modelling language details

Criteria	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Criteria (LONGSTRINGS) Designate the criteria of the corresponding method class.	
Variable (LONGSTRINGS) Designate the applicable variables to the class.	
Method criteria (LONGSTRINGS) Designate a description of the motivation why the selected method was chosen.	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\rect_02_300x159pxx.gif" BITMAP "db:\\rect_02_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

The image displays two screenshots of a software interface titled "Criteria-81099 (Criteria)".


Top Screenshot:

- Name:** Criteria-81099
- Description:** (Empty text area)
- Right-hand pane:** Contains three buttons: "Description" (highlighted), "Criteria variables", and "Criteria miscellaneous".

Bottom Screenshot:

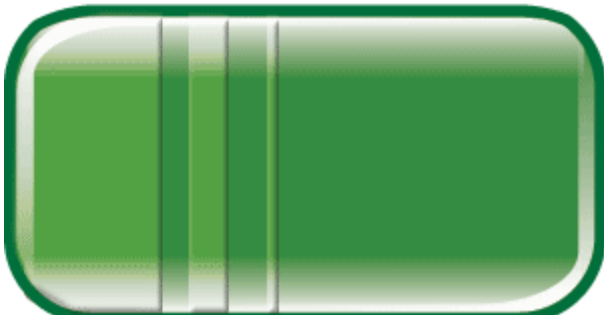
- Criteria:** A table with the following columns: Description, Mandatory, Operator, Value 1, Value 2, and Comment. The table is currently empty.
- Variable:** A table with the following columns: Information objectID, Information object name, Attribute, Measurement unit, Value range, NULL, and Invalid. The table is currently empty.
- Right-hand pane:** Contains three buttons: "Description", "Criteria variables" (highlighted), and "Criteria miscellaneous".

Table 12: Class Criteria – modelling language details

Evaluation	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Method (STRING) Designate method used for evaluation.	
Execution code (LONGSTRING) Designate the description of the code used for execution of the evaluation.	
Method executed in (STRING) Designate the program used for executing the evaluation.	Graph Rep: GRAPHREP BITMAPINFO "db:\\rect_04_300x159px.gif " BITMAP "db:\\rect_04_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

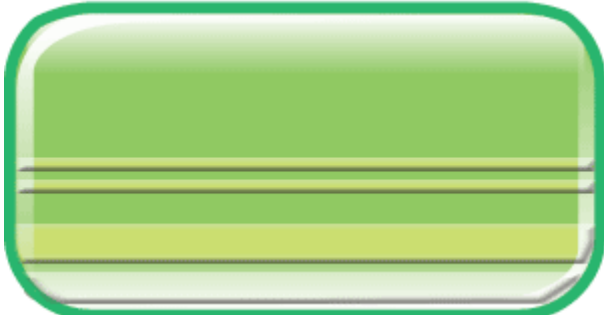
Notebook	
Evaluation-81117 (Evaluation)	
Name:	Evaluation-81117
Description:	
Method:	
Execution code	
Method executed in:	

Table 13: Class Evaluation – modelling language details

Operation	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Method (STRING) Designate method used for operation.	
Execution code (LONGSTRING) Designate the description of the code used for execution of the operation.	
Method executed in (STRING) Designate the program used for executing the operation.	Graph Rep:
Description (LONGSTRINGS) Designate a description of any other relevant information regarding the class.	GRAPHREP BITMAPINFO "db:\\rect_05_300x159pxx.gif" BITMAP "db:\\rect_05_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

Notebook	
<div> <div>Operation-81102 (Operation)</div> <div> <div>Name:</div> <div>Operation-81102</div> </div> <div> <div>Description:</div> <div></div> </div> </div> <div> <div>Description</div> <div>Method description</div> <div>Dataset integration</div> </div>	
<div> <div>Operation-81102 (Operation)</div> <div> <div>Method:</div> <div></div> </div> <div> <div>Execution code:</div> <div></div> </div> <div> <div>Method executed in:</div> <div></div> </div> </div> <div> <div>Description</div> <div>Method description</div> <div>Dataset integration</div> </div>	

Table 14: Class Operation – modelling language details

Result	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Result criteria (RECORD) Designate the result of criteria of the corresponding operation class	
Variable (RECORD) Designate the result of criteria of the corresponding operation class	
	Graph Rep:
	GRAPHREP BITMAPINFO "db:\\rect_06_300x159px.gif"
	BITMAP "db:\\rect_06_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black
	ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

