Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms

Jose-Norberto Mazón a,∗ Juan Trujillo a Jens Lechtenbörger b

a Dept. of Software and Computing Systems
University of Alicante, Spain

b Dept. of Information Systems
University of Münster, Germany

Abstract

Successful data warehouse (DW) design needs to be based upon a requirement analysis phase in order to adequately represent the information needs of DW users. Moreover, since the DW integrates the information provided by data sources, it is also crucial to take these sources into account throughout the development process to obtain a consistent reconciliation of data sources and information needs. In this paper, we start by summarizing our approach to specify user requirements for data warehouses and to obtain a conceptual multidimensional model capturing these requirements. Then, we make use of the multidimensional normal forms to define a set of Query/View/Transformation (QVT) relations to assure that the conceptual multidimensional model obtained from user requirements agrees with the available data sources that will populate the DW. Thus, we propose a hybrid approach to develop DWs, i.e., we firstly obtain the conceptual multidimensional model of the DW from user requirements and then we verify and enforce its correctness against data sources by using a set of QVT relations based on multidimensional normal forms. Finally, we provide some snapshots of the CASE tool we have used to implement our QVT relations.

Key words: Conceptual Modeling, Data Warehousing, Requirements, Multidimensional Normal Forms, QVT

∗ Corresponding author. Tel.: +34-965-903772; fax: +34-965-909326.
Email addresses: jmazon@dlsi.ua.es (Jose-Norberto Mazón), jtrujillo@dlsi.ua.es (Juan Trujillo), lechten@wi.uni-muenster.de (Jens Lechtenbörger).

Preprint submitted to Elsevier 31 January 2007
1 Introduction

A data warehouse (DW) is commonly described as an integrated collection of historical data in support of decision making that structures information into facts and dimensions based on multidimensional (MD) modeling [8,9]. Since the DW integrates several data sources, the development of the underlying MD model has been traditionally guided by an analysis of these data sources [8]. In fact, several approaches have been defined in order to derive a conceptual MD model from given data sources [6,3,7]. Considering these data-driven approaches, MNFs (multidimensional normal forms) have been developed [11] to reason, in a rigorous manner, about the quality (faithfulness, completeness, avoidance of redundancies, summarizability) of a conceptual MD model derived from operational data sources.

Nevertheless, in these data-driven approaches the requirement analysis phase is overlooked, thus resulting in an MD model in which the user needs and expectations may not be satisfied [33]. To overcome this problem, several approaches [33,27,18,5] advocate a requirement-driven DW design process. However, hardly any of these approaches formally considers the data sources in the early stages of the development. Therefore, the correctness of the MD model with respect to the data sources cannot be assured and the DW repository cannot be properly populated from these data sources.

In order to reconcile these two points of view (data-driven and requirement-driven), a Model Driven Architecture (MDA) [23] framework for the development of DWs has been described in [17,15]. Within this approach a conceptual MD model of the DW repository is developed from user requirements. This initial MD model must be then reconciled with the data sources in order to assure its correctness, which is the major contribution of the present paper. The following step is to apply model transformations to obtain several logical models as a base for the implementation of the DW repository. Within this development framework, up to now, we have defined (i) a framework to specify user requirements for DWs [18], (ii) a UML (Unified Modeling Language) [26] profile that contains the necessary stereotypes in order to carry out the conceptual MD modeling successfully [13], (iii) a set of guidelines for deriving a conceptual MD model from user requirements [18], and (iv) a set of QVT (Query/View/Transformation) relations in order to automatically derive several logical representations of the conceptual MD model [17,14,15]. An overview of this overall framework, including the reconciliation of data sources and MD conceptual model by means of MNFs, is shown in Fig. 1.

In the short version of this paper [16], we presented a set of Query/View/Transformation (QVT) relations in order to check the correctness of the conceptual MD model against the available data sources within our DW development framework.
Fig. 1. Overview of our approach for the MD modeling of DWs.

(see Fig. 1). These QVT relations are based on MNFs proposed in [11]. The QVT language allows us to easily integrate this approach in our MDA framework [17,15], while MNFs enable us to formalize the relationship between the data sources and the conceptual MD model of the DW repository.

In this paper, we extend the short version [16] as follows: (i) We provide a new section describing our approach for specifying user requirements for DWs. (ii) We now provide the complete set of the QVT relations in order to check the correctness of the conceptual MD model against the available data sources. (iii) We include a new aspect concerning on how to apply these QVT relations. (iv) We extend the case study, with more details and figures that clarify how to use our approach in real world scenarios. Finally, (v) we provide a new section explaining how we have implemented our QVT relations in a CASE tool in order to facilitate its application in real world case studies.

The motivation of our approach is as follows: since the DW integrates the information provided by source databases, it is important to check (in early stages of the development) if the requirement-driven conceptual MD model agrees with the available data sources in order to assure that (i) the DW repository will be properly populated from data sources, (ii) the analysis potential provided by the data sources is captured by the conceptual MD model, (iii) redundancies are avoided, and (iv) optional dimension levels, i.e., levels allowing NULL values, are controlled via specialization/generalization to enable context-sensitive summarizability and to avoid inconsistent queries.

To illustrate these benefits, consider the following running example, which is inspired by the sample scenario of [10]. We assume that the conceptual MD model for the banking domain shown in Fig. 2 has been derived from user requirements without taking data sources into account, e.g., according to the proposal presented in [18], which is briefly described in Section 3.1. The notation of Fig. 2 is based on our UML profile for MD modeling presented in [13] (see Section 4.2 for details). The figure models Account facts which are composed of several measures (balance, turnover, interest, and customerAge)
and described by dimensions Organization, Product, Time, and Customer. For the sake of clarity, we only focus on the Customer dimension.

