SIRIUS: An Approach for Data Warehouse Refreshment

Stella Gatziu, Athanasios Vavouras, Klaus R. Dittrich

Technical Report **98.07**
June 1998

Department of Computer Science, University of Zurich
{gatziu, vavouras, dittrich}@ifi.unizh.ch

Abstract

Data warehouse refreshment incorporates issues on how to keep the data warehouse up-to-date. Until now, it has not received great attention and has only been investigated by applying techniques of materialized views. In this paper, we introduce the SIRIUS project which investigates a general approach for incremental data warehouse refresh such that only modifications of operational data are propagated into the data warehouse. Our approach can be used with a wide variety of heterogeneous operational sources and independently of a particular database system used for the persistence of the data warehouse.

Keywords: data warehousing, data warehouse refresh process

1 Introduction

The topic of *data warehousing* comprises architectures, algorithms, models, tools, organizational and management issues for integrating information from several operational systems and for storing this information in a data storage called a *data warehouse* [1] (DWH) [1, 3, 5, 15, 16, 20]. The information stored in the data warehouse is targeted for decision support, e.g., by applying data mining techniques, or for direct querying, e.g., with OLAP-Tools. Thus, in contrast to operational systems which contain detailed, atomic and current data accessed by OLTP (On-line Transactional Processing) applications, data warehouses contain integrated, consolidated and historical data (potentially over a long period of time).

A *data warehouse system* (DWS) includes the data warehouse and all components responsible for building, accessing and maintaining the DWH. Implementing a concrete data warehouse solution is a complex task that includes the following major issues:

- DWH *configuration process*: According to user requirements for information, the DWH designer must determine the desired operational data, the appropriate operational sources, the way data will be extracted, transformed and integrated, and how the DWH data will be accessed during analysis.

1. Note that we consider the data warehouse as a physical storage of the integrated information and not as the logical view of the data physically stored in the various operational systems.

- DWH *refresh process*: Data stored in a DWH reflects the state of several operational systems. After the initial load (the first load of the DWH according to the DWH configuration), modifications of operational data must be propagated into the DWH.
- DWH *maintenance*: During the operation of the data warehouse, several maintenance tasks like archiving and purging of DWH data or DWH monitoring must be performed.

Until now, the research community has made attempts to solve particular problems in data warehousing using well-known concepts and results from other areas (like materialized views [12, 13, 14, 22] and indexing [7, 21, 10]). The WHIPS project at Stanford University investigates a wide spectrum of data warehousing problems [3, 19] based on techniques of materialized views. The Esprit Project DWQ (Foundations of Data Warehouse Quality) aims at the development of a semantic foundation that will allow the designers of data warehouses to link aspects like the choice of models, richer data structures and rigorous implementation techniques to data quality [6].

In the context of the SIRIUS project (**S**upporting **I**ncremental **R**efreshment of **I**nformation Wareho**u**se**s**), we investigate the data warehouse refresh process with the main focus on

- the support of incremental warehouse refresh. Data warehouse volumes can reach hundreds of gigabytes or even several terabytes. Thus, avoiding a complete reload of DW data is a crucial issue and can reduce the refreshment time of DW-data significantly.
- the applicability of our approach in data warehouse environments consisting of a wide variety of heterogeneous operational sources (various database systems, flat files, etc.).
- providing facilities in order to refresh a DWH independently of how warehouse data are persistently stored, i.e., we make no assumptions about the database system used for the DWH and about modeling concepts.
- the support of the automatic execution of the whole refresh process based on user-defined specifications (e.g., about the start of the refresh process).
- facilitating the adaptation of new refresh techniques which might, e.g., become necessary because of the extension of a DWH by new sources or new kinds of data from the same sources.

In this paper, we introduce the SIRIUS approach for incremental refresh and we show how the aforementioned goals of SIRIUS are achieved. As mentioned above, the topic of data warehouse refresh so far has been investigated in the research community only in relation with techniques for maintaining materialized views [4, 11, 14, 22, 23, 24]. To the best of our knowledge, SIRIUS is the first approach which deals with the refresh problem without restricting the DW to a set of views on operational data. This allows us to widely explore the refresh problem considering also issues like the cleaning of the operational data. Particularly, the rule-based nature of our approach allows an easy adaptation and extension of the refresh process according to concrete application requirements.