Notebook

Result-81105 (Result)

Name:

Result-81105

Description:

Description

Result variables

Result-81105 (Result)

Result criteria:

Description	Mandatory	Operator	Value 1	Value 2	Result	Comment

Variable:

Information objectID	Information object name	Attribute	Measurement unit	Value range	NULL	Invalid


Import criteria

Import criteria

Description

Result variables

Table 15: Class Result – modelling language details

Decision	
AttRep Name (AttRep Type) / Syntax objective	Notation:
Name (STRING) Designate a name to the class.	
Description (LONGSTRING) Designate a description to the class.	
Motivation (LONGSTRING) Designate a description of the motivation why the decision was chosen	
	Graph Rep: GRAPHREP BITMAPINFO "db:\\rect_07_300x159pxx.gif" BITMAP "db:\\rect_07_300x159px.gif" x:-1.875cm y:-1cm w:(300 / 96 *1.2cm) h:(159 / 96*1.2cm) FONT bold color: black ATTR "Name" x:0cm y:0cm w:c:3cm h:c:1.6cm line-break: words

Notebook

Decision-81108 (Decision)

Name: Decision-81108

Decision:

Motivation:

Description

Table 16: Class Decision – modelling language details

5.6 Mechanism and algorithms

In the ADOxx[®] platform the set up for automated processing for meta-models are handled by the modelling method referred to as mechanisms and algorithms. In order to automatically validate the syntactical rules defined for the model, different functions for model usability were developed. These functions were developed by using the AdoScript, a macro language offered by the platform that is based on LEO syntax and follows a procedural structure. Further algorithms and mechanisms were not implemented due to the following two main reasons:

- **A generic framework with no specific algorithms**

The prototype was developed as to support a methodological framework that can include any data integration problem independently of algorithms used.

- **Automated routines hardly possible**

As recognized by the author of the conceptual model for data integration on which the prototype is based, automated routines are hardly possible for data integration problems¹¹¹. Instead a meta-model supporting a methodological framework for the data integration was developed.

Table 17 gives an overview of the implemented mechanisms of the prototype and Table 18 lists possible algorithms and mechanisms or tools and applications that could extend the prototype in order to tailor it for specific domains. Table 19 and Table 20 describe the functionalities and show details such as context, objective and code about the actual implemented mechanisms.

Functionality / Algorithm	Class	Reference	Type
Import Variables (PROGRAMCALL)	Value Domain (Data model)	Table 19	Model usability
Import Criterias (PROGRAMCALL) -	Result (Process model)	Table 20	Model usability

Table 17: Data model, algorithms and mechanisms overview

¹¹¹ Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009, p.129

Functionality / Algorithm	Type
Similarity Flooding ¹¹² (SF)	Matching A hybrid matching algorithm based on the of similarity propagation.
Artemis ¹¹³ (Analysis of Requirements: Tool Environment for Multiple Information Systems)	Matching An algorithm consisting of affinity-based analysis and hierarchical clustering of source schema elements.
Cupid ¹¹⁴	Matching A hybrid matching algorithm comprising linguistic and structural schema matching techniques
COMA ¹¹⁵ (COMbination ofMATCHing algorithms)	Matching A schema matching tool including a library of matching algorithms. It comprises a framework for combining obtained results, and a platform for the evaluation of the effectiveness of the different matchers. It contains 6 elementary matchers, 5 hybrid matchers, and one reuse-oriented matcher.
NOM ¹¹⁶ (Naive Ontology Mapping)	Matching Is grounded on the idea of composite matching from COMA.
QOM ¹¹⁷ (Quick Ontology Mapping)	Matching Is based on matching rules of NOM and is a successor of the NOM system. The approach of QOM is that loss of quality of matching algorithms is marginal but the improvement in efficiency can be increased significantly.
OLA ¹¹⁸ (OWL Lite Aligner)	Matching OLA belongs to the family of distance based algorithms which converts definitions of distances based on all the input structures into a set of equations.
Anchor-PROMPT ¹¹⁹ (an extension of PROMPT, also formerly known as SMART)	Matching This is an ontology merging and alignment tool. It based on a hybrid alignment algorithm using two ontologies as in input, and a set of anchors-pairs of related terms, which are identified with the

¹¹² Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.17.

