# Data warehousing with IBM cloud db2 warehouse


## ABSTRACT

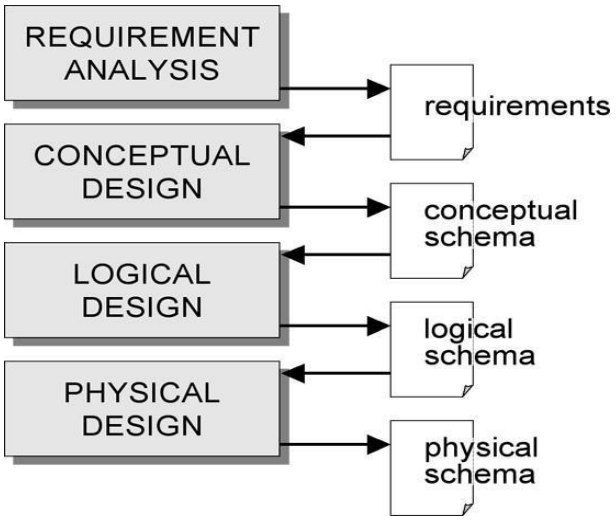
Multidimensional modeling requires specialized design techniques. Though a lot has been written about how a data warehouse should be designed, there is no consensus on a design method yet. This paper follows from a wide discussion that took place in Dagstuhl, during the Perspectives Workshop “Data Warehousing at the Crossroads”, and is aimed at outlining some open issues in modeling and design of data warehouses. More precisely, issues regarding conceptual models, logical models, methods for design, interoperability, and design for new architectures and applications are considered.


## 1. INTRODUCTION

It is well known that data warehouses (DWs) are focused on decision support rather than on transaction support, and that they are prevalently characterized by an OLAP workload. Traditionally, OLAP applications are based on multidimensional modeling, that intuitively represents data under the metaphor of a cube whose cells store events that occurred in the business domain. Adopting the multidimensional model for DWs has a two-fold benefit. On the onehand, it is close to the way of thinking of data analyzers and, therefore, it helps users understand data; on the other hand, it supports performance improvement as its simple structure allows designers to predict users’ intentions.

Multidimensional modeling and non-OLTP workloads require specialized design techniques. The most cited difference between design for transactional databases and DWs is denormalization, yet DW design has several other relevant peculiarities. Though a lot has been written about how a DW should be designed, there is no consensus on a design method yet. Most methods agree on the opportunity for distinguishing between a phase of *conceptual design* and one of *logical design*, like in Conceptual design aims at deriving an implementation-independent and expressive conceptual schema for the DW, according to the chosen conceptual model, starting from the user requirements and from the structure of the source databases. Logical design takes the conceptual schema and creates a corresponding logical schema on the chosen platform by considering some set of constraints (e.g., concerning disk space or query answering time). Several method also support a phase of *physical design*, that addresses all the issues specifically related to the suite of tools chosen for implementation – such as indexing and allocation. In some cases, a phase of *requirement analysis* is separately considered. From the functional point of view, the relationships between these phases can be summarized as in Figure 1 (in practice, this process will likely include feedback loops that allow to re-enter previous phases). Unfortunately, though most vendors of DW technology propose their own CASE solutions (that very often are just wizards capable of supporting the designer during the most tedious and repetitive phases of design), the only tools that currently promise to effectively automate some phases of design are just research prototypes this paper arises as an aterthought following a wide discussion that took place in Dagstuhl, during the Perspectives Workshop “Data Warehousing at the Crossroads” (August 2004). While the aim of the seminar was to discuss the current trends in data warehousing and to pave the way for future research in the whole field, here we will specifically focus on modeling and design, trying to answer the following

question: “Has research on this topic come to an end? If not, what’s left to do?” Thus, in this paper, benefiting from the fruitful discussions that took place there between




**The core phases in DW design**

all participants, we survey some topics related to DW modeling and design and outline the issues that, in our view, still need further exploration. More precisely, in Sections 2, 3, 4, 5, and 6 we address, respectively, conceptual models, logical models, methods for design, interoperability, and design for new architectures and applications.