Every customer is described in terms of a unique identification number, a name, and a date of birth. Every customer lives in a city which is described with a name and a population. Moreover, customers may be associated with job, gender, industry branch, and contact person. Finally, a city belongs to (Rolls-upTo) exactly one region and exactly one district, while a region belongs to exactly one state. This model represents a geographical classification where every region falls into exactly one state, while districts and states appear to be unrelated. From a conceptual perspective, this classification seems reasonable.

However, the data sources shown in Fig. 3 (using part of the relational CWM metamodel explained in Section 4.1) provide geographical information where every district falls into exactly one state, while regions and states are unrelated. Consequently, (i) the source information concerning regions and states cannot be represented faithfully under the MD model and (ii) potential for roll-up queries from level district to level state is not represented, i.e., analysis potential is lost.

Moreover, the MD model does not represent the structural information that industry branches and contact persons are assigned only to company customers while job and gender are only applicable to private customers, which poses challenges for summarizability and complicates querying (see [11,12]). Finally, while it certainly makes sense to analyze the age structure of customers, the measure age is not specific to accounts but only to customers. Thus, this measure should be moved to a different fact schema. To summarize, based on
schema information for the data sources, the conceptual MD model shown in Fig. 2 should be improved in a number of ways to obtain the “better” model shown in Fig. 4. Indeed, in this paper we show how to apply QVT relations, which are derived from MNFs, to obtain the model shown in Fig. 4 from the model shown in Fig. 2 by taking data sources shown in Fig. 3 into account.
The remainder of this paper is structured as follows: Related work is put into perspective next, before necessary background concerning requirement analysis in DWs, QVT and MNFs is collected in Section 3. Our approach is presented in Section 4 by describing our data source model as well as our conceptual MD model, and defining QVT relations based on MNFs. The application of sample QVT relations, as well as their implementation and execution is illustrated in Section 5. The paper ends with conclusions and suggestions for future work in Section 6.

2 Related work

In this section, we briefly describe the most relevant approaches for both data-driven and requirement-driven DW development.

Concerning data-driven approaches, in [3], the authors present the Multidimensional Model, a logical model for MD databases. The authors also propose a general design method, aimed at building an MD schema starting from an operational database described by an Entity-Relationship (ER) schema.

In [6], the authors propose the Dimensional-Fact Model (DFM), a particular notation for the DW conceptual design. Moreover, they also propose how to derive a DW schema from the data sources described by ER schemas. Also in [31], the building of a conceptual MD model of the DW repository from the conceptual schemas of the operational data sources is proposed.

In [7], the authors present a method to systematically derive a conceptual MD model from data sources. In this paper a preliminary set of multidimensional normal forms is used to assure the quality of the resulting conceptual model.

Although in each of these data-driven approaches the design steps are described in a systematic and coherent way, the DW design is only based on the operational data sources, what we consider insufficient because the final user requirements are very important in the DW design [33].

Concerning requirement-driven approaches, in [33] an approach is proposed in order to both determine information requirements of DW users and match these requirements with actual data sources. However, no formal approach is given in order to match requirements with data sources.

In [27], the authors propose a requirement elicitation process for DWs by grouping requirements in several levels of abstraction. Their process consists of identifying information that supports decision making via information scenarios. In this process, a Goal-Decision-Information (GDI) diagram is used.
Although the derivation of GDI diagrams and information scenarios is described, the relationships between information scenarios and requirements are not properly specified. Moreover, requirements are not reconciled with data sources in order to obtain a conceptual MD model.

In [5], the authors present a framework to obtain a conceptual MD model from requirements. This framework uses the data sources to shape hierarchies and user requirements are used to choose facts, dimensions and measures. However, the authors do not present a formal way to reconcile data sources with the conceptual MD model.

To summarize, these requirement-driven approaches do not formalize the relation between the data sources and the requirements to verify and enforce the correctness of the resulting DW. Therefore, we propose to use MNFs [11] in a systematic manner, thus formalizing the development of the DW repository by means of (i) obtaining a conceptual MD model from user requirements, and (ii) verifying and enforcing its correctness against the operational data sources. Details on MNFs are presented in the next section.

3 Background

In this section, we first outline our approach for (i) defining information requirements in DWs, and (ii) obtaining a conceptual MD model from them. Afterwards, we provide a brief overview of the building blocks of our approach, namely Query/View/Transformation and multidimensional normal forms.

3.1 Requirement analysis in DWs

A requirement analysis stage allows designers to develop a DW which meets the real needs of organizations, thus increasing the success of a DW project [29]. The way of proceeding for specifying information requirements in the DW domain presents some particularities that call for specialized methods [34]. One of the main drawbacks is related to the inherent difficulty of understanding and specifying decision processes, so there is a major gap between those that are experts in the domain and their requirements on the one hand (i.e., decision makers) and those that are experts in the design and construction of the DW which satisfies the requirements (i.e., DW developers), on the other hand.

To overcome this problem, in this paper, the requirements for the DW are defined according to the approach presented in [18]. This approach bridges
the above-mentioned gap by means of describing the DW requirements within its business environment. The main issue of this approach is to specify business goals related to DW users (i.e., decision makers). The motivation is twofold [29]: (i) the aim of a DW is to provide users with enough information in a suitable way to improve decision making and accomplish business goals; and (ii) decision makers often express their information needs in general expectations of the DW to improve their business, which can be described by means of goals. Therefore, defining requirements for a DW should take into account the business goals of DW users, and then from these goals the information requirements will be more easily discovered.