¹¹³ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.18.

¹¹⁴ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.18.

¹¹⁵ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.18.

¹¹⁶ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.19.

¹¹⁷ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.19.

¹¹⁸ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.19.

¹¹⁹ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.19.

	help of string-based techniques (edit-distance test), or defined by a user, or another matcher computing linguistic similarity ontology merging and alignment tool with a sophisticated prompt mechanism for possible matching terms.
S-Match ¹²⁰	Matching A hybrid schema-based matching system with a composition at the element level. Its libraries contain 13 element-level matchers, see and 3 structure-level matchers.
Clio ¹²¹	A collaboration between IBM Almaden Research Center and the University of Toronto resulted in Clio which is a tool that based on the mapping set and the user's requirements a source executable query can be automatically be generated.
Merge ¹²²	Matching This is an algorithm which integrates two models based on common correspondences. It is independent to any implementation, e.g. database, XML or ontology.
BDK ¹²³	Matching This referred to as a "general technique" for data integration. With the approach of a generalisation of all schemas in a way that a binary merging operator, that is both commutative and associative, is defined.
HumMer ¹²⁴ (Humboldt-Merger)	Matching The HumMer is a tool that proceeds in three main steps; schema matching, duplicate detection and data fusion and conflict resolution. It allows xml schema and relation database as input and the output is one representation for each real world object.
Dumas ¹²⁵	Matching The Dumas is transforming the schema and one preferred source is chosen and all semantically similar names are renamed to match the preferred source.

Table 18: Possible algorithms and mechanisms overview

Details about the implemented algorithms and mechanisms for the Prototype are listed in the following tables.

¹²⁰ Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011, p.20.

¹²¹ Li Y., Liu D., Zhang W.: A Data Transformation Method Based On Schema Mapping, 2009, online in www at <http://www.gi.de/> (<http://goo.gl/oH2Yq>), accessed on 06.05.2011, p.76.

¹²² Hogg K.: Analysis of Data Integration, 2009, online in www at <http://www.manchester.ac.uk/> (<http://goo.gl/JP72h>), accessed on 18.04.2011, p.11.

¹²³ Hogg K.: Analysis of Data Integration, 2009, online in www at <http://www.manchester.ac.uk/> (<http://goo.gl/JP72h>), accessed on 18.04.2011, p.13.

¹²⁴ Hogg K.: Analysis of Data Integration, 2009, online in www at <http://www.manchester.ac.uk/> (<http://goo.gl/JP72h>), accessed on 18.04.2011, p.15.

¹²⁵ Hogg K.: Analysis of Data Integration, 2009, online in www at <http://www.manchester.ac.uk/> (<http://goo.gl/JP72h>), accessed on 18.04.2011, p.16.

Name of algorithm:
Import variables
Context:
Data model.
Objective / Requirement:
Import attributes defined in the class Variables into the class Value Domain.
Code:
<pre>#CC "AdoScript" INFOBOX (STR currentObjID)# WARNING MESSAGE# CC "AdoScript" WARNINGBOX ("Are you sure that you will overwrite the variables? \n This will reset all values of the table!") yes-no IF (endbutton="no") { EXIT } ##### GLOBALS ##### CC "Core" GET_CLASS_ID classname: ("Variables") SETG nVariablesClassID: (classid) #ClassID of the class "Variables" CC "Core" GET_CLASS_ID classname: ("Value domain") SETG nValueDomainClassID: (classid) #ClassID of the class "Value Domain" CC "Core" GET_ATTR_ID classid: (nVariablesClassID) attrname: ("Name") SETG nVariablesNameAttrID: (attrid) #AttributeID of the attribute "Name" of the class "Variables" CC "Core" GET_ATTR_ID classid: (nVariablesClassID) attrname: ("Attributes")</pre>

```

SETG nVariablesAttributesAttrID: (attrid)

#AttributeID of the attribute "Name" of the class "Value domain"

CC "Core" GET_REC_CLASS_ID attrid: (nVariablesAttributesAttrID)

SETG nRecAttributesClassID: (classid)

#RecordClassID of the record class "Attributes" of the class "Value domain"

CC "Core" GET_ATTR_ID classid: (nRecAttributesClassID) attrname: ("Attribute name")

SETG nRecAttributeNameID: (attrid)

#AttributeID of the attribute "Attribute Name" of the record class "Attributes"

CC "Core" GET_ATTR_ID classid: (nValueDomainClassID) attrname: ("Value domains")

SETG nValueDomainAttrID: (attrid)

#

CC "Core" GET_REC_CLASS_ID attrid: (nValueDomainAttrID)

SETG nRecValueDomainsClassID: (classid)

CC "Core" GET_ATTR_ID classid: (nRecValueDomainsClassID) attrname: ("Attribute name")

SETG nRecValueDomainAttributeNameID: (attrid)


# EMPTY TARGET RECORD #

CC "Core" GET_REC_ATTR_ROW_COUNT objid: (currentObjID) attrid: (nValueDomainAttrID)

SET nValueDomainNumberOfRows: (count)

FOR j from: (nValueDomainNumberOfRows) to: (1) by: (-1)

{

    CC "Core" GET_REC_ATTR_ROW_ID objid: (currentObjID) attrid: (nValueDomainAttrID) index: (j)

    SET nDeleteRowID: (rowid)

    CC "Core" REMOVE_REC_ROW objid: (currentObjID) attrid: (nValueDomainAttrID)
rowid: (nDeleteRowID)

}