## 2. CONCEPTUAL MODELING

Conceptual modeling provides a high level of abstraction in describing the warehousing process and architecture in all its aspects, aimed at achieving independence of implementation issues. Conceptual modeling is widely recognized to be the necessary foundation for building a database that is welldocumented and fully satisfies the user requirements; usually, it relies on a graphical notation that facilitates writing, understanding, and managing conceptual schemata by both designers and users.

In the literature, conceptual modeling for DWs has been tackled from mainly two points of view so far:

- *Multidimensional modeling*. The existing approaches may be framed into three categories: extensions to the Entity-Relationship model extensions to UML While all models have the same core expressivity, in that they all allow the basic concepts of the multidimensional model to be represented, they significantly differ as to the possibility of representing more advanced concepts such as irregular hierarchies, many-to-many associations, and additivity.
- *Modeling of ETL*. The focus is to model the ETL process either from the functional the dynamic or the static point of view. Though the research on ETL modeling is probably less mature than that on multidimensional modeling, we believe that it will have a very relevant impact on improving the overall reliability of the design process and on reducing its duration.

While apparently a lot of work has been done in the field of conceptual modeling, we believe that some very important issues still remain open, as detailed in the following.

### 2.1 LOGICAL MODELING

Once the conceptual modeling phase is completed, the overall task of logical modeling is the transformation of conceptual schemata into logical schemata that can be optimized for and implemented on a chosen target system.

Considerable progress has been made in the area of multidimensional modeling, where target database systems are typically either relational or multidimensional. In relational implementations, the so-called star, constellation, and snowflake schemata are widely accepted to manage data cubes and are supported by various vendors. Concerning multidimensional implementations, several efficient multidimensional data structures such as condensed cubes and QC-Trees have been proposed to manage data cubes.

Nevertheless, we believe that some relevant challenges remain for future research, as summarized in the following subsections.

### 2.2 Semantic gap

With respect to fact modeling, there still is a semantic gap between advanced conceptual data models and relational or multidimensional implementations of data cubes. For instance, no commercial solutions can cope with generalization/specialization relationships in OLAP hierarchies . Additionally, it appears to be an open problem how to represent dimension constraints or even less expressive context dependencies , both of which explain the existence of null values in dimensions in logical implementations and allow to reason about summarizability with respect to *sets* of attributes. Moreover, a systematic treatment of summarizability addressing general aggregate functions beyond SUM remains an open issues. Consequently, future research is necessary to bridge this semantic gap, to preserve all information captured by advanced conceptual multidimensional models in logical implementations. To this end, research could either investigate how to enrich meta-data for tool support in a systematic way or, more ideally, look for more expressive logical models while preserving good query performance. Clearly, without the support of more expressive logical models we cannot expect to achieve a streamlined design process that guarantees quality criteria (e.g., avoidance of inconsistent queries, control over null values, reduction of sparsity to be satisfied and seriously takes security into account.

### 2.3ETL modeling

The transformation of conceptual ETL schemata into logical ones as well as their optimization are not very well understood. Indeed, present first steps towards the modeling and optimization of ETL processes at the logical level appears to be the only design method that includes an algorithmic transformation of conceptual into logical models.

Moreover, research on DW self-maintainability and independence has shown how to set up DWs in such a way that the maintenance processes can be simplified and made more efficient by avoiding maintenance queries. However, a combination of these results with ETL modeling techniques is still missing.

## 3.METHODS FOR DESIGN

While in the subsections above we have discussed the problems related to conceptual and logical models, in this subsection we are concerned with the techniques for building conceptual and logical schemata according to such models, considering them in the context of a comprehensive design

framework that complies with good-design principles such as reusability, extendibility, and manageability.

