RC 22042 (98936) 24 April 2001 Computer Science

IBM Research Report

Developing Requirements for Data Warehouse Systems with Use Cases

Robert Bruckner, Beate List
Vienna University of Technology

Josef Schiefer
IBM Research Division
Thomas J. Watson Research Center
P. O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - T. J. Watson - Tokyo - Zurich

Developing Requirements for Data Warehouse Systems with Use Cases

Robert M. Bruckner[1], Beate List[1], Josef Schiefer[2]

[1]Vienna University of Technology
{bruckner, list}@ifs.tuwien.ac.at

[2]IBM Watson Research Center
josef.schiefer@us.ibm.com

Abstract

Intelligent and comprehensive data warehouse systems are a powerful instrument for organizations to analyze their business. The implementation of such decision systems for an enterprise-wide management and decision support can be very different from traditional software implementations. Because data warehouse systems are strongly data-driven, the development process is highly dependent on its underlying data, which is generally stored in a data warehouse. Since data warehouse systems concern many organizational units, the collection of unambiguous, complete, verifiable, consistent and usable requirements can be a very difficult task. Use cases are considered as standard notation for object-oriented requirement modeling. In this paper we show how use cases can enhance communication between stakeholders, domain experts, data warehouse designers and other professionals with diverse backgrounds. We introduce and discuss three different abstraction levels (business, user and system requirements) of data warehouse requirements and show how use cases can be drivers for the requirements development.

Keywords: Data Warehousing, Requirements Engineering, Use Case Modeling

1. Introduction

Building a data warehouse is a very challenging task because it can often involve many organizational units of a company. A data warehouse is a common queryable source of data for analysis purposes, which is primarily used as support for decision processes. Furthermore, it is multidimensional modeled and is used for the storage of historicized, cleansed, validated, synthesized, operative, internal and external data. Stakeholders of a data warehouse system are interested in analyzing their business processes in a comprehensive and flexible way. Mostly they already have a comprehensive understanding of their business processes, which they want to explore and analyze.

What they actually need is a view of their business processes and its data, which allows them an extensive analysis of their data. For this purpose data warehouses are modeled multidimensional, which corresponds to a typical view of its users. This analysis view of the business processes can be very different to the general view even though the underlying process is the same. Hence it is necessary to elicit requirements from the stakeholder of a data warehouse, which belong to their analysis views. The design of data warehouse system is highly dependent on these requirements. Very often data warehouses are built without understanding correctly these needs and requirements and consequently fail for that reason.

During the requirement definition process system analysts of the IT department or consultants work together with stakeholder and users to describe the requirements for the data warehouse system. The data warehousing team receives these descriptions, but they have often trouble understanding the business terminology and find the description too informal to use for the implementation. Therefore the data warehousing team writes its system specification from a technical point of view. When the system specification is presented to the users, they do not

quite understand because it is too technical. They are, however, forced to accept it in order to move forward.

This approach can easily result in a data warehouse system that does not meet the initially defined requirements because often the users, the system analysts and developers don't speak the same language. Such communication problems can make it difficult to turn a description of an analysis system into a technical specification of a data warehouse system that all parties can understand. In addition, because of a technical system specification that is not fully understood by the users, a data warehouse system becomes too difficult or impractical for the intended purposes. Therefore, it will not deliver the expected effect to the company. In these cases often departments will develop data marts for their own purposes, which can be considered as stovepipes and makes an enterprise-wide analysis system impossible.

The challenge is to model a data warehouse system in a way that is both precise and user-friendly. Each symbol describing the analysis process should be intuitive for the user and have defined semantics, so that the developers can use the description as a general, but precise specification of the data warehouse system.

In this paper, we use use cases for getting a comprehensive, intuitive specification for the data warehouse system, which is satisfying and understandable for all involved parties. Use cases have two advantages, which make them suitable for representing the requirements for a data warehouse system. First, use case diagrams are part of the UML standard and hence are generally accepted as a notational standard in the software community and second, they can be used on a general level, where implementation details are completely suppressed.

In the following sections we show a use case driven approach for the development of data warehouse requirements. The remainder of this paper is organized as follows. In section 2, we discuss the contribution of this paper and related work. In section 3, we discuss the abstraction levels of data warehouse requirements. In section 4, we argue why to use use cases for capturing data warehouse requirements. In section 5, we introduced iterations for an incremental and iterative use case development. Finally, in section 6 we present our conclusion.

2. Contribution and Related Work

Building a data warehouse is different from developing transaction systems, whereby the requirement analysis process for the latter is supported by numerous methods (Coad, Yourdon 1991), (Davenport 1993), (Finkelstein 1996), (Partsch 1998). Up to now the data warehouse design process has not been supported by a formal requirement analysis method although there are some approaches for requirement gathering.