```

```

# GET ALL CONNECTED VARIABLE OBJECTS #

# Returns connector IDs
CC "Core" GET_CONNECTORS objid:(currentObjID) in
SETL connectorIDs:(objids)

FOR connectorID in: (connectorIDs)
{
    CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL connectorID)
    SET fromobjID:(fromobjid)

    #DEMO Read Name

    #CC "Core" GET_ATTR_VAL objid:(fromobjID) attrid:(nVariablesNameAttrID)
    #CC "Core" GET_ATTR_VAL objid:(fromobjID) attrid:(nVariablesAttributesAttrID)
    CC "Core" GET_REC_ATTR_ROW_COUNT objid:(fromobjID) attrid:(nVariablesAttributesAttrID)
    SET nNumberOfRows: (count)

    # GET ALL VARIABLE NAMES #

    FOR i from: (1) to: (nNumberOfRows)
    {
        CC "Core" GET_REC_ATTR_ROW_ID objid:(fromobjID) attrid:(nVariablesAttributesAttrID)
index: (i)

        SET nRowID: (rowid)

        CC "Core" GET_ATTR_VAL objid:(nRowID) attrid:(nRecAttributeNameID)

        SET sAttributeNameValue:(val)

        # SET VARIABLES IN VALUE DOMAINS #

        CC "Core" ADD_REC_ROW objid:(currentObjID) attrid:(nValueDomainAttrID)

        SET nNewRowid: (rowid)

        CC "Core" SET_ATTR_VAL objid:(nNewRowid) attrid:(nRecValueDomainAttributeNameID)
val:(sAttributeNameValue)

    }

}

CC "AdoScript" INFOBOX ("Table has been updated")