The paper is organized as follows. Section 2 presents a running example of a data warehouse application for a mail-order business. Section 3 introduces the general issues of the refresh process. In Section 4, we give an overview of the SIRIUS architecture, while Sections 5, 6, 7 and 8 give a detailed description of the functionality of the involved components. The paper concludes with Section 9 which presents our current and future work.

2 A Running Example

In our running example, a mail-order business aims at building a data warehouse to support decision making for different user groups and departments (illustrated in Figure 1).

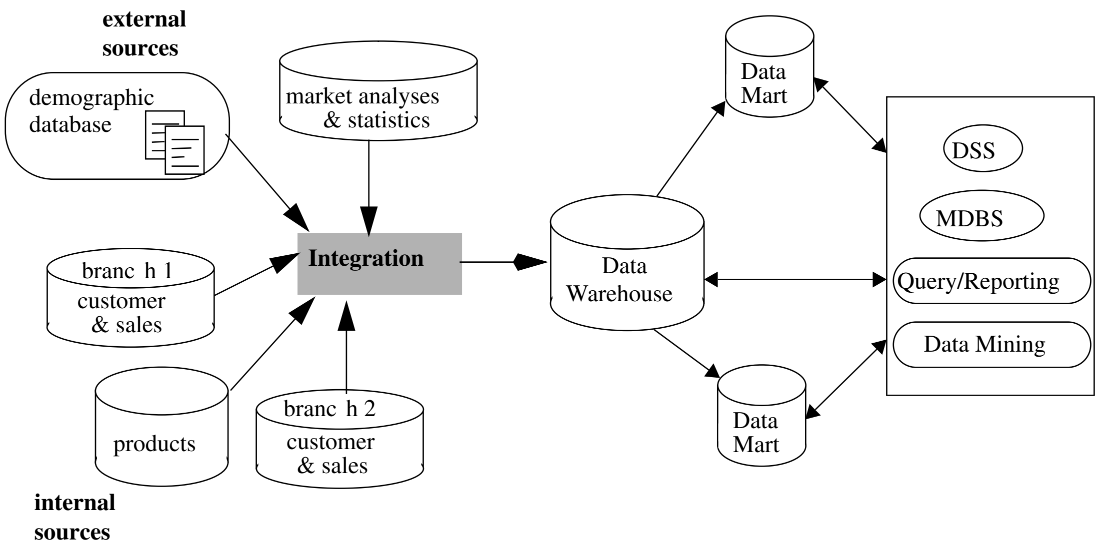

Figure 1 The Data Warehouse System for the Mail-Order business

The central DWH stores products, sales and customer data that are drawn from different information systems. We assume that each company branch locally stores its own sales data (e.g., sold items, quantity, price, date, etc.) and customer data (e.g., name, address, age, gender, marital status etc.). All branches access the same product catalog, which stores information like product number, description, price, marginal return etc. and is managed by the central marketing division. Besides, data delivered from the company branches, external data, like demographic data (in order to classify all customers according to several characteristics) and several market analyses (e.g.,about market shares and market demand) will be integrated into the DWH.

3 An Overview of the Data Warehouse Refresh Process

After the initial loading of the data warehouse, one main task of the DWH-operation is to keep the DWH up-to-date such that it reflects the current state of the various operational sources. A straightforward way is a complete reload of the DWH data (in the same manner as for the initial load). This is, however, very inefficient since data warehouse volumes can reach hundreds of gigabytes or a few terabytes.

In contrast, refreshing the DWH *incrementally* can reduce the effort for the refresh of the DWH-data by a crucial factor. "Incremental" means to consider modifications of the *relevant* [2] entities of the operational systems and to refresh the data warehouse accordingly. This requires that the DWS can be informed about operational data modifications which can be either inserts of new data, deletes, or updates of existing data. The DWS knows *how* each of these operations is performed to achieve the DWH-data modification.

To give an impression of how an insert operation could be performed, let us consider the insertion of a new customer tupel `T1=[value`$_1$`,value`$_2$`,...,value`$_n$`]` into the operational database of a company branch. In this case, the DWH refreshment induces the insert of a a new tupel `T2` into the DWH (if the DWH is persistently stored on a relational system). Tupel `T2` consists of all or a part of the values of `T1` (say, only `value`$_1$, `value`$_2$) and perhaps of new values which are ei-

2. Obviously, from each operational source a part of its entities (tables or classes) are "relevant" for the refresh process, i.e., they contain data which are also stored (in some form) in the DWH. In the sequel of the paper, we often talk about "relevant entities" or "relevant data".

ther derived (e.g, computed from existing values like `average (`$\texttt{value}_5$`, `$\texttt{value}_6$`)`) or retrieved from other sources (e.g., statistical data residing in a statistical database). In this case, the DWS receives the information about $\texttt{value}_1$ and $\texttt{value}_2$, computes further values and inserts tupel `T2` into the DWH.