Several techniques for automating single phases of DW design have been proposed in the literature (for instance, for conceptual design, for logical design, for physical design, for designing the ETL process). On the other hand, despite the basic role played by a wellstructured methodological framework in ensuring that the DW designed fully meets the user expectations, a very few *comprehensive* design methods have been devised so. Overall, we believe that some specific issues in design, discussed in the following subsections, have not been properly investigated yet. Besides, more generally, mechanisms should appear to coordinate all DW design phases allowing the analysis, control, and traceability of data and metadata along the project life-cycle. An interesting approach in this direction consists in applying the Model Driven Architecture in order to automate the interschema transformations from requirement analysis to implementation .

## 3.1 Requirements Analysis

Requirement analysis plays a key role within any software project to reduce the risk of failure. Nevertheless requirement analysis for DWs has not been given much attention so far, and it is often overlooked in DW projects mainly since (1) warehousing projects are long-term ones, in which most requirements cannot be stated from the beginning; and (2) requirements are poorly shared across organizations, unstable in time, and refer to information that must be derived from data sources .

The approaches to DW design are usually classified in two categories . *Data-driven* approaches design the DW starting from a detailed analysis of the data sources; user requirements impact on design by allowing the designer to select which chunks of data are relevant for decision making and by determining their structuring according to the multidimensional model . *Requirement-driven* approaches start from determining the information requirements of end users, and how to map these requirements onto the available data sources is investigated only *a posteriori* . Some other authors, like use some kind of mixture of these two approaches, and consider both ( availability of data and user requirements) at the same time, which appears to be a promising direction of research that is superior to isolated data-driven and requirement-driven approaches. Finally, a novel approach is based on the definition of a set of *design patterns*, so that, once the needed pattern is found, it just has to be adapted to the available data and user requirements.

Though the approaches devised are promising, we believe that some further work needs to be done in order to provide designers with more usable and effective techniques for collecting information needs and quality-of-service requirements, and for translating them into (at least domainspecific, ideally general) conceptual models based on a common vocabulary between IT staff and decision makers. Thus, how quality-of-service can drive the design of the DW should be deeply studied.

## 3.2 Schema evolution

As several mature implementations of data warehousing systems are fully operational within medium to large contexts, the continuous evolution of the application domains is bringing to the forefront the dynamic aspects related to describing how the information stored in the DW changes over time. As concerns changes in data values, a number of approaches have been devised, and some commercial systems allow to track changes and to effectively query cubes based on different temporal scenarios . Conversely, the problem of managing *changes on the schema level* (that may be demanded by changes

either in the business domain or in the user requirements or in the sources) has only partially been explored, and no dedicated commercial tools or restructuring methods are available to the designer yet.

The approaches to management of schema changes in DWs can be framed into two categories, namely *evolution* and *versioning* : while both categories support schema changes, only the latter keeps track of previous versions. If one is sure that previous schema information will never be useful again, schema evolution offers adequate functionality. Otherwise ( to guarantee consistent re-execution of old reports), schema versioning offers the strictly more powerful approach. Actually, in some versioning approaches, besides "real" versions determined by changes in the application domain, also "alternative" versions to be used for what-if analysis are considered . Overall, we believe that versioning is better suited to support the complex analysis requirements of DW users as well as the DW characteristic of non-volatility. Thus, the main research challenges in this field are to provide effective versioning and data migration mechanisms, capable of supporting flexible queries that span multiple versions.

Considering the complexity of the ETL procedures, another very relevant issue is to devise techniques for propagating changes occurred in the source schemata to the ETL process. The obvious benefit in achieving these goals will be to keep the DW in sync with the business requirements, thus avoiding its obsolescence.