```

Table 19: Algorithm for class Value Domain for Data Model

Name of algorithm:
Import criteria
Context:
Process model.
Objective / Requirement:
Import criteria defined in the class Criteria into the class Result.
Code:
<pre> #CC "AdoScript" INFOBOX (STR currentObjID) #WARNING MESSAGE# #CC "AdoScript" WARNINGBOX ("Are you sure that you would like to overwrite the criteria? \n This will reset all values of the table!") yes-no #IF (endbutton="no") #{ #EXIT #} ##### GLOBALS ##### #Returns ClassID of the class "Criteria" CC "Core" GET_CLASS_ID classname:("Criteria") SETG nCriteria_ClassID_CCriteria:(classid) # Returns AttributeID of "Criteria" of the class "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_ClassID_CCriteria) attrname:("Criteria") SETG nCriteria_AttrID_CCriteria:(attrid) #Returns RecordClassID of the Recordclass "Criteria" CC "Core" GET_REC_CLASS_ID attrid:(nCriteria_AttrID_CCriteria) SETG nCriteria_RecClassID_CCriteria:(classid) #Returns AttributeID of "Description" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Description") SETG nDescription_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns AttributeID of "Mandatory" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Mandatory") SETG nMandatory_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns AttributeID of "Value 1" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Operator") SETG nOperator_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns AttributeID of "Value 1" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Value 1") SETG nValue1_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns AttributeID of "Value 2" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Value 2") SETG nValue2_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns AttributeID of "Comment" of the Recordclass "Criteria" CC "Core" GET_ATTR_ID classid:(nCriteria_RecClassID_CCriteria) attrname:("Comment") SETG nComment_RecClassAttrID_RCCriteria_CCriteria:(attrid) #Returns ClassID of the class "Result" CC "Core" GET_CLASS_ID classname:("Result") SETG nResultClassID_CResult:(classid) # Returns AttributeID of "Result Criteria" of the class "Result" CC "Core" GET_ATTR_ID classid:(nResultClassID_CResult) attrname:("Result criteria") SETG nResultCriteria_AttrID_CResult:(attrid) </pre>

```

#Returns RecordClassID of the Recordclass "Result criteria"
CC "Core" GET_REC_CLASS_ID attrid:(nResultCriteria_AttrID_CResult)
SETG nResultCriteria_RecClassID_CCriteria:(classid)

#Returns AttributeID of "Description" of the Recordclass "Criteria"
CC "Core" GET_ATTR_ID classid:(nResultCriteria_RecClassID_CCriteria) attrname:("Description")
SETG nDescription_RecClassAttrID_RCResultCriteria_CCriteria:(attrid)

##### GET CRITERIAS FROM ALL "CRITERIA" CLASSES CONNECTED TO "RESULT" CLASSES VIA
"EVALUATION" CLASSES #####

#Return "Result" Connector IDs
CC "Core" GET_CONNECTORS objid:(currentObjID) in
SETL nResultConnectorIDs:(objids)

FOR connectorID in: (nResultConnectorIDs)
{
  ## Return "Evaluation" Class ID
  CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL connectorID)
  SET nEvaluationClassID:(fromobjid)

#Return "Evaluation" Connector IDs
CC "Core" GET_CONNECTORS objid:(nEvaluationClassID) in
SET nEvaluationConnectorIDs:(objids)

FOR connectorID in: (nEvaluationConnectorIDs)
{
  #Return "Criteria" Class IDs
  CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL connectorID)
  SET nCriteriaClassID:(fromobjid)

  #GET NUMBER OF ROWS IN THE ATTRIBUTE TABLE "CRITERIA"#
  CC "Core" GET_REC_ATTR_ROW_COUNT objid:(nCriteriaClassID)
  attrid:(nCriteria_AttrID_CCriteria)
  SET nCriteriaNrofRows: (count)

  #GET ALL DESCRIPTIONS FROM THE ATTRIBUTE TABLE "CRITERIA" IN THE CLASS "CRITERIA"
  #AND ADD TO ATTRIBUTE TABLE "RESULT CRITERIA" IN THE CLASS "RESULT" #
  FOR i from: (1) to: (nCriteriaNrofRows)
  {
    #GET ID FROM EACH ROW IN THE ATTRIBUTE TABLE "CRITERIA"#
    CC "Core" GET_REC_ATTR_ROW_ID objid:(nCriteriaClassID)
    attrid:(nCriteria_AttrID_CCriteria) index: (i)
    SET nCriteriaRowID: (rowid)

    #GET THE VALUE OF THE ID FROM EACH ROW IN THE ATTRIBUTE TABLE "CRITERIA"#
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nDescription_RecClassAttrID_RCCriteria_CCriteria)
    SET sDescriptionValue:(val)
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nMandatory_RecClassAttrID_RCCriteria_CCriteria)
    SET sMandatoryValue:(val)
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nOperator_RecClassAttrID_RCCriteria_CCriteria)
    SET sOperatorValue:(val)
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nValue1_RecClassAttrID_RCCriteria_CCriteria)
    SET sValue1Value:(val)
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nValue2_RecClassAttrID_RCCriteria_CCriteria)
    SET sValue2Value:(val)
    CC "Core" GET_ATTR_VAL objid:(nCriteriaRowID)
    attrid:(nComment_RecClassAttrID_RCCriteria_CCriteria)
    SET sCommentValue:(val)

    #SET IN VALUE DOMAINS #
    CC "Core" ADD_REC_ROW objid:(currentObjID) attrid:(nResultCriteria_AttrID_CResult)
    SET nResultRowid: (rowid)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
    attrid:(nDescription_RecClassAttrID_RCResultCriteria_CCriteria) val:(sDescriptionValue)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
    attrid:(nMandatory_RecClassAttrID_RCCriteria_CCriteria) val:(sMandatoryValue)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
  }
}