If we take a closer look at the relevant entities of the operational sources regarding their meaning for the refresh, we can distinguish between entities where each kind of modification (insert, delete and update) must be propagated into the DWH (e.g., data about customers, products, etc.), and entities where only insert operations are of interest (e.g., in the case of sales data, we are interested only in new sales data that emerged since the last DWH refresh).

Until now, we only mentioned *that* the DWS can be informed about operational data modifications. A further question is *when* the refresh process begins. The DWS should provide a variety of possibilities for defining the exact time of data extraction, for example:

- periodically, e.g., every day at 19:00, every Friday at 24:00, at the end of every month, etc.,
- depending on changes of operational data, e.g., when a certain attribute value limit in one or more operational systems has been reached,
- on explicit request of the data warehouse administrator or a DWH-user.

Once the beginning of the refresh process is signalled, the DWS should access at this specific point of time all involved operational sources to guarantee the consistency of warehouse data.

4 The SIRIUS Approach for Data Warehouse Refreshment

A data warehouse system (DWS) includes the data warehouse and all components responsible for building, refreshing, accessing and maintaining the DWH. In SIRIUS, we consider the *Data Warehouse Refresh Manager* (DWRM) as a central component of a DWS which has the knowledge about the tasks that must be accomplished during the DWH refresh process and is responsible for their execution. Figure 2 illustrates the DWRM and components of a DWS relating to the refresh process (in the sense that they cooperate with the DWRM).

The DWRM consists of

- a set of cooperating *mediators*. A mediator provides intermediary services [18] and includes the knowledge how a specific DWH-task can be solved. Each SIRIUS mediator is responsible for the execution of a particular task of the DWRM, like data extraction, cleaning and loading.
- a *metadata repository*. The metadata repository is used for the persistent storage and management of all metadata used in the refresh process in a way that facilitates the convenient and efficient definition and access of the metadata.
- *a coordinator*. The DWRM has to accomplish a number of tasks in which several mediators are involved. The coordinator is responsible for initiating, controlling and monitoring the entire refresh process.