Business goals are considered to form a hierarchy of goals depending on the level of abstraction:

- **Strategic goals** represent the highest level of abstraction. They are main objectives of the business process. They are thought as changes from a current situation into a better one. For example: “increase sales”, “increase number of customers”, “decrease cost”, etc. Their fulfilment causes an immediate benefit for the organization.

- **Decision goals** represent the medium level of abstraction. They try to answer the question: “how can a strategic goal be achieved?”, and they aim to take the appropriate actions to fulfil a strategic goal. For example “determine some kind of promotion” or “open new stores”. Their fulfilment only causes a benefit for the organization if it helps to reach strategic goals, since decision goals only take place within the context of a strategic goal.

- **Information goals** represent the lowest level of abstraction. They try to answer the question: “how can decision goals be achieved in terms of information required?”, and they are related to the information required by a decision goal to be achieved. For example “analyze customer purchases” or “examine stocks”. Their fulfilment helps to achieve decision goals and they only happen within the context of a decision goal.

In order to define a goal hierarchy two strategies can be adopted [32]: (i) discovering goals by refinement, or (ii) discovering goals by abstraction. The former consists on asking “how” questions over goals already identified (“how can this goal be satisfied?”), while the latter is driven by “why” questions (“why is this goal useful?”).

Once the hierarchy of goals is defined, information requirements can be directly obtained from the information goals above-described. These requirements are related to interesting measures of business processes (contained in facts) and the context for analyzing these measures (dimensions and their hierarchies).

In order to model these goals and information requirements for DWs, we adapt the i* framework [35,36]. This framework consists of two models: the strategic
dependency (SD) model to describe the dependency relationships among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The elements of i* that we use in our approach are represented in Fig. 5. These elements are actors, intentional elements (goals, tasks or resources) and their relationships (dependency, means-end or task-decomposition). In this subsection we focus on exemplifying how to use i* for DW domain; we refer the reader to [35] for a further and general explanation of i*.

An SD model for DWs must take into account that the organization depends on the DW to obtain proper information to achieve its goals. So, business goals and information that the DW must provide to achieve business goals must be addressed in the SD model. Therefore, users (i.e., decision makers), and the DW under construction are the main actors in the SD model. On the other hand, the SR model provides a more detailed level of modeling internal relationships of each actor (i.e., decision makers and the DW). For decision makers, every goal must be specified according to the above-described classification of goals. Information requirements for decision makers have to be derived from goals and specified as tasks in the SR model. According to these information requirements, the SR model for the DW actor is specified with the necessary intentional elements. From this SR model, we will be able to derive a conceptual MD model. An overview of the approach is shown in Fig. 6.

Several guidelines are presented in [18] to support the specification of the SD and the SR models for DWs and the derivation of a conceptual MD model. Afterwards, the resulting conceptual model could be also refined in order to add some new features. The corresponding SD and SR models for our running example are shown in Fig. 7-9. We would like to point out that, for understandability reasons, we only focus on capturing the requirements related to the customer dimension.

Since our example is drawn from the banking domain (focused on the “account” fact and the “customer” dimension), the main actor is the bank man-
ager. It is assumed that the main goal of this actor is to increase the number of customers. On the other hand, the DW itself is another actor whose aim is to provide the bank manager with adequate information about the bank accounts. Taking into account these considerations, the SD model is specified in Fig. 7: there are two actors ("bank manager" and "data warehouse"). The goal of the "bank manager" is "increase number of customers". The information required from the "data warehouse" is related to bank accounts ("accounts information"), and it is specified as a resource dependency.

The SR model for the "bank manager" actor is shown in Fig. 8. This model is built starting from the strategic goal "increase number of customers". From this strategic goal, two decisional goals are obtained: "give incentives when an account is opened" and "determine the best feasible interest". From each of these decision goals, the following information goals have been obtained (as it is shown in Fig. 8): "study accounts by customer profile", "examine customer age", "study accounts by customer branch", and "study accounts by customer job". These information goals must be related to information requirements which can be provided by the DW. These information requirements are as follows (see Fig. 8): "analyze balance and turnover by customer location", "analyze balance and turnover by customer branch", "analyze balance and turnover by customer job and gender", "analyze customer age for accounts", "analyze balance, turnover and interest by branch", and "analyze balance, turnover and interest by job".

The next step is to specify the SR model for the DW actor. From the information requirements previously defined we specify the following SR model (see Fig. 9): the main goal is "provide information about accounts". The measures
Table 1
Information requirements and their corresponding MD elements.

<table>
<thead>
<tr>
<th>Information requirement</th>
<th>Fact attribute</th>
<th>Base or dimension attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyze balance and turnover by customer location</td>
<td>Balance, turnover</td>
<td>City, District, Region, State</td>
</tr>
<tr>
<td>Analyze balance and turnover by customer branch</td>
<td>Balance, turnover</td>
<td>Branch</td>
</tr>
<tr>
<td>Analyze balance and turnover by customer job and gender</td>
<td>Balance, turnover</td>
<td>Job, gender</td>
</tr>
<tr>
<td>Analyze customer age for accounts</td>
<td>Customer age</td>
<td>-</td>
</tr>
<tr>
<td>Analyze balance, turnover and interest by branch</td>
<td>Balance, turnover, interest</td>
<td>Branch</td>
</tr>
<tr>
<td>Analyze balance, turnover and interest by job</td>
<td>Balance, turnover, interest</td>
<td>Job</td>
</tr>
</tbody>
</table>

Fig. 7. SD model for our bank case study.

related to information requirements are “balance”, “turnover”, “interest”, and “customer age” which are considered as resources. On the other hand, the task “provide information by customers” is considered as general context of analysis, while the following resources, among others (see Fig. 9), give a more detailed context: “customer”, “city”, “district”, “region”, and “state”.