```

```

attrid:(nOperator_RecClassAttrID_RCCriteria_CCriteria) val:(sOperatorValue)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
attrid:(nValue1_RecClassAttrID_RCCriteria_CCriteria) val:(sValue1Value)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
attrid:(nValue2_RecClassAttrID_RCCriteria_CCriteria) val:(sValue2Value)
    CC "Core" SET_ATTR_VAL objid:(nResultRowid)
attrid:(nComment_RecClassAttrID_RCCriteria_CCriteria) val:(sCommentValue)
    }
}
}

#
#}
#
#CC "AdoScript" INFOBOX ("Table has been updated")
#

```

Table 20: Algorithm for class Result for Process Model

6 Conclusion

The result of the master thesis at hand is a prototypical implementation of the meta-model supporting “Data Integration for Business Analytics” resulting in a modelling toolkit for the end-user. Additionally it describes the intellectual process of transforming a theoretical method into a conceptual model that can be applied and implemented on the ADOxx[®] meta-model platform and the documentation of the actual implementation.

Besides the attached prototype the prototype is also available on the Open Model Platform (<http://www.openmodels.at/>). The contribution to the Open Model Platform aims to support the goal of extending the usage of models and to make it more accessible to more users. Using the prototype on the platform users has the possibility to model data integration tasks. The prototype implements two different types of models, the data model and the process model corresponding to a methodological guideline that constitutes the procedural foundation of the modelling method. Using the data model, the user has the possibility to describe and manage the formal structure of the domain data whereas the process model is used to make a structured description of the transforming process.

The base idea behind the work in this thesis is that a generic approach as implemented in the prototype enabling a methodological guidance framework can support any data integration problem, independently of algorithms and BI solutions and/or technology used. By customizing the model framework the prototype is extendable to specific algorithms and mechanisms or could be integrated and connected to other tools and applications in order to tailor it to different specific domains.

The major part of the work performed relates to the conceptualization, the process of turning the given theoretical method into the conceptual model. Besides acquiring the knowledge of the available modelling methods in ADOxx[®], it was indispensable to gain deep insights in the domain of “Data Integration for Business Analytics” and the specific method itself.

7 References

- [1] Batini C., Lenzerini M.: A Comparative Analysis of Methodologies for Database Schema Integration, 1986, online in WWW at <http://www.ubc.ca> (<http://goo.gl/45OIa>), accessed on 21.10.2010
- [2] Bergmayr, A., Karagiannis, D., Schwab, M.: i* on ADOxx®: A Case Study, in Fourth International i* Workshop (iStar'10) at CAiSE'10, 22nd International Conference on Advanced Information Systems Engineering (07-08 June 2010), pp. 92-97
- [3] Bernstein P. A., Rahm E.: A survey of approaches to automatic schema matching, in the VLDB Journal 10: 334–350, 2001
- [4] Blake R.: A Survey of Schema Matching Research, 2007, online in WWW at <http://www.umb.edu> (<http://goo.gl/9dOXr>), accessed on 30.10.2010
- [5] Bohannon p., Elnahrawy E., et al: Putting Context into Schema Matching, 2006, online in WWW at <http://www.vldb.org/> (<http://goo.gl/G5bPY>), accessed on 09.05.2011
- [6] Bright M. W., Pakzad S. H.: A Taxonomy and Current Issues in Multidatabase Systems, 1992, online in WWW at <http://www.mst.edu> (<http://goo.gl/psh36>), accessed on 27.11.2010
- [7] Cali A.: Reasoning in Data Integration Systems: why LAV and GAV are Siblings, 2003, online in WWW at <http://www.csd.uoc.gr/> (<http://goo.gl/xQBcQ>), accessed on 10.05.2011
- [8] Calvanese D., Giacomo G., Vardi Y. M.: Simplifying Schema Mappings, 2011, online in WWW at <http://www.edbt.org/index.php> (<http://goo.gl/OFFTw>), accessed on 10.05.2011
- [9] Cruz I. F., Xiao H.: The Role of Ontologies in Data Integration, 2005, online in WWW at <http://www.uic.edu/uic> (<http://goo.gl/0yRrR>), accessed on 29.11.2010
- [10] Cudré-Mauroux P.: Peer Data Management system, 2008, online in WWW at <http://www.csail.mit.edu> (<http://goo.gl/f7Gmo>), accessed on 18.10.2010
- [11] Dittrich K., Ziegler, R.: Three Decades of Data Integration - All Problems Solved?, 2004, online in WWW at <http://www.uzh.ch> (<http://goo.gl/8pu71>), accessed on 05.10.2010

