UREM-A UML-based Requirement Engineering Model for Data Warehouse

Vishakha Gupta¹, Anupam Chauhan², Amit Kumar³ and Shweta Taneja⁴
¹²³ Bhagwan Parsuram Institute of Technology, Sector-17, Rohini, Delhi-110085
⁴vish0590@yahoo.com,²anup2436@gmail.com,³amit0205kk@gmail.com and
⁴shweta_taneja08@yahoo.co.in

ABSTRACT
Data Warehouse is a collection of data collected not only for storage purposes but for business analysis as well. For this the data that is to be stored must conform to some standards. As this data is to be used for business intelligence, it makes the design of data warehouse a difficult task. For any system, gathering the requirements forms the basis for design. The quality of the data warehouse depends on the quality of the requirements gathered. Hence a good amount of time and resources should be devoted to requirements engineering. For this various techniques have been proposed such as GORE, AORE, MORE, SCENARIO based etc. each handling requirements in a different way.

In this paper we have combined scenario based approach and UML (Unified Modeling Language), and proposed a model for requirement engineering of a data warehouse which will help in its design process.

KEYWORDS
Requirement Engineering, Scenarios, Data Warehouse, Unified Modeling Language.

1. INTRODUCTION
The term Data Warehouse was coined by Bill Inmon in 1990, in which he defined: “A warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management’s decision making process”. The term subject oriented means that Data gives information about a particular subject instead of about a company’s ongoing operations. The term Integrated means that data is gathered into the data warehouse from a variety of sources and merged into a coherent whole. The term Time-variant means that the data is identified with a particular time period. The term Non-volatile means that data is stable in a data warehouse. More data is added but data is never removed. This enables management to gain a consistent picture of the business [14].

A data warehouse is basically a pool of related data mainly required for business oriented decision making. It takes data from various operational systems and consolidates all of it into a single repository. But it has been seen that many number of data warehouses fail to meet business objectives. A major reason for this is the neglect of importance of requirement analysis.

Requirement engineering aims at defining a step by step process from identifying to documenting requirements which helps reduce failures of data warehouse. The processes involved in requirement engineering include eliciting, understanding, analyzing and documenting requirements [12, 13]. Broadly speaking, requirements engineering (RE) is the process of discovering the purpose, by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation.

We provide a way of integrating scenario based approach with object oriented mechanisms like UML. This can be used for Data Warehouse design within a mixed supply/demand-driven framework [8]. Scenarios provide a useful way of eliciting and validating requirements. Scenarios can be used to derive the requirements systematically using various different methods [6]. Also various different situations can be easily exemplified using scenarios which can further be used to make separate modules. This modular approach helps us to study the requirements in greater detail. These modules can be further used in other similar situations implying reusability. Scenarios play three different roles: (i) problem statement scenarios explain the unsatisfactory state of affairs with an existing system that the new system needs to resolve, (ii) visioning scenarios describe how the new system should operate, (iii) usage or behavioral scenarios are used as descriptions of the behavior of users and the current system [9].

A practical application of three selected scenario-based methods to software system development shows that scenarios can provide significant results [15]. Also, scenarios can be integrated with use case approach to object-oriented development [1].

Unified Modeling Language (UML) is a popular technique for documenting and modeling a system. The UML uses set of symbols to represent graphically the various components and relationships within the system and it can be used for business processing modeling and requirements modeling. It mainly is used to support object oriented system analysis and to develop the object models. It represents the system with structural and behavioral language models and those, designing modeling, include the architectural, mechanical and detailed design.

In this paper we propose a UML-based Requirement Engineering Model for Data Warehouse (UREM). This model utilizes the benefits of the scenario-based requirement engineering as well as of UML (Unified Modeling Language). The paper is organized as follows. In Section 2, the related work is discussed. In Section 3, we compare the various RE
approaches. In Section 4, we propose our model UREM. In Section 5, we make a conclusion. In Section 6, we discuss its future scope. Section 7 consists of the references.

2. RELATED WORK

There have been a number of researches conducted in requirements engineering unearth various approaches. These include GORE (Goal Oriented Requirements Engineering), AORE (Aspect Oriented Requirements Engineering), MORE (Model-based Object-oriented Requirements Engineering), Scenario-Based Requirements Engineering etc.