Finally, from this SR model for the DW, we obtain a conceptual MD model of Fig. 2. This model provides enough information in a suitable way to accomplish the information requirements and business goals of decision makers. For an overview of which MD elements are created to obtain the MD model of Fig. 2 from the SR model of the DW (Fig. 9), please see Table 1.

3.2 Query/View/Transformation language

The MOF 2.0 Query/View/Transformation (QVT) language [24] is a standard approach for defining formal relations between MOF-compliant models. Furthermore, QVT is an essential part of the MDA standard as a means of defining formal and automatic transformations between models.

QVT consists of two parts: declarative and imperative. The declarative part
provides mechanisms to define relations that must hold between the model elements of a set of candidate models (source and target models). This declarative part can be split into two layers according to the level of abstraction: the relational layer that provides graphical and textual notation for a declar-
ative specification of relations, and the core layer that provides a simpler, but
verbose, way of defining relations. The imperative part defines operational
mappings that extend the declarative part with imperative implementations
when it is difficult to provide a purely declarative specification of a relation.

In this paper, we focus on the relational layer of QVT. This layer supports the
specification of relationships that must hold between MOF models by means
of a relations language. A relation is defined by the following elements:

- **Two or more domains**: each domain is a distinguished set of elements
  of a candidate model (source or target model). This set of elements must
  be matched in that model by means of patterns. A domain pattern can be
  considered as a template for elements, their properties and their associations
  that must be located, modified, or created in a candidate model in order to
  satisfy the relation. A relation between domains can be marked as check-only
  (labeled as C) or as enforced (labeled as E). When a relation is executed in
  the direction of a check-only domain, it is only checked if there exists a valid
  match in the model that satisfies the relationship (without modifying any
  model if the domains do not match); whereas for a domain that is enforced,
  when the domains do not match, model elements are created, deleted or
  modified in the target model in order to satisfy the relationship. Moreover,
  for each domain the name of its underlying metamodel is specified.

- **When clause**: it specifies the conditions under which the relation needs to
  hold (i.e., precondition).

- **Where clause**: it specifies the condition that must be satisfied by all model
  elements participating in the relation (i.e., postcondition).

Defining relations by using the QVT language has the following advantages:
(i) it is a standard language, (ii) relations are formally established and auto-
matically performed, and (iii) relations can be easily integrated in an MDA
approach.

### 3.3 Multidimensional normal forms

The formal guidelines that we are using to formulate our QVT relations in
the following are the three multidimensional normal forms 1MNF, 2MNF, and
3MNF presented in [11]. Here, we recapitulate the essence of these normal
forms informally. The reader is referred to [11] for formal definitions. Prelim-
inarily we recall that within a conceptual MD model the terminal dimension
levels of a fact are those that are attached immediately to the dimensions, i.e.,
those that provide the finest level of detail within each dimension.

The goal of 1MNF is to ensure that a conceptual MD model “matches” with
the information provided by the source databases. More specifically, 1MNF is
characterized by four conditions as follows:

1. **Faithfulness.** The functional dependencies (FDs) implied by the MD model must be a subset of those observed in the source databases. (Otherwise, some source data cannot be represented under the MD model.)

2. **Roll-up completeness.** The FDs among dimension levels contained in the source databases must be represented as roll-up arcs in the MD model. (Otherwise, analysis potential is lost.)

3. **Derivation completeness.** The FDs among sets of measures contained in the source databases must be represented via derivation formulas in the MD model. (Otherwise, derivation relationships are lost.)

4. **Avoidance of redundancies.** Each measure must be assigned to a fact in such a way that the terminal dimension levels of the fact form a key for the measure without transitive dependencies. (Otherwise, a measure is recorded redundantly at the “wrong” level of detail. E.g., in Fig. 2 in the Introduction measure customerAge was repeated for each account owned by a customer.)

In addition to 1MNF, the normal forms 2MNF and 3MNF aim to control optional dimension levels by means of so-called contexts of validity. Roughly, a context of validity for an optional dimension level explains the occurrence (and absence) of structural null values (such as NULL for industry branch of private customers in Fig. 2) based on the values of so-called discriminating levels. E.g., for the scenario in Fig. 2, we may assume that in the data sources there is an attribute customerType with values “private” and “company”, which acts as discriminating level, such that a customerType of “private” implies NULL for Branch and ContactPerson, whereas “company” implies NULL for Job and Gender. As argued in [12] and elaborated in more detail in [11], structural NULL values can and should be avoided by suitable introduction of specialization hierarchies. In fact, in [10] it has been shown that 3MNF allows to construct a class hierarchy of dimension levels with an implementation as relational database that avoids null values. Note that such a class hierarchy is indeed part of the improved model shown in Fig. 4.

Importantly, the MD model considered in [10,11] does not provide mechanisms for specialization/generalization explicitly, which necessitates the use of context dependencies. As in this paper we consider a richer MD model that explicitly supports subclassing, we are able to explain the occurrence of NULL values directly by moving an attribute with structural NULL values into the appropriate subclass. As a result, we obtain a simplified approach.