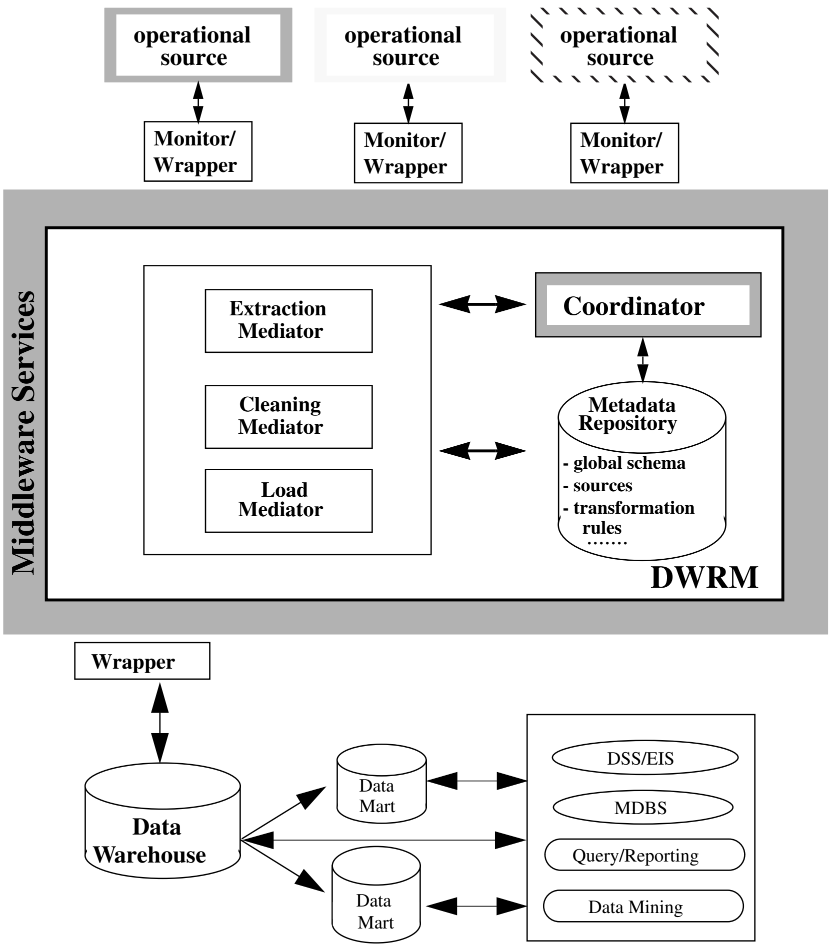

Figure 2 Data Warehouse Refresh Manager as part of a DWS environment

The DWRM cooperates with operational sources through appropriate *wrappers/monitors* which detect relevant data modifications, translate them into the common warehouse format, and send them to the DWRM. The interaction between the DWRM and the further components of the DWS is supported by *middleware services* which provide distribution transparency and are responsible for identifying the suitable component providing a requested task. Object-oriented, service-specific middleware (like CORBA [8, 9]) is an appropriate communication platform.

In the sequel of the paper, we discuss each of the components of the DWRM in detail. How the refresh process will be executed according to our architecture is discussed in the presentation of the Coordinator (Section 8).

5 Monitor/Wrappers

Each Monitor/Wrapper (M/W) is responsible for the detection of data (residing in the appropriate operational source) that has changed since the last DWH refresh. Depending on the kind of integrated operational systems, there are several techniques that can be used for this purpose:

- *log-based* monitoring: Assuming that the source system is a database system maintaining a log, log entries with information about committed transactions that changed relevant operational data can be used in order to refresh the DWH incrementally. Once the beginning of the refresh process is signalled, the M/W inspects the log file and extracts the relevant modifications that have occurred since the last execution of the refresh process.
- *trigger-based* monitoring: For sources that support active mechanisms [17] like triggers, appropriate events can be defined on the update operations of the relevant data items. The source system gathers all information about updates, and notifies the corresponding M/W.
- *replication-based* monitoring: Using replication services of commercial DBMS is a further possibility to detect changes occurring in source systems. Tools like IBM's Data Propagator and Sybase's Replication Server provide mechanisms for propagating changes of base tables outside the operational environment.

- *application-assisted* extraction: Particularly for non-DBMS data management systems, changing existing applications or implementing new ones that notify data changes is the only possibility to support the incremental DWH refresh process. Creating snapshots of relevant data and comparing it (on a per-record basis) with the previous version which has been used for the DWH refresh could be a solution for this problem. A further possibility is to change application programs to timestamp changed data. The M/W will periodically poll the source and select data with a timestamp greater than the one of the previous refresh.

Relevant operational data reside in a wide range of information systems, running on diverse platforms and having a variety of external representations, including fixed and variable length records, character and binary formats, etc. This data must be "homogenized" according to the defined global warehouse format in order to allow further processing steps. For this purpose, after detecting and collecting relevant changes, each M/W transforms changed data from the local representation into the global warehouse format.

As mentioned in Section 3, in some cases it is useful to define the start point of the DWH refresh process in conjunction with certain changes in operational systems, e.g., when a predefined (value) limit has been reached. In the SIRIUS approach, information gathered by each M/W about changes of operational data can also be used for defining the beginning of the refresh process. In this case, the M/W notifies the Coordinator about the occurrence of the interesting event and waits for the initiation of the refresh process.

6 The Metadata Repository

In a DWS, there is a wide variety of metadata that has to be defined and accessed during the refresh process by the involved components (like the mediators and the Coordinator), including:

- the description of operational systems and their contents
- the description of involved components
- the transformations required to move data from the sources into the DWH
- the rules for data extraction, filtering and validation
- the documentation of executed transformation steps

A central element of the metadata repository is the *global schema* which provides the conceptual view of the warehouse data as it is visible to the mediators and the DWH administrator. The global schema describes the structure of the warehouse data and is usually derived from one or more enterprise conceptual data models. On the other hand, a *storage schema* is used for defining the structure of the DWH data as it is stored in the database system used for the persistence of the DWH and as it is usually visible to the DWH users. Distinguishing between a global and a storage schema enables the independence from the used persistence component.

The global schema of our running example is shown in Figure 3. The schema is defined in an extended UML notation[3] (we have added a prefix for a so-called history attribute, i.e., an attribute on which a history is defined).