## 3.3 Quality metrics

Due to the strategic importance of DWs, it is absolutely crucial to guarantee their quality from the early stages of a project. While some relevant work on the *quality of data* has been carried out there is still no agreement on the *quality of the design process* and its impact on decision making. The most significant approaches to measuring the design quality can be framed as follows:

- *At the conceptual level*. There have been preliminary attempts towards defining metrics that allow the intuitive notions of quality of conceptual schemata to be replaced with quantitative measures (such as the number of facts, the number of degenerated dimensions, the number of shared hierarchy levels, etc.), in order to reduce subjectivity in evaluation and guide designers in their work. Obviously, the existence of a standard conceptual model could give a strong push in this direction.
- *At the logical/physical level*. Besides the recommendations and subjective criteria stated for instance in , some works were focused on quantitatively evaluating the complexity of dimensional models. Other relevant research directions include normal forms for DW and quality-driven view selection .

Overall, we believe it is necessary to devise more comprehensive metrics for measuring quality, encompassing both schema quality ( to better model application requirements and to guarantee good querying performance) and data quality ( to ensure timeliness of information and to take care of data aging). After their formal and empirical validation, these metrics will support the designer in evaluating and ranking different design alternatives; besides, they will be useful to better plan the project and meet user requirements, by predicting the cost and complexity of later stages in design. Particular care should be taken in addressing the *traceability* of metrics, how metrics are translated from one phase of design to the next one, and in defining thresholds to discriminate "good" schemata from "bad" ones. Besides, techniques will be needed to *monitor* the metrics and appropriately respond to their deviations during the DW lifetime, in order to better manage extensions and evolutions. Finally, these metrics must be considered from the user point of view, by studying their impact on information analysis: methods must

be devised to propagate data quality metrics to query results, like in , and to have data retrieval driven by the quality requirements expressed by users, like in .

# 4.METHODS FOR DESIGN

While in the subsections above we have discussed the problems related to conceptual and logical models, in this subsection we are concerned with the techniques for building conceptual and logical schemata according to such models, considering them in the context of a comprehensive design framework that complies with good-design principles such as reusability, extendibility, and manageability.

Several techniques for automating single phases of DW design have been proposed in the literature (for instance, for conceptual design, for logical design, for physical design, for designing the ETL process). On the other hand, despite the basic role played by a wellstructured methodological framework in ensuring that the DW designed fully meets the user expectations, a very few *comprehensive* design methods have been devised so. Overall, we believe that some specific issues in design, discussed in the following subsections, have not been properly investigated yet. Besides, more generally, mechanisms should appear to coordinate all DW design phases allowing the analysis, control, and traceability of data and metadata along the project life-cycle. An interesting approach in this direction consists in applying the Model Driven Architecture in order to automate the interschema transformations from requirement analysis to implementation .

## 4.1 Requirements Analysis

Requirement analysis plays a key role within any software project to reduce the risk of failure. Nevertheless requirement analysis for DWs has not been given much attention so far, and it is often overlooked in DW projects mainly since (1) warehousing projects are long-term ones, in which most requirements cannot be stated from the beginning; and (2) requirements are poorly shared across organizations, unstable in time, and refer to information that must be derived from data sources .

The approaches to DW design are usually classified in two categories . *Data-driven* approaches design the DW starting from a detailed analysis of the data sources; user requirements impact on design by allowing the designer to select which chunks of data are relevant for decision making and by determining their structuring according to the multidimensional model . *Requirement-driven* approaches start from determining the information requirements of end users, and how to map these requirements onto the available data sources is investigated only *a posteriori* . Some other authors, like use some kind of mixture of these two approaches, and consider both ( availability of data and user requirements) at the same time, which appears to be a promising direction of research that is superior to isolated data-driven and requirement-driven approaches. Finally, a novel approach is based on the definition of a set of *design patterns*, so that, once the needed pattern is found, it just has to be adapted to the available data and user requirements.