Goal-oriented requirements engineering is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analyzing, negotiating, documenting, and modifying requirements. Goals may be formulated at different levels of abstraction, ranging from high-level, strategic concerns. Goals also cover different types of concerns: functional concerns, associated with the services to be provided and non-functional concerns, associated with quality of service - such as safety, security, accuracy, performance, and so forth [7].

A Model-based OO requirement (MOOR) is defined as a data warehouse requirement signified by an object oriented representation with model support. Applying modeling and OO technologies to requirement phases, the domain knowledge can be captured in a well-defined model, so the completeness, consistency, traceability and reusability of requirement and its integration with the artifacts of other phases can be cost effectively improved [5].

Requirements engineering techniques that explicitly recognize the importance of clearly identifying and treating crosscutting concerns are called Aspect-oriented Requirements Engineering Approaches (AORE approaches). Aspect-oriented requirements engineering approaches improve existing requirements engineering approaches through an explicit representation (and modularization) of concerns that were otherwise spread throughout other requirements artifacts (such as use cases, goal models, viewpoints, etc.) [3].

3. COMPARISON WITH OTHER R.E. APPROACHES

Various requirements engineering techniques have been used to elicit, validate and document requirements each having its own set of positive and negative aspects. Here we have tried to compare some of them.

3.1 GOAL ORIENTED REQUIREMENT ENGINEERING

Advantages

Its ability to specify and manage positive and negative interactions between goals allows the analyst to reason about design alternatives [10]. Its capability to trace low-level details back to high-level concerns is very appropriate to bridge the gap between architectural models and requirements [11]. It has facilities for elaboration and deeper analysis of requirements specification [16]. Alternative system proposals could be proposed with the help of alternative goal refinement [4].

Disadvantages

It concentrates on the goals that are being aimed for in doing a project, rather than looking at the concrete requirements of the data warehouse. Goals are not necessarily measurable (i.e. verifiable) and therein may differ from requirements. If not analyzed properly they may lead to an abstract model. It is not even possible to transform all the goals into requirements. In complex systems, requirements specifications suffer from crosscutting, which affects elaboration, readability and maintainability of the specification, even when using a Goal-Oriented approach [16].

3.2 MODEL-BASED OBJECT-ORIENTED REQUIREMENT ENGINEERING

Advantages

Being object oriented, the system can be easily modeled using this approach.

It has the benefits of higher maintainability, reusability, and productivity. MORE provides the mechanism to transform the existed requirement documents in text form in natural language to MOORMs.

The domain knowledge can be properly kept and then effectively reused. It develops a uniform requirement models defined by MOORMs as opposed to the traditional approach with loose information scattered in different documents [5].

Disadvantages

It mainly focuses on documenting the requirements rather than the entire requirement engineering process.

3.3 ASPECT ORIENTED REQUIREMENT ENGINEERING

Advantages

It identifies and manages the crosscutting in an elegant and effective way, based on separation of concerns [16]. It helps in tackling tangled requirement representations that are difficult to understand and maintain. It also ensures consistency of stakeholders’ concerns with global requirements and constraints [2].

Disadvantages

A major problem comes in defining aspects. Aspects may interfere with one another. The correctness of an aspect may be difficult to ascertain or to prove. Also, the intent of an aspect may not be clear, since it is stated operationally.

3.4 SCENARIO BASED REQUIREMENT ENGINEERING

Advantages

A scenario based design method is very useful to determine what functionalities are required for the system. Scenarios can be used to systematically derive all the requirements. Varied situations can be easily exemplified using scenarios. They can
also help to reason out specific details. One can find out any discrepancies in the requirement specification phase itself. Also, implementation of the data warehouse becomes easier due to the clear structure of scenarios. Majorly scenarios provide extensive reuse of requirement knowledge.

We have proposed a model, UREM, based on scenario-based requirements engineering approach, taking into consideration all its above mentioned benefits. Scenarios can be used to elicit as well as validate requirements effectively. Also, using the modular approach provides us the facility of reusing scenarios, in addition to being easily manageable and traceable.