- [12] Doan A., Halevy, A. Y.: Semantic-Integration, Research in the Database Community, A Brief Survey, 2005, online in WWW at <http://www.wisc.edu> (<http://goo.gl/392LN>), accessed on 05.12.2010
- [13] Dou D., Liu H.: An Exploration of Understanding Heterogeneity through Data Mining, in Proceedings of KDD'08 Workshop on Mining Multiple Information Sources (MMIS 2008), pp. 18-25, 2008
- [14] Fagin R., Kolaitis P., Nash A.: Towards a Theory of Schema-Mapping Optimization, 2008, online in WWW at <http://www.almaden.ibm.com/> (<http://goo.gl/fmcW8>), accessed on 09.05.2011
- [15] Fahl G.: Object Views of Relational Data in Multidatabase Systems, 1994, online in WWW at <http://www.uu.se> (<http://goo.gl/dLqZt>), accessed on 27.11.2010
- [16] Gal A., Trombetta A.: A Model for schema Integration in heterogeneous Databases, 2003, online in WWW at <http://www1.technion.ac.il> (<http://goo.gl/lGjsN>), accessed on 21.10.2010
- [17] Gartner: The Gartner Glossary of Information Technology Acronyms and Terms, 2004, online in WWW at <http://www.gartner.com> (<http://goo.gl/3JOxh>), accessed on 21.08.2010
- [18] Gawinecki M.: How schema mapping can help in data integration?, 2009, online in WWW at <http://www.agentgroup.unimore.it> (<http://goo.gl/EkR6w>), accessed on 20.10.2010
- [19] Gertz M.: Distributed Computing Architectures, 2003, online in WWW at <http://www.db.cs.ucdavis.edu> (<http://goo.gl/pPiCv>), accessed on 26.09.2010
- [20] Grossmann W.: Data Integration for Business Analytics: A Conceptual Approach, in Proceedings of Knowledge Science, Engineering and Management (KSME 2009), pp. 122-133, 2009
- [21] Grossmann, W., Höfferer, P., Karagiannis, D.: Open Model Initiative – A feasibility Study, 2008, online in WWW at <http://www.openmodels.at> (<http://goo.gl/bVw3I>), accessed on 15.08.2010
- [22] Hammer J., Pluempitiwiriawej C.: A Classification Scheme for Semantic and Schematic Heterogeneities in XML Data Sources, 2000, online in WWW at <http://www.cise.ufl.edu> (<http://goo.gl/2xgAr>), accessed on 08.10.2010

