Data & Knowledge Engineering 69 (2010) 1138–1157

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

A framework for multidimensional design of data warehouses from ontologies

Oscar Romero [a,*], Alberto Abelló [b]

[a] Universitat Politècnica de Catalunya - BarcelonaTech, Dept. Llenguatges i Sistemes Informàtics, Barcelona, Spain
[b] Universitat Politècnica de Catalunya - BarcelonaTech, Dept. d'Enginyeria de Serveis i Sistemes d'Informació, Barcelona, Spain

ARTICLE INFO

Available online 15 July 2010

Keywords:
OLAP
Multidimensional design
Ontologies

ABSTRACT

The data warehouse design task needs to consider both the end-user requirements and the organization data sources. For this reason, the data warehouse design has been traditionally considered a *reengineering* process, guided by requirements, from the data sources.
Most current design methods available demand highly-expressive end-user requirements as input, in order to carry out the exploration and analysis of the data sources. However, the task to elicit the end-user information requirements might result in a thorough task. Importantly, in the data warehousing context, the analysis capabilities of the target data warehouse depend on what kind of data is available in the data sources. Thus, in those scenarios where the analysis capabilities of the data sources are not (fully) known, it is possible to help the data warehouse designer to identify and elicit unknown analysis capabilities.
In this paper we introduce a *user-centered* approach to support the end-user requirements elicitation and the data warehouse multidimensional design tasks. Our proposal is based on a reengineering process that derives the multidimensional schema from a conceptual formalization of the domain. It starts by fully analyzing the data sources to identify, without considering requirements yet, the multidimensional knowledge they capture (i.e., data likely to be analyzed from a multidimensional point of view). Next, we propose to exploit this knowledge in order to support the requirements elicitation task. In this way, we are already conciliating requirements with the data sources, and we are able to fully exploit the analysis capabilities of the sources. Once requirements are clear, we automatically create the data warehouse conceptual schema according to the multidimensional knowledge extracted from the sources.

1. Introduction

Data warehousing systems were conceived to support decision making within organizations. According to [17], a data warehousing system is a *collection of methods, techniques, and tools used to support knowledge workers — e.g., senior managers, directors, etc. — to conduct data analysis that helps with performing decision making processes and improving information resources*. Typically, relevant data for decision making is extracted from the organization data sources, transformed (i.e., cleaned and homogenized) and finally integrated within a huge repository of data (the *data warehouse*), in what is known as the ETL (extraction/transform/loading) process. The data warehouse provides a single and detailed view of the organization, and it is intended to be exploited by means of the *exploitation tools*, which provide different mechanisms to navigate and perform analysis tasks over the data warehouse. Among the different kinds of exploitation tools, *OLAP (On-line Analytical Processing) tools* have gained relevance in the last years so much so that the data warehousing and OLAP concepts are now tightly related. OLAP tools are

* Corresponding author.
E-mail addresses: oromero@lsi.upc.edu (O. Romero), aabello@essi.upc.edu (A. Abelló).

0169-023X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2010.07.007

intended to facilitate information analysis and navigation through the business data based on the *multidimensional* paradigm. The multidimensional view of data is distinguished by the *fact/dimension* dichotomy, and it is characterized by representing data (i.e., the fact of interest) as if placed in an n-dimensional space (with as many axes as dimensions of analysis of interest). This paradigm allows to easily understand and analyze data showing the different points of view from where a subject can be analyzed. In consequence, the multidimensional model fits for non-expert users like knowledge workers (from here on, the data warehouse *end-users*). For example, a typical fact of interest would be the business `sales`, whereas its typical dimensions of analysis would be the `item` sold, `where` it was sold (i.e., the place) and `when` (i.e., the time). One fact and several dimensions of analysis form what is known as *multidimensional schema* or *star-schema*. Nowadays, it is widely accepted that the conceptual schema of a data warehouse must be structured according to the multidimensional model, so that it can be exploited by OLAP tools.