As explained in [11,12] control over NULL values enables context-sensitive summarizability (e.g., if an analyst rolls up from individual customers to industry branches, then schema information explains that the context of analysis has changed to a subclass of all customers) and avoids inconsistent queries.
(e.g., a query such as “group private customers by industry branch” can be rejected based on schema information).

4 Checking correctness of the conceptual MD model

In this section, we present our approach to check the correctness of a conceptual MD model with respect to the source databases. To this end, we present a set of QVT relations based on MNFs and obtain their inherent desirable design objectives: The resulting conceptual MD model faithfully represents the data sources and captures their analysis potential completely, redundancies are avoided, and NULL values are controlled to allow context-sensitive summarizability and avoid contradictory queries. Our approach consists of two main phases:

- The elements of the data sources are marked as dimensional elements (fact, dimension, measure and so on). We assume that the model which represents the data sources is a relational representation of the data sources in third normal form.
- A set of QVT relations between the data source model and the conceptual MD model (previously derived from user requirements) are applied, thus checking and enforcing that the conceptual MD model is aligned with data sources.

In the following subsections we present both data source model and conceptual MD model, as well as the set of QVT relations.

4.1 Data source model

We assume that the data source model is the relational representation of the data sources in third normal form. (Note that third normal form is not a restriction as well-known algorithms such as Synthesis [1] can transform any input schema into third normal form.) In particular, we use the CWM (Common Warehouse Metamodel) relational metamodel [22] in order to specify this data source model. The CWM relational metamodel is a standard to represent the structure of data resources in a relational database. This relational metamodel allows us to represent tables, columns, primary keys, foreign keys, and so on. Since every CWM metamodel is MOF-compliant [22], it can be used as source or target for QVT relations [24]. In this paper, we use the part of the relational metamodel shown in Fig. 10.

The data source model must be marked before the QVT relations can be
applied. Marking models is a technique that provides mechanisms to extend elements of the models in order to capture additional information [23,20]. Marks are used in MDA to prepare the models in order to guide the matching between them. A mark represents a concept from one model, which can be applied to a corresponding element of another model. These marks indicate how every element of the source model must be matched. In our approach, the data source model is marked by appending a suffix to the name of each element according to the conceptual MD model. Table 2 shows how the data source model must be marked. Several guidelines are given in [3,6] to properly identify the MD elements in the data sources and perform the marking.

### 4.2 Conceptual MD model

The conceptual modeling of the DW repository is based on a UML profile for MD modeling presented in [13]. This profile contains the necessary stereotypes in order to elegantly represent main MD properties at the conceptual
Table 3
Main stereotypes of our UML profile for MD modeling of DWs.

<table>
<thead>
<tr>
<th>Stereotypes</th>
<th>Description</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact</td>
<td>Classes of this stereotype represent facts in a MD model, consisting of measures (the values being analyzed).</td>
<td>![Fact Icon]</td>
</tr>
<tr>
<td>Dimension</td>
<td>Classes of this stereotype represent dimensions in a MD model, consisting on hierarchy levels.</td>
<td>![Dimension Icon]</td>
</tr>
<tr>
<td>Base</td>
<td>Classes with this stereotype represent dimension hierarchy levels in a MD model consisting of dimension attributes (descriptive information about the values being analyzed).</td>
<td>![Base Icon]</td>
</tr>
<tr>
<td>FactAttribute</td>
<td>Attributes of this stereotype represent attributes of a fact (i.e., measures) in a MD model. It can represent a derived measure, if a derivation rule is included.</td>
<td>![FactAttribute Icon]</td>
</tr>
<tr>
<td>DimensionAttribute</td>
<td>Attributes of this stereotype represent attributes of a dimension hierarchy level (i.e., base) in a MD model.</td>
<td>![DimensionAttribute Icon]</td>
</tr>
<tr>
<td>Descriptor</td>
<td>Attributes of this stereotype represent descriptor attributes of a dimension hierarchy level (i.e., base) in a MD model.</td>
<td>![Descriptor Icon]</td>
</tr>
<tr>
<td>Rolls-UpTo</td>
<td>Associations of this stereotype represent relationships between two levels (i.e., bases) of a classification hierarchy in a MD model. Role $R$ represents the direction in which the hierarchy rolls up, whereas role $D$ represents the direction in which the hierarchy drills down.</td>
<td>![Rolls-UpTo Icon]</td>
</tr>
</tbody>
</table>

Fig. 11. Extension of the UML with the stereotypes for MD modeling used in this paper.

level by means of a UML class diagram. An overview of our UML profile is given in Table 3 and Fig. 11. Apart from these defined stereotypes the generalization/specialization relationships of UML are used for suitably representing optional dimension levels.

We note that this UML profile includes further MD features (degenerate dimensions, degenerate facts, non-strict hierarchies, and so on), which are not taken into account in this paper, since only the characteristics related to MNFs are considered.
Our profile is formally defined and uses the Object Constraint Language (OCL) [25] for expressing well-formed rules of the newly defined elements (see Table 3), thereby avoiding an arbitrary use of the profile. We refer reader to [13] for a further explanation of this profile and its corresponding OCL constraints.

4.3 QVT relations

In the following, the QVT relations that align data sources and MD models are described: Check1MNF1, Check1MNF1, Check1MNF1, Check1MNF1, Check1MNF2, Check1MNF3, and Check1MNF4 are based on the 1MNF; Check2MNF3MNF is based on both 2MNF and 3MNF.