Though the approaches devised are promising, we believe that some further work needs to be done in order to provide designers with more usable and effective techniques for collecting information needs and quality-of-service requirements, and for translating them into (at least domainspecific, ideally general) conceptual models based on a common vocabulary between IT staff and decision makers. Thus, how quality-of-service can drive the design of the DW should be deeply studied.

## 4.2 Schema evolution

As several mature implementations of data warehousing systems are fully operational within medium to large contexts, the continuous evolution of the application domains is bringing to the forefront the dynamic aspects related to describing how the information stored in the DW changes over time. As concerns changes in data values, a number of approaches have been devised, and some commercial systems allow to track changes and to effectively query cubes based on different temporal scenarios . Conversely, the problem of managing *changes on the schema level* (that may be demanded by changes either in the business domain or in the user requirements or in the sources) has only partially been explored, and no dedicated commercial tools or restructuring methods are available to the designer yet.

The approaches to management of schema changes in DWs can be framed into two categories, namely *evolution* and *versioning* : while both categories support schema changes, only the latter keeps track of previous versions. If one is sure that previous schema information will never be useful again, schema evolution offers adequate functionality. Otherwise ( to guarantee consistent re-execution of old reports), schema versioning offers the strictly more powerful approach. Actually, in some versioning approaches, besides "real" versions determined by changes in the application domain, also "alternative" versions to be used for what-if analysis are considered . Overall, we believe that versioning is better suited to support the complex analysis requirements of DW users as well as the DW characteristic of non-volatility. Thus, the main research challenges in this field are to provide effective versioning and data migration mechanisms, capable of supporting flexible queries that span multiple versions.

Considering the complexity of the ETL procedures, another very relevant issue is to devise techniques for propagating changes occurred in the source schemata to the ETL process. The obvious benefit in achieving these goals will be to keep the DW in sync with the business requirements, thus avoiding its obsolescence.

## 4.3 Quality metrics

Due to the strategic importance of DWs, it is absolutely crucial to guarantee their quality from the early stages of a project. While some relevant work on the *quality of data* has been carried out there is still no agreement on the *quality of the design process* and its impact on decision making. The most significant approaches to measuring the design quality can be framed as follows:

- *At the conceptual level*. There have been preliminary attempts towards defining metrics that allow the intuitive notions of quality of conceptual schemata to be replaced with quantitative measures (such as the number of facts, the number of degenerated dimensions, the number of shared hierarchy levels, etc.), in order to reduce subjectivity in evaluation and guide designers in their work. Obviously, the existence of a standard conceptual model could give a strong push in this direction.
- *At the logical/physical level*. Besides the recommendations and subjective criteria stated for instance in , some works were focused on quantitatively evaluating the complexity of dimensional models. Other relevant research directions include normal forms for DW and quality-driven view selection .

Overall, we believe it is necessary to devise more comprehensive metrics for measuring quality, encompassing both schema quality ( to better model application requirements and to guarantee good querying performance) and data quality ( to ensure timeliness of information and to take care of data aging). After their formal and empirical validation, these metrics will support the designer in evaluating

and ranking different design alternatives; besides, they will be useful to better plan the project and meet user requirements, by predicting the cost and complexity of later stages in design. Particular care should be taken in addressing the *traceability* of metrics, how metrics are translated from one phase of design to the next one, and in defining thresholds to discriminate "good" schemata from "bad" ones. Besides, techniques will be needed to *monitor* the metrics and appropriately respond to their deviations during the DW lifetime, in order to better manage extensions and evolutions. Finally, these metrics must be considered from the user point of view, by studying their impact on information analysis: methods must be devised to propagate data quality metrics to query results, like in , and to have data retrieval driven by the quality requirements expressed by users, like in .

# 5.INTEROPERABILITY AND METADATA

The heterogeneity in conceptual and logical models proposed for DWs, together with the wide variety of tools and software products available on the market, has lead to a broad diversity in metadata modeling. In practice, tools with dissimilar metadata are integrated by building complex metadata bridges, but some information is lost when translating from one form of metadata to another. Thus, there is a need for a standard definition of metadata in order to better support DW interoperability and integration, which is particularly relevant in the recurrent case of mergers and acquisitions.