Considering the ease of modeling provided by the object oriented methodologies we have used UML to complement scenarios. UML simplifies the diagrammatic representations of the system leading to a well defined documentation of the entire set of requirements. Following is the detailed explanation of our proposed model.

4. THE PROPOSED UREM MODEL

Using the advantages of the scenario based requirement engineering and UML; we have proposed a new model called UREM. This model firstly depicts the generalized requirements for the data warehouse. It then categorizes requirements based on scenarios. Pertaining to each scenario various modules are defined. These are then aggregated to get the refined requirements. The requirements are then subjected to various constraints and also extended relationships are identified. Finally, a UML document is prepared which encompasses all the aspects of the requirements.

The model consists of five levels (fig 1):

4.1 SCENARIO LEVEL

We have proposed the use of scenario based requirement engineering to identify the different scenarios. This level decomposes the generalized requirements into scenarios which can be further modularized. We feel that scenarios can be reused particularly in applications pertaining to a common problem domain. Each scenario defines one situation and functionalities related to it.

4.2 MODULE LEVEL

This level consists of making of modules based on the scenarios identified in the previous level. These modules can be classified into basic and composite. Basic module pertains to one function. The composite modules are composed of several basic modules. Each basic module can participate in one or more composite modules. Thus the model proposes modules with different levels of granularity.

All these modules are together used to find the refined set of requirements. These are the processed requirements on which the data warehouse has to be designed. These modules have the ability to grow as needed and help in reusability. Also, these can be independently used to define requirements accurately. Each module will have the capability to be restructured independently.

4.3 CONSTRAINT LEVEL

This level defines constraints on the attributes and their values. Constraints provide a mechanism for ensuring that data conforms to specified guidelines. This helps in ensuring that data in the data warehouse conforms to basic level of data consistency and correctness. Besides, constraints may be useful in various aspects of query optimization.

Constraints are properties for specifying semantics and/or conditions that must be held true at all times for the elements of a model. They can be added by extending the semantics of a UML building block by adding new rules, or modifying existing ones.
4.4 EXTENSIBILITY LEVEL
In this level we identify the various extended features of UML like generalization, specialization, aggregation etc. and incorporate them in our proposed model. Generalization relationships denote inheritance between classes. It simplifies the model by eliminating redundancy. It denotes the bottom up approach whereas specialization denotes the top down approach. An aggregation is a special type of association in which objects are assembled or configured together to create a more complex object. An aggregation describes a group of objects and how you interact with them.

4.5 DOCUMENT LEVEL
This level comprises the documentation of the various requirements along with their constraints and extensions. This is done using UML (Unified Modeling Language). By modeling the requirement document in UML, the requirements can be kept in a way in which they can be analyzed and integrated with artifacts in further development of the data warehouse. This will help in maintaining traceability and consistency.

5. CONCLUSION
In this paper we concentrated on the aspect of data warehouse design using requirements engineering. We carried out a formal scenario-based approach to model data warehouse requirements by introducing the UREM model. We have used UML for documenting warehouse requirements. In doing so, we have highlighted the benefits of using such an approach.

6. FUTURE SCOPE
For future work, many issues deserve investigation like to make a plan for validating both our newly proposed requirement model and already existing models. Also, we need to investigate the available technologies to automate the mapping between the scenarios and UML documents. The performance issues associated with such a demanding task require examination.

REFERENCES
[3]. A. Yakout A. Mohamed, Prof Dr. A. E. Fatah. A. Hegazy, Dr. A. R. Dawood “Aspect Oriented Requirements Engineering” Vol. 3, No. 4; November 2010, ISSN 1913-8989 E-ISSN 1913-8997.
[4]. N. Rehman, S. Bibi, S. Asghar, S. Fong “Comparative Study of Goal-Oriented Requirements Engineering”.
[9]. S. Misra, V. Kumar, U. Kumar “Goal–oriented or Scenario-based Requirements Engineering Technique—What should a practitioner select?” IEEE, CCECE/CCGEI, Saskatoon, May 2005, 7803-8886-0/05.