Inmon (1996) argues that the data warehouse environment is data driven, in comparison to classical systems, which are requirement driven, and the requirements are understood after it is populated with data and being used by the decision support analyst. He derives the data model by transferring the corporate data model into a data warehouse schema and by adding performance factors.

Anahory and Murray (1997) propose a catalogue for conducting user interviews in order to collect end user requirements. They state that a data warehouse is designed to support the business process rather than specific query requirements, but do not further discuss their statement.

Another process driven approach is applied by Kimball (1996 and 1998), whereby the fundamental step of the design process is based on choosing a business process to model. As this approach has proven its success in various projects, and as enterprises in general have shifted to process-centered organizing, we adopt the process-oriented approach for the basis of our work: a formal requirement analysis concept for data warehousing.

In this paper we present three different abstraction levels of data warehouse requirements to address better all parties, which are involved in the requirements development process. In order to enhance the communication between stakeholders, end users, and the data warehouse development team we focus on modeling the requirements with use cases and derive the detailed system requirements from them instead of representing the requirements just in specification templates.

3. Data Warehouse Requirements

Requirements of an enterprise-wide data warehouse system determine its functional behavior and its available information, for example what data must be accessible, how it is transformed and organized, as well as how it is aggregated or calculated. The requirements enable the stakeholders to communicate the purpose, establish the direction and set the expectations of information goals for the enterprise. Stakeholders often express their needs in general expectations of the data warehouse system to improve their business. This business view describes the goals and expectation of stakeholders, which is the foundation of the data warehouse requirements. On the other hand, the development team of a data warehouse system expects a complete, correct and unambiguous specification of the system it has to build, which means a further refinement of the business requirements from the stakeholders. Therefore, it is necessary to transform the business requirements to a detailed, testable, and complete specification for the data warehouse team (Wiegers 1999).

For that reason, data warehouse requirements have different abstraction levels. They include the following levels (see Figure 1):