Two industry standards developed by multi-vendor organizations have arisen in this context: the Open Information Model (OIM) by the Meta Data Coalition (MDC) and the Common Warehouse Metamodel (CWM) by the OMG (see for a comparison of the two competing specifications). In 2000, MDC joined OMG for developing the CWM as a standard metadata model. The CWM is a platform-independent metamodel definition for interchanging DW specifications between different platforms and tools. It is based on the standards UML, XMI, and MOF, and basically provides a set of metamodels that are comprehensive enough to model an entire DW including data sources, ETL, multidimensional cubes, relational implementations, and so on. These metamodels are meant to be generic, external representations of shared metadata and to provide a framework for data exchange. Unfortunately, their expressivity is not sufficient to capture all the complex semantics represented by conceptual models, so they hardly can be used for effective integration of different DWs.

An alternative approach in this direction is described in [8], where a notion of dimension compatibility based on information consistency is proposed, aimed at cross-querying over autonomous, federated data marts. We believe that another interesting possibility for integration would be to use domain ontologies in order to establish semantic mappings between different data marts.

# 6.DESIGN FOR NEW ARCHITECTURES AND APPLICATIONS

Advanced architectures for business intelligence are emerging to support new kinds of applications, possibly involving new and more complex data types. The modeling and design techniques devised so far are mainly targeted towards traditional business applications, and aimed at managing simple alphanumerical data. Thus, it appears inevitable that more general, broader techniques will have to be devised. In this section we discuss the impact of some of the new applications and architectures on modeling and design; other related topics, that we do not address here due to space constraints, are active DWs and DWs for the life sciences.

## 6.1Spatial data warehousing

Spatial DWs are characterized by a strong emphasis on spatial data, coming in the form of spatial dimensions or spatial measures. Several works, like, show the advantages of using Geographic Information Systems (GIS) characteristics in the analysis of multidimensional data in specific domains. Other works, like , implemented more general systems mixing GIS and OLAP.

While all existing conceptual models support basic modeling of a spatial dimension (most business DWs include a geographic hierarchy built on customers), location data are usually represented in an alphanumeric format. Conversely, picking a more expressive and intuitive representation for these data would reveal patterns that are difficult to discover otherwise.

Preliminary approaches to conceptual modeling for spatial DWs are proposed in , where multidimensional models are extended with spatial dimensions, spatial hierarchies, and spatial measures. Also topological relationships and operators such as *intersect* and *inside* as well as user-defined aggregate functions are included to augment the expressivity of these models. From the point of view of logical modeling, the main issue raised by spatial warehousing is how to seamlessly integrate the classical ROLAP and MOLAP solutions ( the star schema) with the specialized data structures used in GISs while preserving high-level performance. In this line, investigates the definition of mappings between the geographical dimension of an OLAP tool and a GIS. Finally, as concerns design methods, adequate solutions for properly moving from conceptual to logical schemata in presence of spatial information must be devised.

## 6.2Web warehousing

Web warehouses are DWs that collect Web data. The characteristics of the Web raise new difficulties, mainly due to the semi-structured nature of data, to the lack of control over the sources, and to the frequency of changes on them.

The main challenges in this field are how to integrate heterogeneous web sources and how to automate the process of conceptual design when some or most data sources reside on the Web. Some attempts have been made in this direction, mainly aimed at building a conceptual schema from XML data . In other approaches, like , the design of the Web warehouse is driven by frequent user queries and by data quality. Importantly, the development of the Semantic Web opens new exciting possibilities since knowledge is represented according to formal ontologies capable of expressing semantic relationships, which will allow more powerful methods for conceptual design and for data integration to be devised.

