

The construction and maintenance of materialised object-oriented views in data warehousing systems*

Robert Wrembel

Poznań University of Technology, Institute of Computing Science

Piotrowo 3A, 60-965 Poznań, Poland

Robert.Wrembel@cs.put.poznan.pl

1. Introduction

For a few recent years special attention has been paid to relational data warehousing systems [Wid95], where views are used as means for integration of data coming from different distributed sources. Since data warehouses contain huge volumes of data and complex analytical queries are addressed to them, the system efficiency is very important. To speed up the evaluation of OLAP queries some data are materialised by means of materialised views (see [Rous98] for an overview).

Contemporary database systems used in such areas as: CAD, CAM, CASE, GIS, CSCW, multimedia, and expert systems must support data models allowing to model objects of complex structure and behaviour. This requirement can not be fulfilled by relational databases that are suitable for storing and processing simple data (mainly strings and numbers). For this reason, object-oriented databases are used more and more frequently for modelling, storing, and processing data of complex structure (e.g. program codes, diagrams, and video sequences).

The natural research manifesto and the next technological step is the integration of object-oriented technology with the technology of data warehousing and the development of object-oriented data warehousing systems. Such data warehousing systems allow to store and process complex information. For these systems object-oriented views are essential for the same purposes as in traditional relational systems (i.e., providing: logical data independence, mechanism for data hiding and security, simplification of a database schema, and shorthand for queries). When object-oriented views are to be used in data warehousing, they should additionally support partial and total data materialisation and maintenance of these materialised data. While constructing object-oriented data warehousing systems we face open research problems concerning: (1) the design of efficient and scalable object-oriented views, (2) techniques of object-oriented views materialisation, and (3) the maintenance algorithms of such materialised views.

Several approaches to object-oriented views were proposed in scientific publications (the reviews can be found in [Mot96], [Wre98]). However, none of them supports all the features of an object-oriented view detailed above. As a consequence, the existing approaches can be applied only to a limited range of problems. As it concerns materialised object-oriented views, only few approaches exist. However, the main problem of maintaining consistency between source and materialised data in a view has not been covered yet.

2. The thesis objective

The objective of this Ph.D. thesis is the development of: (1) the formal model of an object-oriented database supporting materialised views, (2) the techniques of object-oriented views construction and materialisation, (3) the algorithms for maintaining consistency between source data and data materialised in a view, (4) and experimental evaluation of the proposed algorithms.

3. State of the art

In the majority of approaches a view is defined as a *virtual class* derived from one or more classes ([AbBo91], [AgMi94], [Bert92], [BFN95], [HeZd90], [KiKe95], [SDA94], [SLT91], [TYI88]), called *base classes*. The approaches of *MultiView* ([Rund92]) and *eXoT/C* ([Dobr97], [DoEd96]) consider a view as a more complex structure, i.e. a schema. In *MultiView* a view is defined as a portion of a database schema containing virtual as well as base classes. In *eXoT/C* a view, called *external schema*, is composed of derived classes only but the *eXoT/C* views can reference base classes in domains of attributes and in method calls.

* This work is partially supported by the grant No. KBN 8T11C04315 from the State Committee for Scientific Research (KBN), Poland

Another aspect to be considered when defining object-oriented views is the integration of virtual classes in a database schema. The approaches of [SLT91], *MultiView*, and [AgMi94] integrate virtual classes in a base schema. Because the integration of view classes in a database schema leads to a few serious problems (see [Wre98] for an overview), the approaches of: [TYI88], *O₂Views* [SDA94], *eXoT/C*, and *Uni/SQL* [KiKe95] use a separated schema for virtual classes.

When views are used in a distributed environment or data warehousing it is desirable to materialise their contents for the reason of efficiency ([LMSS95, QuWi97, SoSa99]), rather than compute them on demand. Materialising data requires mechanisms that will propagate modifications of a database to materialised views. A lot of work has been done to maintain relational materialised views, where efficiency is achieved by using *incremental propagation* ([CeWi91, GMS93, CGLMT96, RSS96, AASY97, CKLMK97, Rous98]). While (to the best of our knowledge) much less work has been done to support object-oriented views materialisation [DoEd98] and maintenance [KuRu95, KuRu96]. In [DoEd98] an arbitrary portion of a source database schema can be replicated (together with associated class instances), but the consistency maintenance between source and materialised objects has not been considered yet, whereas the *MultiView* system ([KuRu95], [KuRu96]) supports incremental view maintenance. However, the mechanism used in *MultiView* may be useful for maintenance of only simple materialised views (based on *selection*, *projection*, or *set* operations). Furthermore, as the approach supports only the materialisation of pointers to base objects, it may not be suitable for data warehousing systems where integrated data should be stored/materialised locally in a warehouse for the reason of efficiency, rather than accessed remotely.