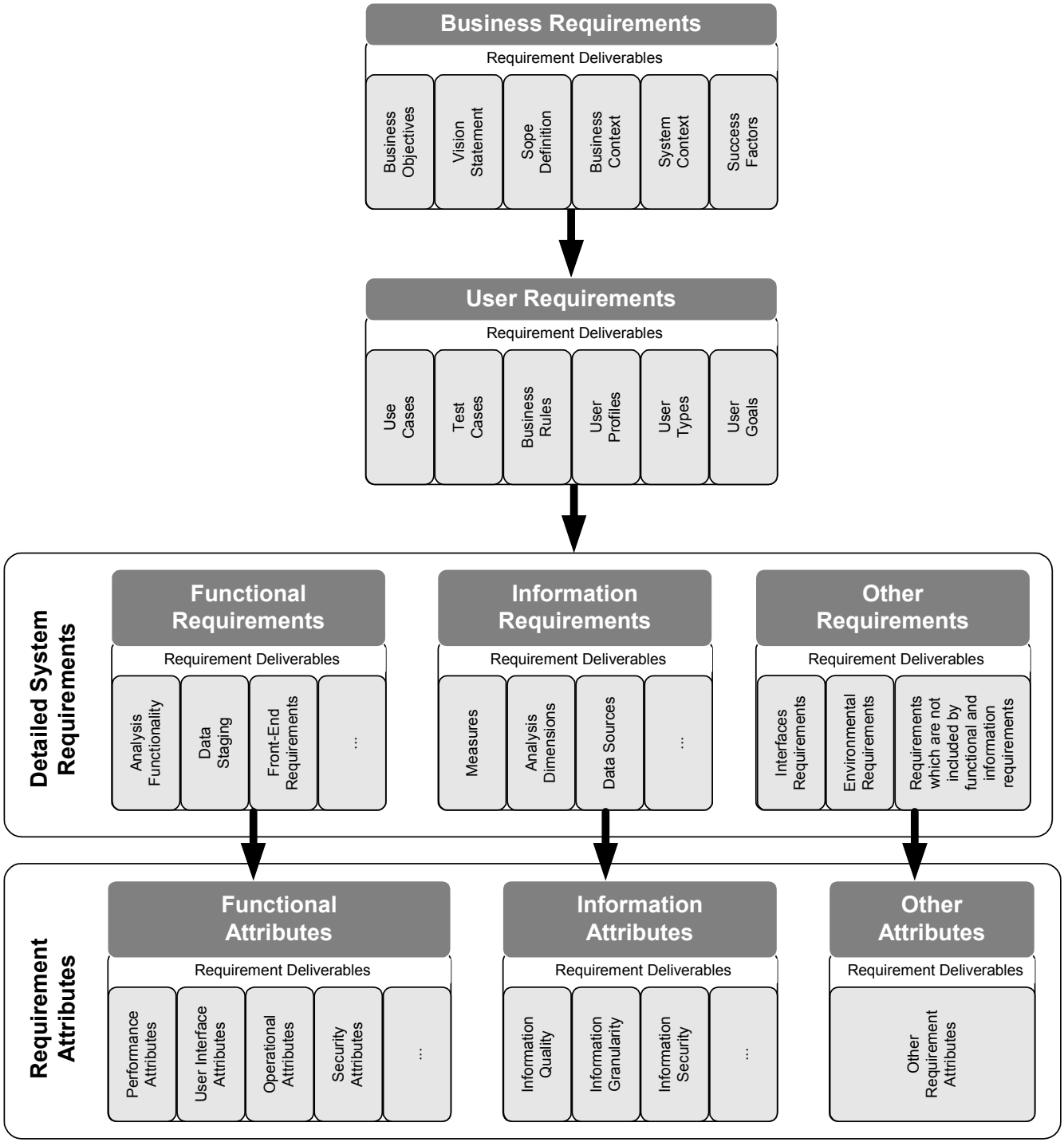

Figure 1: Abstraction Levels of Data Warehouse Requirements

Business Requirements

Business requirements represent high-level objectives of the organization for the data warehouse system. They are captured in a document describing the project's vision and scope. Further services of the data warehouse system are derived from the business requirements, which are represented in a system context diagram. The business requirements identify the primary benefits that the data warehouse system will provide to the organization and its users. They represent the top level of abstraction in the requirements chain. They express business opportunities, business objectives and describe the typical users and organizations requirements and their provided value of the system at a high level.

User Requirements

User requirements describe the tasks that the users must be able to accomplish with the help of the data warehouse system. User requirements must be collected from people who will actually use and work with the data warehouse system. Therefore, these users can describe both the tasks they need to perform with the data warehouse system and the nonfunctional characteristics that are important for the data warehouse to be well accepted. The user requirements must align with the context and objectives established by the business requirements. They are captured in use cases or scenario descriptions; they focus on what the users need to do with the data warehouse system and are therefore much more powerful than the traditional requirements elicitation approach of asking users what they want the system to do.

Detailed System Requirements

They represent the data warehouse system requirements on a very detailed level. The high level of detail facilitates the complete, fine-grained specification of the requirements, which are an important input for the development team. They must align with the user and business requirements and contain a fit criterion, which allows requirement verification.

Functional requirements define the functionality that the development team must build into the data warehouse system to enable users to accomplish their tasks, thereby satisfying the business requirements. Functional requirements capture the intended behavior of the data warehouse system. This behavior may be expressed as services, tasks or functions the system is required to perform. They describe what the analysis system must do – an action that the system must take if it is to provide useful functionality for its users.

Information requirements define the information needs of the company. They describe the information and data, which the data warehouse should deliver or should have access to. They specify the provided data, by what quality it should have, where it comes from, how it should be processed, how it should be combined for the analysis process and which analysis methods will be used.

Other requirements besides functional and information requirements can be specified, to describe further relevant aspects of the data warehouse system, like interface requirements or environmental (cultural, political, legal) requirements.

Requirement Attributes

They augment the description of the functional, information or other requirements by describing the characteristics in various dimensions that are important either to users or to the data warehouse team. Requirement attributes are properties, or qualities that the data warehouse system should have. These might include standards, regulations, and conditions to which the data warehouse system must conform; description of external interfaces; performance requirements, design and implementation constraints; and quality attributes. Requirement attributes are usually attached to the detailed system service requirements. For instance, when the functional requirements are known, it can be determined how they are to behave, what qualities they are to have, and how big or fast they should be. When the information requirements are known, its attributes can be determined, like data quality or data granularity.

Functional vs. Information Requirements

Traditional software requirements distinguish between two types of requirements: 1) functional requirements and 2) non-functional requirements. This separation of requirements can be applied also to data warehouse systems. But the heart of data warehouse system is the data and information that is delivered. Information requirements are included in traditional software requirements in the functional and non-functional requirements.

Because data warehouses are built for delivering data, the requirements, which describe this data, are highly significant for the data warehouse design and more comprehensive than the requirements in traditional software systems. Therefore, traditional software requirements types are not very suitable for data warehouse projects. In order to address the aspect that data warehouse systems are information-centric, we explicitly distinguish between functional requirements and information requirements.