Other information contained in the metadata repository is strongly related to the global schema. For example, information about how to map data structures of the operational systems to the concrete storage schema or what kind of further preprocessing is needed before loading data into the data warehouse is defined in conjunction with the global schema.

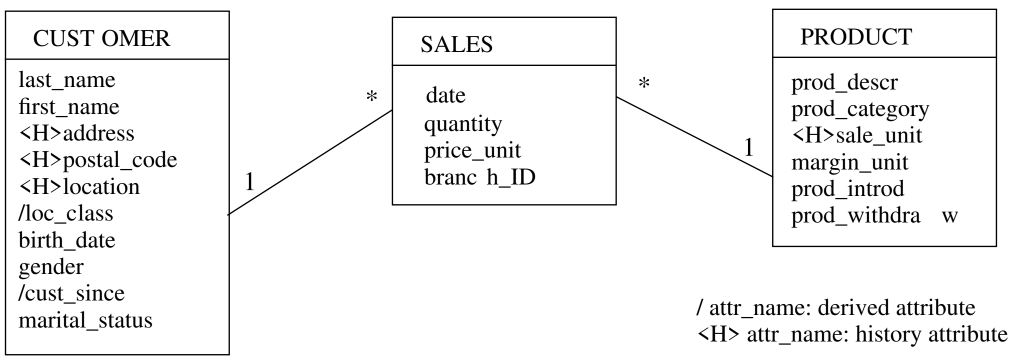

Figure 3 The global schema of our running example

7 The Mediators

In SIRIUS, a mediator includes the knowledge of how a specific DWH-task can be solved. Each mediator is responsible for the execution of a particular task of the DWRM, like data extraction,

3. Note that in this paper, we only aim to illustrate the global schema and we confine ourselves to present a particular model used for its description. The development of a sufficiently expressive model is subject of our current work.

cleaning and loading. We assume that there are several techniques which can be used for the execution of such a task, depending on the specific properties of a concrete DWH solution. Mediators include the knowledge about how these tasks can be implemented in the form of so-called *mediator rules*. Note that there are several kinds of mediator rules due to the diversity of the accomplished tasks. Mediator rules are defined during the data warehouse configuration process and are applied during data warehouse refresh. During the lifetime of the DWS, new mediator rules can be defined adopting new techniques which are necessary because of the extension of a DWH by new sources or new data from the same sources.

7.1 Extraction Mediator

During the DWH configuration process, information about which operational system will provide the data for the DWH has been defined in the metadata repository. We assume that *for each attribute of the global schema* there are *one or more sources* from which instances of this attribute will be stored in the DWH. As soon as the beginning of the refresh process is signalled by the Coordinator, the Extraction Mediator reads the information stored in the metadata repository about the assignment between global schema and operational sources, and initiates the appropriate M/W.

The Extraction Mediator collects the results returned by the various M/W and performs the appropriate transformation rules such that the extracted data conform to the global schema. For this purpose, various operations may be necessary. First, data that come from different sources must be merged. In our example, detail sales data that are separately stored by each company's branch must be merged by the Extraction Mediator to build the company's over-all sales data. Furthermore, the different schematic representations of multiple operational systems must be reconciled. For example, some systems store address information in a single, aggregated field. Assuming that the same information is defined in the global schema as a set of individual fields (last name, first name, address, location, etc.), the Extraction Mediator must split the extracted field and build the individual ones.

Moreover, the extracted operational data in some cases contain attributes that will not be further used for the DWH. Thus, the Extraction Mediator must perform an additional *filtering* to select the needed attributes and eliminate superfluous data.

The structure of data processed by the Extraction Mediator should conform to the global schema before further processing steps like data cleaning (Section 7.2) or calculation of derived and aggregated data (Section 7.3) can be executed.

7.2 Cleaning Mediator

Usually, data extracted from operational systems contain errors, and must first be cleaned before loading it into the data warehouse. Data values from operational systems can be incorrect, inconsistent, unreadable or incomplete. Furthermore, different formats and representations are used in the various operational systems. Particularly, for the integration of external data, cleaning is an essential task in order to get correct data into the data warehouse.

Improving data quality is performed by the Cleaning Mediator using several cleaning rules for the following activities:

- reconcile semantic differences between multiple sources, due to the use of homonyms (same name for different things), synonyms (different names for same things) or different units of measurement,
- transform and enrich data to correct values using rules related to data semantics (in contrast to the transformations related to data structures accomplished by the Extraction Mediator),
- identify and eliminate duplicates and irrelevant data.