4. Contribution

In our work we proposed so far: a database model with materialised object-oriented views, a method for defining a complex object-oriented view, and criteria for checking the consistency of a view.

We have defined a set of basic features that object-oriented views suitable for data warehousing should possess. These features are as follows: (1) a view should be defined as a schema of an arbitrary complex structure and behaviour, as data to be accessed by views is complex, (2) a view schema should be separated from a base schema to ensure logical data independence, (3) a view should provide means for materialisation of its instances and consistency maintenance of such materialised data.

We extended the model of an object-oriented database, presented in [AHV95] with the following concepts supporting object-oriented views: *view class*, *view object*, and *view schema* [Wre99a]. In our model *object-oriented view* is defined as a *view schema* composed of view classes. Each *view class* is derived from one or more base classes, i.e. classes in a database schema, called *base schema*. View classes in a view schema are connected by inheritance and aggregation relationships. The structure, behaviour, and set of instances of a view class are defined by an OQL-like expression [Wre99b]. Several view schemas can be created from the same base schema. A view schema is derived in the four following steps: (1) creating a new empty view schema, (2) deriving view classes and integrating them in the view schema, (3) rewriting methods in the context of a view schema elements, and (4) checking consistency of a view schema. In order to evaluate the correctness of a given view schema we proposed the three following consistency criteria: (1) completeness with respect to types used to define view classes, called *structural closure*, (2) completeness with respect to methods calls and types used in methods, called *behavioural closure*, and (3) *inheritance hierarchy correctness* [Wre99c].

In our approach a view schema is totally separated from its base schema, i.e. we neither allow to reference base classes from a view schema nor call from a view schema the methods defined in a base schema. The reason for this separation is that a view schema composed of view classes as well as base classes or a view schema that uses methods defined in a base schema has two serious drawbacks. Firstly, such a view schema does not provide a strong mechanism for hiding data, as a user can navigate through aggregation links to base schema and further following these links he or she can read large parts of base objects. Secondly, a view schema does not provide full logical data independence because there is no separation between an external schema (i.e. view) and a logical schema.

Additionally, our model supports the materialisation of instances accessible through a view schema. In order to maintain consistency between source objects and their materialised view counterparts, links between base classes and their corresponding view classes were introduced to the model. Upon materialisation, each view object is assigned a new identifier. Additionally, a list of base object identifiers from which a given view object was created is associated with each view object. For example, when a view class merges four base classes (e.g. a join) one instance of such a view class will have four counterparts in instances of corresponding base classes and

the list will contain four base object identifiers. This information will be used when propagating modifications from base to view objects, in order to find these view objects that are affected by the modification.

Our approach differs from *MultiView* in the concept of view definition and materialisation. In our approach a view schema is totally separated from its base schema, whereas in *MultiView* view classes are integrated into a base schema. Furthermore, in our approach new objects are created as a result of view materialisation but it is not the case of *MultiView* where only pointers to base objects are used. The approach to view definition that we use is similar to the one used in [DoEd98] as both allow to define a view as a separate schema of an arbitrary complex structure and behaviour. However, our approach considers also the maintenance of materialised views.

5. Ongoing and future work

The ongoing work concerns view materialisation functions and view objects creation. Other important issues that will be addressed are the following: the analysis of different policies of view maintenance (immediate, deferred on demand, deferred using batch mode) and the development of algorithms of incremental maintenance of materialised view schemas.

As a next step we are going to implement our concept of view schemas in the O2 system. The matters that need to be implemented are as follows: (1) a view schema manager and its underlying metamodel, (2) a language to manipulate view schemas, (3) the consistency checker, (4) view schema materialisation functions, and (5) algorithms for maintaining consistency between source and materialised data. The efficiency of proposed algorithms will be evaluated by simulation experiments.