## 6.3Real-time data warehousing and BPM

As DW systems provide an integrated view of an enterprise, they represent an ideal starting point to build a platform for business process monitoring (BPM). However, performing BPM on top of a DW has a deep impact on design and modeling, since BPM requires extended architectures that may include components not present on standard DW architectures and may be fed by non-standard types of data (such as data streams). In particular, the fact that BPM implies real-time requirements leads to rethinking ETL components, making the ETL design techniques devised so far questionable. In addition, achieving satisfactory performance for continuous monitoring queries will require more sophisticated logical models for storing data cubes. Arising design issues are summarized in [26]:

- *Right-time design*. While strict real-time will not actually be needed for most applications, data processing must take place in so-called right-time, meaning that information must be ready and

complete not later than required by the decision-making process. Thus, a relevant problem for the designer is to understand what is the right-time for the specific business domain.

- *KPI and rule design*. BPM architectures typically include dashboards for viewing key performance indicators (KPIs) and inference engines for managing business rules aimed at giving the decision maker an accurate and timely picture of the business. Hence, suitable techniques for modeling and designing KPIs and business rules, capable of establishing a conceptual connection with the related business goals and of coping with quickly changing requirements, will be necessary.

- *Process design*. In BPM a leading role is played by processes. Hence, BPM design also requires to understand business processes and their relationships in order to find out the relevant KPIs and rules, and to determine where the data to compute them can be found.

### 6.4Distributed data warehousing

As in distributed databases, in distributed data warehousing a new phase needs to be added to the design method: the one for designing the distribution, from both the architectural and the physical points of view. During architectural design, general decisions will be taken about which distribution paradigm (P2P, federation, grid) better suits the requirements, how to deploy the DW on the infrastructure, which communication protocols to use, etc. For example, makes the case for a P2P infrastructure for warehousing XML resources, whereas reports how DW systems can be deployed on a grid. On the other hand, the physical point of view mainly addresses how to fragment the DW and how to allocate fragments on the different sites in order to maximize local references to data and to take advantage of the intrinsic parallelism arising from distribution, thus optimizing the overall performance. Though some approaches to fragmentation of DWs have been tempted , they are mainly aimed at exploiting local parallelism or at designing ad hoc view fragments for a given workload.

Indeed, distribution is particularly useful in contexts where new data marts are often added, typically because of company mergers or acquisitions. In this case, the most relevant issue is related to integration of heterogeneous data marts as already mentioned in Section 5.

## 7.CONCLUSION

In this paper we have discussed open issues related to modeling and design of DWs. It is apparent that, though these topics have been investigated for about a decade, several important challenges still arise. Furthermore, ad hoc techniques are required for dealing with the emerging applications of data warehousing and with advanced architectures for business intelligence. Besides, the need for real-time data processing raises original issues that were not addressed within traditional periodically-refreshed DWs. Thus, overall, we believe that research on DW modeling and design is far from being dead, partly because more sophisticated techniques are needed for solving known problems, partly because of the new problems raised during the adaptation of DWs to the peculiar requirements of today's business.

## Acknowledgment

We would like to thank the anonymous referees for their careful reading and constructive comments, which helped to improve the presentation. In addition, we would like to warmly thank all the friends who participated in the Dagstuhl Seminar for sharing their ideas with us: Alex Buchmann, Karen Davis, Matteo Golfarelli, Joachim Hammer, Matthias Jarke, Manfred Jeusfeld, Mirek Riedewald, Nick Roussopoulos, Markus Schneider, Timos Sellis, Alkis Simitsis, Dimitri Theodoratos, A Min Tjoa, and Panos Vassiliadis. This paper is based upon our section "Design and Modeling" of an unpublished draft

co-authored by all Dagstuhl participants. Our work has been partially supported by the Spanish Research Program PRONTIC and FEDER under project TIN2005-05406, by the Valencia Government (Spain)under DADASMECA, and by the CastillaLa Mancha Government (Spain) under DADS.