Like in most information systems, the data warehouse design has been typically carried out manually, and the experts' knowledge and experience are crucial to identify relevant multidimensional knowledge contained in the sources. Data warehousing systems need to consider both the end-user requirements and the organization data sources. In this context, the end-user requirements often come as business queries or service level agreements (SLAs), and represent the end-user analytical necessities, whereas the data sources are needed to know from where to extract the required data (and how to eventually populate the target data warehouse) in order to give answers to the information requirements. For this reason, the data warehouse design task has been considered a *reengineering* process, ideally guided by requirements, from the data sources: i.e., creating a data warehouse does not require the addition of new information but rearrange the existing information (indeed, the data warehouse is nothing else than a strategic view on the organization data). In this sense, some research efforts have proposed the automation of the data warehouse design in order to free this task of being (completely) performed by an expert, and facilitate the whole process. However, the more the process gets automated, the more the integration of requirements is overlooked on the way.

In our previous work [39] we addressed how to automatically validate requirements and conciliate them with the data sources, to support the data warehouse design task. This work fits in traditional scenarios in which the end-user requirements are known before hand (i.e., by the point we start the design task). However, this scenario does not always hold in data warehousing, and the task to elicit the end-user information requirements might result in a thorough task. Importantly, note that the analysis capabilities of the target data warehouse depend on what kind of data is available in the data sources. Thus, in those scenarios where the analysis capabilities of the data sources are not (fully) known, it is possible to help the data warehouse designer to identify and elicit unknown analysis capabilities. Eventually, these unknown capabilities may provide strategic advantages for the organization.

In this paper we introduce a *user-centered* approach to support the end-user requirements elicitation and the data warehouse multidimensional design tasks. It consists of three steps:

- First, our approach starts by fully analyzing the data sources to identify, without considering requirements yet, the multidimensional knowledge they capture (i.e., data likely to be analyzed from a multidimensional point of view).
- Next, we propose to exploit this knowledge in order to support the requirements elicitation task. In this way, we are already conciliating requirements with the data sources, and we are able to fully exploit the analysis capabilities of the sources.
- Finally, once requirements are clear, we automatically create the data warehouse conceptual schema according to them, and the multidimensional knowledge extracted from the sources.

Thus, we say it is a user-centered approach since the feedback of the user[1] is needed to filter and shape results obtained from analyzing the sources, and eventually produce the desired conceptual schema. In this scenario, our main contribution is the AMDO (*Automating Multidimensional Design from Ontologies*) method, our proposal for discovering the multidimensional knowledge contained in the data sources (i.e., corresponding to the first stage discussed above). Importantly, note that AMDO focuses on identifying the multidimensional knowledge contained in the sources regardless of the requirements (as discussed in Section 2, this kind of approaches are known as *supply-driven* approaches). Relevantly, current supply-driven approaches suffer from two major drawbacks, which we claim to overcome with AMDO.

The first one is that supply-driven approaches tend to generate too many results. Consequently, they unnecessarily overwhelm users with blindly generated combinations whose meaning has not been analyzed in advance. Eventually, they put the burden of (manually) analyzing and filtering results provided onto the designer's shoulder, but the time-consuming nature of this task can render it unfeasible when large data sources are considered.

Filtering the results provided by these approaches is a must, and AMDO aims at filtering the results obtained by means of objective evidences. Specifically, we introduce the concepts of *filtering function* and *searching patterns* (see Section 4 for further information), which filter and rank results obtained and eventually facilitate the analysis of AMDO's output.

The second drawback is that current supply-driven approaches mostly carry out the design task from relational OLTP (*On-Line Transaction Processing*) systems, assuming that a RDBMS is the most common kind of data sources we may find, and taking as starting point a relational schema (i.e., a logical schema). As a result, these approaches require a certain degree of normalization in the input logical schema to guarantee that it captures as much as possible the to-one relationships existing in the domain. As detailed in Section 2, discovering this kind of relationships is crucial in the design of the data warehouse, and the most common

[1] Note that we distinguish between the end-user (i.e., the users that will exploit the data warehouse once devised), and the users benefiting from our approach (i.e., the data warehouse designers).