The relations are applied as follows: first Check1MNF1, Check1MNF1, Check1MNF1, and Check1MNF1 are applied in order to check that the FDs of the MD model are contained in those of the sources (first condition of the 1MNF); since both domains are check-only, it is only checked whether there exists a valid match that satisfies these relations without modifying any model if the domains do not match. If the check fails, there typically is no automatic solution, and the DW developer must redesign the conceptual MD model. (E.g., in our example given in Fig. 2, the user requirements express that Regions roll-up to States, whereas the data sources do not provide this information. Thus, either the conceptual model has to be modified or the source data has to be aligned with the model.) Otherwise, i.e., if the check succeeds, the remaining relations can be applied to properly modify the conceptual MD model (according to the second, third, and fourth condition of 1MNF as well as according to 2MNF and 3MNF). Therefore, these QVT relations not only check the correctness of the conceptual MD model according to the data sources, but also enforce this correctness by creating the necessary elements of the conceptual MD model until each relation holds.

In order to graphically show how to apply every developed QVT relation, the approach defined in [19] is followed. This approach is based on specifying a map of QVT relations, thus indicating the flow of execution. In this kind of map, QVT relations are related to each other by means of several kinds of relationships, in two of which we are interested: composition (several relations are composed regardless of execution order) and dependency (a relation must be executed before another relation). The map of QVT relations of our approach is shown in Fig. 12. This map indicates that first a set of relations to verify the first condition of the 1MNF is applied. Once these relations are successfully executed, it is possible to apply the remainder of the defined QVT relations.
Throughout the checks, we assume that the names of corresponding elements in both models are equal (apart from the previously added marks) according to a linguistic approach based on name similarity [28]. This issue is captured in the \textit{when} clause of each relation.

\subsection{Verify 1MNF (first condition).}

According to the first condition of 1MNF, for every FD in the conceptual MD model we have to check that there is a corresponding FD in the data source model, i.e., the FDs implied by the MD model must be a subset of those observed in the source databases. Therefore, this condition assures that the source data can be properly represented under the MD model. We have defined one QVT relation (see Fig. 13) for each case in which an FD arises in the conceptual MD model in order to check if the same FD occurs in the data source model. These cases are as follows:

(1) Descriptor determines DimensionAttributes. This case is checked by Check1MNF1_1 (see Fig. 13): The elements related to the conceptual MD model are a Base (b), a Descriptor (d) and a DimensionAttribute (da). These elements of the conceptual MD model must be matched against a set of elements of the data source model: a table (t) with a column (c1) which is part of the primary key (pk). This table is marked as a Dimension or Base.
(m_n,t) and the column (c1) is marked as a Descriptor (m_n_c1). There is also a column (c2) which is functionally determined by the primary key. This column is marked as a DimensionAttribute (m_n_c2).

(2) A Rolls-upTo association is an FD between hierarchy levels (Bases). This case is checked by Check1MNF1_2 (see Fig. 13): A set of elements that represent two Bases (b1 and b2) related by means of a Rolls-upTo association must be checked against the following pattern in the data source model: a set of elements that represents a table (t1) with a foreign key (fk) that references the other table (t2). This represents a many-to-one relationship in a third normal form relational database. Furthermore, table t1 must be marked as Dimension or Base, t2 as Base and foreign key fk as Rolls-upTo.

(3) Derived measures. This case is checked by Check1MNF1_3 (see Fig. 13): If there is a derived FactAttribute (with a derivation rule) in the MD model, then in the data sources there must be a procedure which implements this derivation rule.

(4) Dimensions (and their terminal dimension levels) functionally determine FactAttributes (i.e., measures). This case is checked by Check1MNF1_4 (see Fig. 13): In this relation, a set of elements of the conceptual MD model that represents the association between a Dimension (d), together with its terminal dimension level, i.e., Base (b) and a Fact (f) together with its attributes (fa) is matched against the following pattern of the data sources: a table (t1) with a column (c), a primary key (pk) which contains a foreign key that references another table (t2). Table t1 is marked as a Fact, while table t2 is marked as Dimension and column c is marked as FactAttribute.

4.3.2 Verify 1MNF (second condition).

The relation Check1MNF2 checks the second condition of 1MNF, i.e., roll-up completeness (the FDs among dimension levels contained in the source databases must be represented as roll-up arcs in the MD model). Therefore, if this relation holds then there exists a Rolls-upTo association between bases in the conceptual MD model if there is an FD between columns of different tables in the data source model. This relation (see Fig. 14) is outlined as follows: The data source model consists of a portion of the CWM relational metamodel that represents a table t1 (marked as a Dimension or Base) with a foreign key fk (marked as Rolls-upTo) to another table t2 (marked as Base). This set of elements must be matched against the following elements of the conceptual MD model: a Base (b1) with the name of table t1, a Base (b2) with the name of table t2, and the corresponding properties to define the Rolls-upTo association between the two Bases. We wish to point out that role R in the Rolls-upTo association is provided by the table t1 (owner of the foreign key), since it indicates the direction in which the hierarchy rolls up,
Fig. 13. Check conditions of 1MNF.

whereas role D is provided by the table t2, since it represents the direction in which the hierarchy drills down.

Fig. 14. Check the second condition of 1MNF.

4.3.3 Verify 1MNF (third condition).

The relation Check1MNF3 checks the third condition of 1MNF, i.e., derivation completeness. If a certain measure can be computed from a set of other
measures, then this computation indicates an FD among measures. Therefore,
the FDs among measures that appear in the data source model should be
reflected as derived FactAttributes of the conceptual MD model. The relation
that verifies this condition (Check1MNF3) is shown in Fig. 15. It checks that
if a procedure exists in data sources in order to compute the value of a column
(pdk_return parameter) from a set of other columns (pdk_in parameters), a
derived FactAttribute (with a derivation rule) in the target model must be
matched.