The separation of functional and information requirements can be very advantageous for the data modeler of a data warehouse. Because primarily only information requirements are relevant for the data model of a data warehouse, the data modeler isn't diverted by other requirements.

4. Requirements Gathering with Use Cases

For many years, system analysts have used scenarios to describe ways a user can interact with a software system to help elicit requirements. Ivar Jacobson (1992 and 1995) and others formalized this into the use case approach to requirements elicitation and modeling. Although use cases emerged from the object-oriented development world, they can be applied to projects that follow any development approach, because the user does not care how to build the system. Therefore, the shift in perspective and thought processes that use cases bring to requirements development is more important than drawing formal use case diagrams. The focus on what the *users* need to do with the system is much more powerful than other traditional elicitation approaches of asking users what they want the *system* to do.

Use Cases and Detailed System Requirements

The use case descriptions often do not provide the data warehouse team with enough detail about the functionality they must build and about the information that must be made available. Gathering all requirements in use case descriptions can result in cumbersome and complex use cases. On the other hand, if stopping requirements development at the user requirements stage, at construction time the data warehouse team will have many questions to fill their information gaps. To reduce this uncertainty, each use case needs to be elaborated into its detailed system requirements (Arlow 1998). For very small use cases, the use case description can be sufficient and the elaboration of the detailed system requirements can be skipped.

Each use case leads to a number of detailed system requirements that will enable a data warehouse user to perform their pertinent task, whereby several use cases might need the same detailed system requirement. For example, if three use cases require the same data with the same quality, the specification of the data should not be written down in every use case. The specification of the data and its quality can be described by detailed system requirements, which can be associated with a use case in several ways.

The approach that is taken depends on whether the data warehouse team should perform the design, construction and testing from use case documents, from detailed system requirements, or from a combination of both. None of these methods is perfect and it has its advantages and disadvantages. By selecting an appropriate approach, it is important to avoid any duplicated information in multiple locations, as redundancy makes requirements management more difficult.

Use Case Refinement

Use case refinement activities ensure that requirements are accurate, complete, and demonstrate the desired quality characteristics. Data warehouse requirements that seem fine when they are captured by use cases might turn out to have problems when the data warehouse team works with them. If for instance test cases for use cases are defined, possible ambiguities and vagueness in some of the use cases may be found. They must be removed if the data warehouse requirements are to serve as a reliable foundation for the design and implementation. These use case refinement activities include activities to ensure that:

- The use cases describe the intended system behavior and characteristics of the data warehouse system.
- The use cases were correctly derived from the business requirements or other origins.
- The use cases are complete and of high quality.
- All views of the use cases are consistent.
- The use cases provide an adequate basis to proceed with the design and implementation of the data warehouse system.

5. Iterative and Incremental Use Case Development

For the development of use cases, we suggest an iterative and incremental approach. An incremental approach to requirements specification includes completing batches of use cases together, but not assuming that all artifacts must be completed at the same time. An iterative approach to requirements specification includes the refinement of use cases through a number of iterations.

Iterative steps in gathering requirements with use cases are highly dependent on individual situations. We can't make a suggestion about the exact number of iterations that are needed to complete the requirements in all situations. However, it is possible to say that iterative requirements specification always proceeds through the same logical steps in every situation. Kulak and Guiney (2000) introduced an approach for an iterative use case development. We extended this approach to be more suitable for data warehouse environments. We suggest following four logical steps for the use case development:

Table 1: Use case Iterations

Iteration	Description
Facade	Outline and high-level descriptions
Filled	Broadening and deepening
Analyzed	Evaluation, narrowing and pruning
Optimized	Reconciliation, reassessment, enhancement
Finished	Touching up and fine-tuning

Facade Iteration

The Facade Iteration is the first iteration in the requirements lifecycle of use cases. Its purpose is to create

placeholders for each major interaction of a user with the data warehouse system. A Facade use case contains only the minimum information as a placeholder is gathered, which includes names and a short description of each user interaction with the data warehouse system. It also identifies the major actors of the system. Defining use cases for this iteration is difficult, because you don't have any concept of the data warehouse system yet. For this reason, the data warehouse team and the stakeholders will do their best work if the environment encourages openness and creativity.

Filled Iteration

The objective of the Filled Iteration is to sharpen the ideas of the use cases from the Facade Iteration and to create a comprehensive set of use cases and business rules that describe the data warehouse system. As you delve into more detail during this iteration, it is possible to find Facade use cases which are too general and need to be broken down into several Filled use cases. During this iteration, most of the time is spent to understand the requirements from the stakeholders, and exploring possible solutions for the data warehouse system. Although these requirements are rough and unpolished, they contain details and include functional and information requirements as well requirement attributes.