In our example, the Cleaning Mediator has to perform various checks. For instance, customer's postal codes (or further address information) must be validated using auxiliary (external) address databases. Since address fields will be further used as a criterion to classify customers (see Section 7.3) and analysis of customer profiles is based on this information, it is very important to get correct values for this data. Furthermore, different synonyms may be used in the operational sources for the attribute "marital_status", e.g., '1 / 2', 's / m' and 'single / married'. These different representations must be cleaned and treated in a uniform way. Finally, duplicates of customer data (the same customer has purchased a product from different branches) must be eliminated by the Cleaning Mediator.

7.3 Load Mediator

In the last step of the warehouse refresh process, extracted and cleaned data must be complemented with further information and then loaded into the DWH. In most cases, additional pre-processing is required before loading data into the DWH, for example:

- enrich loaded data with time-related information (usually by timestamping data with the update date),
- provide a common level of detail for data from different sources and build aggregations,
- calculate derived data.

Whereas several operational attributes can be directly mapped to the warehouse schema, some others have to be derived according to value-dependent rules. For instance, an attribute "loc_class" may be stored in the warehouse to classify each customer according to his/her residence (this classification can be made based on the external demographic database). Another example for a derived attribute that must be calculated by the Load Mediator is the date "customer_since". Assuming that this information is not explicitly stored in the source system, there must be a (business) rule to calculate this value, e.g., set "customer_since" equal to the earliest date for which a sale record with the related customer can be found.

8 The Coordinator

We coarsely present in this section how the refresh process will be executed according to our architecture. We assume that during DWH configuration, one of the alternatives mentioned in Section 5 has been selected for starting a refresh process. Furthermore, for each attribute of the global schema, the exact steps of the refresh process (extraction, cleaning and loading) have been defined in the metadata repository. For instance, updating sales data in our example includes extracting operational data from each branch, bringing it into a common format and level of detail, and finally loading the data into the DWH. Additionally, we assume that all mediator rules have been defined as well.

The Coordinator is then responsible for *initiating* and *controlling* the execution of this process. For the initialization, active mechanisms (as proposed in the field of active databases systems [17]) can be used to implement this behavior in a simple but powerful way. For instance, the beginning of the refresh process could be defined as a reaction on the occurrence of a primitive event (e.g., a time event) or of a composite event (e.g., modifications in operational sources which are detected by the corresponding M/W and notified to the Coordinator).

Besides the invocation of the adequate components, the Coordinator is also responsible for the documentation of the refresh process in a uniform way. For this purpose, the Coordinator provides a common interface to all mediators which can be used to report the successful or failed execution of individual steps.

9 Conclusion

In this paper, we introduced the SIRIUS approach for incremental refresh and sketched first ideas of how to achieve the goals of SIRIUS by proposing the use of

- a conceptual view of the warehouse data in the form of a global schema which describes the structure of the warehouse data as it is visible to the mediators and the DW administrator. Distinguishing between a global and a storage schema enables the independence from the DBMS used for the storage of the DWH data.
- various mediators having the knowledge about performing a specific task in the refresh process. The use of mediators enables the grouping of activities must be done during the refresh process into components which cooperate with well-defined interfaces. Mediators include the knowledge about how these activities can be implemented in form of rules. This enables the flexibility and extensibility of the refresh process to adapt different needs for information gain and different technical environments of each organization.
- a coordinator of the involved mediators. During refresh, the DWS must accomplish a number of tasks. The coordinator is responsible for the automatic execution of all these activities.

We currently implement a data warehouse for the example of the mail-order business presented above. Operational sources are database systems (using Oracle and O_2) as well as a flat files. We also implement various wrappers according to the classification of Section 5. The DWH is implemented using DB2.

The main focus of our future work is the investigation of constructs for the specification of the various mediator rules. Beyond, we deal with the implementation of these rules, focusing on appropriate execution models. Furthermore, we investigate how ECA-rules can be used for the implementation of the Coordinator and of event-driven mediator rules, e.g., the rules of the Load Mediator. For this purpose, we consider active functionality provided by a component that allows the definition and processing of ECA-rules in a distributed and heterogeneous environment and is separated from a particular DBMS.