Fig. 15. Check the third condition of 1MNF.

4.3.4 Verify 1MNF (fourth condition).

The relation Check1MNF4 checks the fourth condition of 1MNF, i.e., avoidance of redundancies (see Fig. 16). Therefore, each measure must be assigned
to a Fact (as a FactAttribute) in such a way that the terminal dimension lev-
els of the Fact form a key for the measure without transitive dependencies. A
set of elements that represents a table (marked as a Fact) with a foreign key
to another table (marked as a Dimension) corresponds to a set of elements
which represents a Fact together with a Dimension and a Base that repres-
ents the terminal dimension level. The data source columns that represent
FactAttributes are marked as measures.

Fig. 16. Check the fourth condition of 1MNF.
4.3.5 Verify 2MNF and 3MNF.

The relation Check2MNF3MNF checks the conditions of 2MNF and 3MNF (see Fig. 17). These normal forms control optional dimension levels by avoiding structural NULL values. The aim of this relation is check or enforce a class hierarchy of dimension levels in order to avoid these NULL values. As in this paper we consider a conceptual MD model that explicitly supports subclassing, the Check2MNF3MNF relation covers both 2MNF and 3MNF by moving an attribute with structural NULL values into the appropriate subclass.

This relation is shown in Fig. 17. A table (t1) with two columns, an optional column (l0) and a discriminating level (l) is matched against a generalization hierarchy: a superclass is a base (b1), and a subclass is other base (b2) with a dimension attribute that corresponds to the optional column. Furthermore, we use context dependencies as schema level constraints to identify discriminating levels, so in the when clause there is a function (isDiscriminatingLevel) that checks whether the column l is a discriminating level according to the table t1 and the other column l0.

![Fig. 17. Check the 2MNF and the 3MNF.](image-url)

5 Case study

In this section, we show how our QVT relations are properly applied. Furthermore, we explain how to implement and execute these defined QVT relations.

5.1 Sample applications of QVT relations

The sample scenario previously introduced in the Introduction (see Fig. 2–4) is used to show how our QVT relations are properly applied to assure the
correctness of the conceptual MD model of the DW repository against data sources.

In our scenario the QVT relations are applied according to the process described in Fig. 12. First, a subset of the QVT relations is executed to check the first condition of the first MNF. In our scenario, this verification process warns us about two main issues regarding the faithfulness of the conceptual MD model. On the one hand, the Rolls-upTo association between Region base and State base in the initial conceptual MD model (see Fig. 2) does not correspond to any FD in the data sources, since the Check1MNF1_2 relation does not hold. Therefore, this conceptual MD model must be modified by removing the Rolls-upTo association, for example. On the other hand, the Check1MNF1_4 relation states that dimensions must functionally determine fact attributes. However, according to the available data sources (see Fig. 3), the terminal dimension levels of the fact do not form a key for the Customer-Age measure, which implies that this measure should not be recorded as in Fig. 2. Therefore, the solution is to remove the CustomerAge fact attribute from the model (and to create a customer related MD model that incorporates this measure). Regarding the other QVT relations, they are successfully executed. Therefore, after applying these two changes the data sources can be properly represented under the conceptual MD model.

Once the first set of relations has been executed, the remaining relations can be applied to check and enforce the correctness of the conceptual MD model against the available data sources, as shown in Fig. 12, thus obtaining the conceptual MD model of Fig. 4. An example of the application of the QVT relations in our scenario is given in Fig. 18. On the left hand side of this figure the marked model of the available data sources is represented. On the right hand side, the resulting conceptual MD model after applying the set of QVT relations (presented in the previous section) is represented. In the middle of the figure, the applied relations between both models are represented. In the next subsections, the application of these relations is described in-depth.

5.1.1 Verify second condition of 1MNF

The second condition of 1MNF is verified by the relation Check1MNF2. Once this relation is applied, it checks or enforces the following:

- Relation Check1MNF2 checks and enforces that FK_ToState_ROLLS, a foreign key in the District_BASE table referencing the State_BASE table (which embodies a many-to-one relationship between districts and states), is represented via a Rolls-upTo association between District base and State base in the conceptual MD model. We point out that this Rolls-upTo association was missing in the requirement-driven conceptual MD model (recall
Fig. 18. Sample applications of QVT relations.
Relation Check1MNF2 checks that FK.ToCity_ROLLS, a foreign key in the Customer_DIM table referencing the City_BASE table, is represented via a Rolls-upTo association between Customer_BASE and City_BASE in the conceptual MD model. Figure 19 presents the application of this relation in detail. On the left hand side of this figure the pattern of the data source model according to the Check1MNF2 relation is shown. This pattern corresponds to the piece of conceptual MD model presented on the right hand side of the figure.

Fig. 19. Sample application of the Check1MNF2 relation.

Relation Check1MNF2 checks that FK.ToRegion_ROLLS, a foreign key in the City_DIM table referencing the Region_BASE table, is represented via a Rolls-upTo association between City_BASE and Region_BASE in the conceptual MD model.

Relation Check1MNF2 checks that FK.ToDistrict_ROLLS, a foreign key in the City_DIM table referencing the District_BASE table, is represented via a Rolls-upTo association between City_BASE and District_BASE in the conceptual MD model.