Analyzed Iteration

The requirements from the previous iterations can be discouragingly large. Therefore, in the Analyzed Iteration only the best options are selected. The Analyzed Iteration clears a path through the existing requirements and leaves clear project requirements. It separates the essential from the nice-to-have. At the end of this iteration, there is sufficient information and material to build a successful data warehouse system.

Use case analysis involves refining, evaluating, and scrutinizing the gathered requirements to make sure all stakeholders understand what they mean and to find error, omissions, or other deficiencies. The goal is to develop use cases of sufficient quality and detail, so that they can be used to construct realistic project estimates and to proceed with the design, construction, and testing of the data warehouse system. Each use case is reviewed to make sure that the descriptions are complete and provide sufficient information without being too detailed or vague. This will be achieved by continual formalizing the use cases by specifying priorities, traceability links, fit criteria, pre- and postconditions, triggers, and inputs/outputs of the use case etc. Furthermore, in this iteration use cases are inspected, whether there are possible improvements or optimizations of the event course.

Optimized Iteration

The objective of the Optimized Iteration is an inspection for conflicting, inconsistent or missing use cases. The difference between this iteration and the previous iteration is, that in the Analyzed Iteration, each use case is checked independently, whereas the this iteration considers all of the use cases and requirements and their effect on each other. The input of this iteration are all use cases, whereas the input of the Analyzed Iteration are the individual use cases.

A further goal of this iteration is a new assessment of the complete specification of the data warehouse system:

- Remeasurement of the effort required building the data warehouse system. The initially estimated budget, schedule, and staff for building the data warehouse system, can be adjusted according to the final requirements specification.
- Reanalyzing the risks involved in completing the data warehouse project. Due to the earlier requirements elicitation and analysis activities and thus a better understanding of the functionality and impacts of the data warehouse system for the business, the risks for the data warehouse project can be assessed more accurately.
- Reassessment of the go/no-go criteria, which where defined in the project blast-off

Finished Iteration

In the Finished Iteration the stakeholders and the requirements analysts decide if the use cases are correct and complete to go on with the design or implementation of the data warehouse system. The objective of this iteration is to make reasonable decisions to craft requirement attributes, which are important for the data warehouse design and implementation. Finished use cases are the primary input to the design phase of the data warehouse system. They influence the design of a data warehouse system by communicating the scope and comprehensive descriptions of what the users want from the data warehouse system.

6. Conclusion

The use case model is an excellent means of both expressing user analysis activities and providing a comprehensive picture of what the data warehouse is intended to perform. In this paper we presented a use case driven approach for the development of data warehouse requirements. We specified data warehouse requirements by different abstraction levels, which allow a straightforward development of the requirement specification. Because data warehouses are information-centric, we stated, that there is a need to separate information requirements as individual requirement type. We discussed the iterations for a use case development, which facilitate the modelling of user requirements for data warehouse systems.

References

Anahory, S. Murray, D., Data Warehousing in the real World – A practical Guide for Building Decision Support Systems, Addison-Wesley Publishing 1997.

Arlow, Jim, *Use Cases, UML Visual Modeling and the Trivialisation of Business Requirements,* Requirements Engineering

Coad, P., Yourdon, E., *Object-Oriented Analysis. Englewood Cliffs*, N.J., Prentice Hall 1991.

Davenport, Th., *Process Innovation: Reengineering Work through Information Technology*, Harvard Business School Press 1993.

Finkelstein, A., *Requirements Engineering Research: Coordination and Infrastructure*, Requirements Engineering 1, 1996.

Inmon, W. H., *Building the Data Warehouse*, Wiley & Sons 1996.

Jacobson, I., Christerson, M., Jonsson, P., Overgaard, G., *Object-Oriented – Software Engineering – A Use Case Driven Approach*, Addison Wesley 1992.

Jacobson, I., Ericson, M., Jacobson, A., *The Object Advantage – Business Process Reengineering with Object Technology*, ACM Press, Addison-Wesley Publishing 1995.

Kimball, R., *The Data Warehouse Toolkit: Practical Techniques For Building Dimensional Data Warehouse*, John Wiley & Sons 1996.

Kimball, R., Reeves, L., Ross, M., Thornthwaite, W., *The Data Warehouse Lifecycle Toolkit*, John Wiley & Sons, 1998.

Kulak, D.; Guiney E., *Use Cases – Requirements in Context*, Addison Wesley, New York, 2000.

Partsch, H., *Requirements Engineering (in German)*, Springer 1998.

Wiegers, Karl, E., *Software Requirements*, Microsoft Press, 1999.