References

1 S. Chaudhuri, U. Dayal. An Overview of Data Warehousing and OLAP Technology. *ACM SIGMOD Record, 26:1*, March 1997.

2 A. Gupta, I.S. Mumick. Maintenance of materialized views: Problems, techniques, and applications. *IEEE Data Engineering Bulletin, 18(2)*, June 1995.

3 J. Hammer, H. Garcia-Molina, J. Widom, W. Labio, Y. Zhuge. The Stanford Data Warehousing Project. In [15].

4 N. Huyn. Multiple-View Self-Maintenance in Data Warehousing Environments. *Proc. of the 23rd Intl. Conf. on Very Large Data Bases*, Athens, Greece, 1997.

5 W. H. Immon. *Building the Data Warehouse*. John Wiley, 1996.

6 M. Jarke, Y. Vassiliou. Data Warehouse Quality: A Review of the DWQ Project. Invited paper, *Proc. 2nd Conf. on Information Quality*, Massachusetts Institute of Technology, Cambridge, May, 1997.

7 T. Johnson. Coarse Indices for a Tape-Based Data Warehouse. *Proc. of the Fourteenth Intl. Conf. on Data Engineering*, Orlando, Florida, Februar 1998.

8 Object Management Group. *CORBA services: Common Object Services Specification*. Object Management Group, Inc. (OMG). March 1995.

9 Object Management Group. *The Common Object Request Broker: Architecture and Specification, Version 2.0*. Object Management Group, Inc. (OMG). July 1995.

10 P. O'Neil, G. Graefe. Multi-Table Joins Through Bitmapped Join Indices. SIGMOD Record, Vol. 24, No. 3, September 1995.

11 D. Quass, A. Gupta, I. S. Mumick, J. Widom. Making Views Self-Maintainable for Data Warehousing. *Proc. of the Conf. on Parallel and Distributed Information Systems*, Miami Beach, FL, December 1996.

12 N. Roussopoulos. Materialized Views and Data Warehouses. *SIGMOD Record, 27(1), p21-26*, March 1998.

13 D. Srivastava, S. Dar, H.V. Jagadish, A.Y. Levy. Answering Queries with Aggregation Using Views. *Proc. of the 22th Intl. Conf. on Very Large Data Bases*, Bombay, India, September 1996.

14 M. Staudt, M. Jarke. Incremental Maintenance of Externally Materialized Views. *Proc. of the 22th Intl. Conf. on Very Large Data Bases*, Bombay, India, September 1996.

15 J. Widom (ed.). *Special Issue on Materialized Views and Data Warehousing, IEEE Data Engineering Bulletin*, 18:2, June 1995.

16 J. Widom. Research Problems in Data Warehousing. *Proc. of the 4th Intl. Conf. on Information and Knowledge*, Baltimore, 1995.

17 J. Widom, S. Ceri (ed). *Active Database Systems: Triggers and Rules for Advanced Database Processing*. Morgan-Kaufmann, 1996.

18 G. Wiederhold. Intelligent Integration of Information. *Proc. of ACM SIGMOD Intl. Conf. of Management of Data*, Washington, May 1993.

19 J. Wiener, H. Gupta, W. Labio, Y. Zhuge, H. Garcia-Molina, J. Widom. A System Prototype for Warehouse View Maintenace. *Proc. of the ACM Workshop on Materialized Views: Techniques and Applications*, Montreal, June 1996.

20 M-C. Wu, A. P. Buchmann. Research Issues in Data Warehousing. *Datenbanksysteme in Büro, Technik und Wissenschaft: GI-Fachtagung*, Springer-Verlag, Ulm, 1997.

21 M-C. Wu, A. P. Buchmann. Encoded Bitmap Indexing for Data Warehouses. *Proc. of the 14th Intl. Conf. on Data Engineering*, Orlando, Florida, Februar 1998.

22 J. Yang, K. Karlapalem, Q. Li. Algorithms for Materialized View Design in Data Warehousing Environment. *Proc. of the 23rd Intl. Conf. on Very Large Data Bases*, Athens, Greece, August 1997.

23 J. Yang and J. Widom. Maintaining Temporal Views Over Non-Historical Information Sources For Data Warehousing. *Proc. of the 14th Intl. Conf. on Data Engineering*, Orlando, Florida, Februar 1998.

24 Y. Zhuge, H. Garcia-Molina, and J. L. Wiener The Strobe Algorithms for Multi-Source Warehouse Consistency. *Proc. of the 4th Intl. Conf. on Parallel and Distributed Information Systems*, (PDIS '96), December 1996.