5.1.2 Verify fourth condition of 1MNF

The fourth condition of 1MNF is verified by the Check1MNF4 relation. This relation checks that the Account_FACT table, its primary key (PK_Account), foreign key (FK.To_Customer) to the Customer_DIM table, and its columns (Balance_MEASURE, Turnover_MEASURE, and Interest_MEASURE,) correspond to the Account fact (including fact attributes) and the Customer dimension (including the terminal dimension level Customer base).

5.1.3 Verify 2MNF and 3MNF

In order to verify 2MNF and 3MNF, the Check2MNF3MNF relation is applied. The enforcement of this relation creates subclasses of the Customer base in the conceptual MD model, whose names are determined by the values of the
discriminating level Type_DA: company and private. Furthermore, it enforces that the optional columns Job_DA and Gender_DA in the data source model belong to the private subclass in the conceptual MD model, while the optional columns Branch_DA and ContactPerson_DA belong to the company subclass of the Customer base.

5.2 Implementation of QVT relations

The defined set of QVT relations are implemented and executed in an MDA-compliant CASE tool (Borland Together Architect [2]). Together Architect is a tool designed by Borland to support developers in the design of software applications by using several MDA features. One of the most valuable features of Borland Together Architect is the QVT language in order to implement transformations and relations between models. However, this tool only implements the imperative part of the latest QVT specification [24]. Therefore, we have implemented the imperative versions of the declarative relations described in Section 4.3. In spite of this, since declarative relations are clearer and more understandable than imperative ones, throughout this paper declarative relations are shown. This section illustrates the implementation and execution of one of the the QVT relations (Check1MNF2) in Borland Together Architect, and its application to our running example.

Borland Together Architect is based on the Eclipse development platform [4], so it can be extended by means of plugins in order to add more features and new functionality. In order to implement the set of QVT relations, two new plugins have been developed: (i) a plugin for MD modeling (according to the UML profile described in section 4.2), and (ii) a plugin that comprises the relational metamodel of CWM in order to model data sources. Once these plugins have been developed and integrated into the Eclipse platform, the facilities provided by Borland Together Architect can be used to develop our MNF-based QVT relations. The source code of the Check1MNF2 relation is shown in Fig. 20.

In order to execute this relation on the one hand we have specified the conceptual MD model for the running example (Fig. 2) by using the developed plugin of our UML profile for MD modeling of DWs. This model is presented in Fig. 22. On the other hand, the available data sources (Fig. 3) are specified via the developed CWM plugin, which is shown in Fig. 21.

After the execution of the QVT relations we obtain the model shown in Fig. 23. In this model, we focus on the elements related to the Check1MNF2 relation: Base classes and the Rolls-upTo associations between them. Once this relation is executed, the resulting MD model shown in Fig. 23 has the following
features:

- Rolls-upTo associations, which are represented by many-to-one relationships in the data source model, are preserved in the resulting MD model. These Rolls-upTo associations are those between Customer and City, City and District, and City and Region.

- It is enforced that the missing association between District and State in the requirement-driven conceptual MD model is represented as a Rolls-upTo association, according to the many-to-one relationship between districts and states in the data source model. Furthermore, we wish to point out that the Rolls-upTo association between Region and State is not longer present in the MD model.

Although Borland Together Architect is a widely used and powerful CASE tool, we have encountered two main pitfalls during the implementation stage. On the one hand, since the QVT standard is very recent, the QVT relational part is still not supported by Borland Together Architect. Thus, we had to adapt our QVT relations to the supported operational mappings (imperative part of QVT). On the other hand, the UML metamodel provided by Borland Together Architect is not fully compliant with the OMG specification, so we had to adapt the implementation of our MD profile to this metamodel.
In this paper, we have presented an approach to assure the correctness of a conceptual MD model of the DW repository according to the data sources that will populate this repository. This approach is outlined as follows: We first obtain the conceptual MD model of the DW from user requirements and then we verify and enforce its correctness against data sources by using a set of QVT relations based on MNFs. By using MNFs, we can assure that the conceptual MD model satisfies certain desirable properties such as faithfulness, completeness, avoidance of redundancies, and context-sensitive summarizability. Furthermore, QVT relations allow us to integrate this approach into an MDA framework for the development of DWs.

Our immediate future work is to extend our approach by defining QVT relations in order to automatically transform the conceptual MD model into logical models that are closer to the relational implementation. Furthermore, non-strict hierarchies, many-to-many relationships between a fact and a dimension, degenerate facts, and other MD issues should be taken into account. Therefore, MNFs will also assure the correctness of these logical models.

On the other hand, an emerging standard called Architecture-Driven Modern-
Fig. 22. Implementation of the initial MD model of the running example.

Fig. 23. MD model after applying the set of QVT relations.

Figurative (ADM) is being proposed [21] in order to define a set of interoperability metamodels that facilitate the collection, analysis, refactoring and transformation of existing systems. Since this standard aims to incorporate a scenario to facilitate the identification of relevant data that need to be analyzed, reconciled, validated and loaded into the DW repository, our intention is to align our approach with ADM.

Finally, we plan to experiment with other emerging implementations of QVT, like SmartQVT [30], to overcome the encountered problems in the implementation of our QVT relations.

7 Acknowledgements

This work has been partially supported by the METASIGN (TIN2004-00779) project from the Spanish Ministry of Education and Science, by the DADAS-MECA project (GV05/220) from the Valencia Ministry of Enterprise, University and Science (Spain), and by the DADS (PBC-05-012-2) project from the Castilla-La Mancha Ministry of Education and Science (Spain). Jose-Norberto
Mazón is funded by the Spanish Ministry of Education and Science under a FPU grant (AP2005-1360).

References