References

- [AASY97] Agrawal D., El Abbadi A., Singh A., Yurek T.: *Efficient View Maintenance at Data Warehouses*, SIGMOD Record 1997
- [AbBo91] Abiteboul S., Bonner A.: *Object and Views*, in proc. ACM SIGMOD Conference on Management of Data, 1991
- [AgMi94] Agrawal R., DeMichiel L. G.: *Type Derivation Using the Projection Operation*, in proc. of Conference on Extending Database Technology (EDBT), 1994
- [AHV95] Abiteboul S., Hull R., Vianu V.: *Foundation of Databases*, Adison Wesley Publishing Company, 1995
- [Bert92] Bertino E.: *A View Mechanism for Object-Oriented Databases*, in proc. of Conference on Extending Database Technology (EDBT), 1992
- [BFN95] Busse R., Fankhauser P., Neuhold E. J.: *Federated Schemas in ODMG*, in proc. of Second Int. East/West Database Workshop, Workshops in Computing, 1995
- [CeWi91] Ceri S., Widom J.: *Deriving Production Rules for Incremental View Maintenance*, in proc. of VLDB, 1991
- [CGLMT96] Colby L. S., Griffin T., Libkin L., Mumick I. S., Trickey H.: *Algorithms for Deferred View Maintenance*, SIGMOD Record 1996
- [CKLMK97] Colby L. S., Kawaguchi A., Lieuwen D. F., Mumick I. S., Ross K. A.: *Supporting Multiple View Maintenance Policies*, in proc. ACM SIGMOD, USA, 1997
- [Col+96] Colby L. S., Griffin T., Libkin L., Mumick I. S., Trickey H.: *Algorithms for Deferred View Maintenance*, SIGMOD Record 1996
- [Dobr97] Dobrovnik M.: *Externe Schemata in objekt-orientierten Datenbankmanagementsystemen; Logische Datenunabhängigkeit durch Änderungen über Sichten*, vol. 25, DISDBIS, Infix, 1997
- [DoEd96] Dobrovnik M., Eder J.: *Logical data independence and modularity through views in OODBMS*, in proc. of Engineering Systems Design and Analysis Conference, 1996
- [DoEd98] Dobrovnik M., Eder J.: *Partial Replication of Object-Oriented Databases*, in proc. of ADBIS'98, Poland, 1998
- [GMS93] Gupta A., Mumick I. S., Subrahmanian V. S.: *Maintaining Views Incrementally*, SIGMOD Record 1993
- [HeZd90] Heiler S., Zdonik S.: *Object Views: Extending the vision*, in proc. of Int. Conf. on Data Engineering, 1990
- [KiKe95] Kim W., Kelley W.: *On View Support in Object-Oriented Database Systems*, in W. Kim (ed.): *Modern Database Systems, The Object Model, Interoperability, and Beyond*. Adison-Wesley Publishing Company, 1995
- [KuRu95] Kuno H. A., Rundensteiner E.: *Materialised Object-Oriented Views in MultiView*, ACM Research Issues in Data Engineering Workshop, 1995
- [KuRu96] Kuno H. A., Rundensteiner E.: *Using Object-Oriented Principles to Optimize Update Propagation to Materialised Views*, in proc. Int. Conf. on Data Engineering, 1996
- [LMSS95] Levy A.Y., Mendelzon A.O., Sagiv Y., Srivastava D.: *Answering Queries Using Views*, in proc. of PODS, 1995
- [Mot96] Motschnig-Pitrik R.: *Requirements And Comparison of View Mechanisms for Object-Oriented Databases*, Information

Systems, 21, 1996

- [QuWi97] Quass D., Widom J.: *On-Line View Maintenance*, SIGMOD Record, vol. 26, no. 2, 1997
- [Rous98] Roussopoulos N.: *Materialized Views and Data Warehouses*, SIGMOD Record, vol. 27, no. 1, 1998
- [RSS96] Ross K. A., Srivastava D., Sudarshan S.: *Materialised View Maintenance and Integrity Constraint Checking: Trading Space for Time*, SIGMOD Record, 1996
- [Rund92] Rundensteiner E.: *MultiView: A Methodology for Supporting Multiple View Schemata in Object-Oriented Databases*, in proc. of VLDB, 1992
- [SDA94] dos Santos C. S., Delobel C., Abiteboul S.: *Virtual Schemas and Bases*, in proc. of Conference on Extending Database Technology (EDBT), 1994
- [SLT91] Scholl M. H., Laasch C., Tresch M.: *Updatable Views in Object-Oriented Databases*, in proc. of DOOD, 1991
- [SoSa99] de Sousa M. F., Sampaio M. C.: *Efficient Materialization and Use of Views in Data Warehouses*, SIGMOD Record, vol. 28, no. 1, March, 1999
- [TYI88] Tanaka K., Yoshikawa M., Ishihara K.: *Schema Virtualization in Object-Oriented Databases*, in proc. Int. Conf. on Data Engineering, 1988
- [Wid95] Widom J.: *Research Problems in Data Warehousing*, in proc. of 4th Int. Conf. on Information and Knowledge Management (CIKM), 1995
- [Wre98] Wrembel R.: *Object-Oriented Views: Virtues and Limitations*, in proc. of the 13th International Symposium on Computer and Information Sciences – ISCIS'98, Turkey, 1998
- [Wre99a] Wrembel, R.: *On a formal model of an object-oriented database with views supporting data materialisation*, in proc. of Third East-European Conference on Advances in Databases and Information Systems - ADBIS'99, Slovenia, 1999
- [Wre99b] Wrembel R.: *Deriving consistent view schemas in an object-oriented database (extended version)*, research report, Poznań University of Technology, Institute of Computing Science, No. RA-007/99
- [Wre99c] Wrembel R.: *Deriving consistent view schemas in an object-oriented database*, in proc. of the Fourteenth International Symposium on Computer and Information Sciences – ISCIS'99, Turkey, 1999