- [23] Hakimpour F.: Using Ontologies to Resolve Semantic Heterogeneity for Integrating Spatial Database Schemata, 2003, online in WWW at <http://www.uzh.ch> (<http://goo.gl/5gMyr>), accessed on 30.11.2010
- [24] Halevy A. Y.: Why Your Data Won't Mix Semantic Heterogeneity, 2005, online in WWW at <http://www.washington.edu> (<http://goo.gl/peCY2>), accessed on 02.12.2010
- [25] Härder T., Sauter G., Thomas J.: The Intrinsic Problems of Structural Heterogeneity and an Approach to their Solution, in the VLDB Journal 8:1, pp. 25-43, 1999
- [26] Hintringer S., Karagiannis D. Schwab M., Specht G., Utz W.: ADOxx® - Customizing Einführung, Universität Wien, Department Knowledge Engineering, 2009
- [27] Hogg K.: Analysis of Data Integration, 2009, online in www at <http://www.manchester.ac.uk/> (<http://goo.gl/JP72h>), accessed on 18.04.2011
- [28] Isik, O.: Business Intelligence Success: An Empirical Evaluation of the Role of BI Capabilities the Decision Environment, 2010, online in WWW at <http://digital.library.unt.edu/> (<http://goo.gl/Qm2bt>), accessed on 20.11.2010
- [29] Karagiannis, D., Kuehn H.: Metamodelling Platforms. IN: Bauknecht, K. Min Tjoa A, Quirchmayer G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin Heidelberg, p.182
- [30] Koch C.: Data Integration against Multiple Evolving Autonomous Schemata, 2001, online in WWW at <http://www.csd.uoc.gr> (<http://goo.gl/afmsi>), accessed on 11.10.2010
- [31] Kühn H.: Methodenintegration im Business Engineering, PHD Thesis, University of Vienna 2004
- [32] Kühn, H.: The ADOxx® Metamodelling Platform, Workshop “Methods as Plug-Ins for Meta-Modelling” in conjunction with “Modellierung 2010”, Klagenfurt (24-26 March 2010)
- [33] Larsson A. J., Seth A. P.: Federated Database Systems for Managing Distributed Heterogeneous and Autonomous Databases, in ACM Computing Surveys, Vol. 22, No. 3, 1990
- [34] Lenzerini M.: Data Integration: A Theoretical Perspective, 2002, online in WWW at <http://www.cs.ubc.ca> (<http://goo.gl/WJfRv>), accessed on 26.10.2010

- [35] Leser U.: Query Planning in Mediator Based Information Systems, 2000, online in WWW at <http://www2.informatik.hu-berlin.de> (<http://goo.gl/HyLpv>), accessed on 18.10.2010
- [36] Leser U., Naumann F.: Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Dataenquellen, dpunkt.verlag, 2007
- [37] Li Y., Liu D., Zhang W.: A Data Transformation Method Based On Schema Mapping, 2009, online in www at <http://www.gi.de/> (<http://goo.gl/oH2Yq>), accessed on 06.05.2011
- [38] Loshin D.: Business Intelligence, The Savy Manger's guide, Getting Onboard with Emerging IT, Morgan Kaufmann, 2003
- [39] Manakanatas D., Plexousakis D.: A Tool for Semi-Automated Semantic Schema Mapping: Design and Implementation, 2006, in WWW at <http://www.ics.forth.gr> (<http://goo.gl/t9eVG>), accessed on 21.10.2010
- [40] Moss T. L., Atre S.: Business Intelligence Roadmap: the complete project lifecycle for decision-support applications, Addison-Wesly, 2003
- [41] Object Management Group, online in WWW at <http://www.omg.org>, accessed on 05.11.2010.
- [42] PASW, Chapman P., Clinton J., Kerber R., Khabaza T., Reinartz T., Shearer C., Wirth R.: CRISP-DM 1.0 - Step-by-step data mining guide, 2000, online in WWW at <http://www.crisp-dm.org> (<http://goo.gl/kGvnA>), accessed on 04.12.2010
- [43] Pentaho-Kettle, online in WWW at <http://kettle.pentaho.com>, accessed on 04.12.2010
- [44] Power, D. J.: A Brief History of Decision Support Systems, 2007, online in WWW at <http://dssresources.com> (<http://goo.gl/dj9N1>), accessed on 20.05.2010
- [45] Rababaah H.: Distributed databases, fundamentals and research, 2005, online in WWW at <http://www.cs.iusb.edu> (<http://goo.gl/USU3E>), accessed on 26.09.2010
- [46] SAS, online in WWW at <http://www.sas.com> (<http://goo.gl/rlNUI>), accessed on 04.12.2010
- [47] Schwab M., Utz W.: ADOxx® Customizing-Schulung, Methodenumsetzung, Erfahrungsberichte, Univeristät Wien, Department Knowledge Engineering, 2009

- [48] Shvaiko P., Euzenat J.: A Survey of Schema-based Matching Approaches, 2005, online in www at <http://www.unitn.it/> (<http://goo.gl/bu2Nj>), accessed on 18.04.2011
- [49] Wells D.: Business Analytics – Getting the Point, 2010, online in WWW at <http://www.b-eye-network.com/> (<http://goo.gl/F9Yup>), accessed on 22.05.2010
- [50] Williams N., Williams S.: The Profit Impact of Business Intelligence, Elsevier Inc., 2007