Optimising Mediator Queries to Distributed Engineering Systems

Mattias Nyström[1] and Tore Risch[2]

[1] Luleå University of Technology, S-971 87 Luleå, Sweden
Mattias.Nystrom@cad.luth.se
[2] Uppsala University, S-751 05 Uppsala, Sweden
Tore.Risch@it.uu.se

Abstract. Data and computations of a CAD system have been wrapped by a mediator system using CORBA's IIOP Protocol. This allows ad hoc declarative mediator queries to be translated into optimised execution plans sending dynamic IIOP messages to a CAD server. Queries may call computational methods provided only by the CAD system. Dynamic query re-optimisation taking into account what data is currently materialised in the mediator substantially improves the performance compared to static optimisation, despite the added optimisation time. The approach provides both increased flexibility and efficiency compared to the conventional navigational use of CORBA-based CAD interfaces. Furthermore, CAD independences is provided since transparent queries and views can be defined over different CAD systems.

1 Introduction

We have used CORBA's IDL and IIOP to semi-automatically generate interfaces to an external CAD system from a client mediator database system. Such an architecture has the following advantages:

- It provides a flexible ad hoc query interface to the possibly large amount of data that resides inside a CAD system. The queries may involve calls to advanced computational functions available by the CAD.
- It provides a flexible view definition facility where simple specialised views of CAD data can be defined for different kinds of applications.
- By wrapping different CAD systems through the same mediator views, the applications become CAD independent while still having access to the powerful computational functionality of the wrapped CAD systems.
- The mediator technology allows views that access more than one underlying CAD along with other data sources for broad data integration.

Naive use of the CAD system's interface primitives decreases query performance by unnecessary messages and data transfers. Query optimisation techniques are shown to very significantly improve the performance:

1. The order of CAD calls in an execution plan may significantly influence the performance and is substantially improved by cost-based query optimisation using a simple cost model for the CAD interfaces.
2. Incrementally materialising retrieved data saves CORBA communication. Cached data in the mediator significantly influences the query execution plan so different plans are optimal when data has been materialised. We show that dynamic optimisation every time a query is executed pays off significantly, despite the additional time for query optimisation.

The implemented system has been used at Volvo Car Corporation for accessing CAD data from the commercial CAD system I-DEAS [16]. The work shows that cost-based and dynamic query optimisation combined with data materialisation and query re-writes are viable techniques for accessing engineering data. The use of the IIOP interface protocol makes the techniques applicable to many other kinds of CORBA-based data producing systems.

This paper is organised as follows. Section 2 discusses related work. Section 3 describes the experimental set-up and performance measurement for some typical CAD queries. Section 4 summarises and outlines ongoing and future work.

2 Related Work

Dogac et al. [2] provide a CORBA view of database data where entire databases are registered as objects, while the work presented here provides a technique for querying CORBA-provided internal application data. Sheu et al. [15] use the Object Transaction Service and wrapper techniques to transparently map CORBA-based object data stores to an object-oriented programming language, without providing query facilities. [7] focus on non-queryable CORBA views of objects extracted from relational databases.

A commonly used method to exchange data between different CAD systems is to translate the internal representation into a neutral file format such as STEP [6] or IGES [18] and then load and translate the file in the receiving system. Such a method can also be used for querying CAD models [8] but queries can then not use computation methods provided by the CAD.

We use cost based query optimisation techniques where cost and selectivity computations are associated with foreign functions [10]. By associating cost models with foreign functions the framework is able to correctly place expensive foreign function calls in the generated execution plans as in [5]. In particular we show that this situation appears when an expensive function has been materialised in the mediator. In order to handle the case of different optimal query execution plans when some functions are materialised we show that even a naive dynamic reoptimisation strategy pays off. The time for query reoptimisation could be further minimised by using some of the techniques in [3,9,1].

3 System Overview

The architecture of the system is illustrated in Figure 1. As mediator database engine we use the AMOS II object-oriented and main memory resident mediator system [14] which uses the mediator wrapper [19] approach to data integration. The mediator engine includes an extensible main-memory database system having a cost-based query optimiser. The presented techniques could be applied for other mediator engines as well.

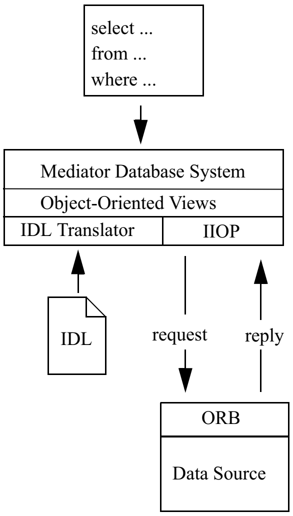

Fig. 1. System Architecture

To access data from external data sources the mediator contains a *wrapper* for each kind of source. A wrapper is a program module in the mediator system performing schema importation and query translation.

Our general IIOP wrapper can access IDL for schema importation and then send dynamically created general IIOP messages from the mediator system to the ORB based on the imported IDL.

The wrapped CORBA methods may have side effects and other properties and the imported IDL definitions therefore have to be complemented with more information, e.g. classification of different kinds of side effects [12,11].

The execution of CORBA calls is relatively expensive because of the overhead in communication, IIOP interpretation, etc. Therefore the mediator system automatically materialises incrementally IIOP member function computations. A costing function is associated with each interface function for cost-based query optimisation and the costs depend on whether data has been materialised or not.

4 Performance Experiments

Performance measurements were made to evaluate the efficiency of queries to the mediator. The measurements were performed using the commercial I-DEAS [16] CAD system giving access to geometrical models using a subset of the CORBA-based Open-IDEAS API [17] for retrieving geometrical objects in the CAD system. Some of this data is illustrated with the ER-diagram in Figure 2.

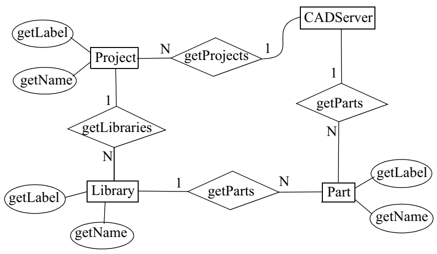

Fig. 2. ER-diagram of subset of CAD system data

The geometrical models, called *parts*, are located in some *library* that belong to a *project*. All objects belong to a *CADServer*.

For the experiments, we defined some typical queries needed by our applications to retrieve various data from the CAD system.

Query 1 retrieves a part when the engineer knows its name, library name, and project name. There are three different levels that must be traversed to retrieve the part and there is a selection condition at each level.

```
               Query 1.
select p
from Part p, Library l, Project pr, CADServer s
where  p = getParts(l)
           and  l = getLibraries(pr)
           and  pr = getProjects(s)
           and "part1" = getName(p)
           and "library1" = getName(l)
           and "project1" = getName(pr);
```

Query 2 finds a library in the CAD system with the label equal to 1 and a part in it named part1'.

```
                Query 2.
select l
from CADServer s, Library l, Part p
where l = getLibraries(getProjects(s)))
          and 1 = getLabel(l)
          and p = getParts(l)
          and "part1" = getName(p);
```

To test the performance of the queries we populated the CAD system with various data sets of different sizes. Four different *data fanouts* for number of parts per library, number of libraries per project, and number of projects per CAD server were used during the tests: F = 2, F = 5, F = 10 and F = 15. Consequently if a data fanout F = 10 is used, the CAD system will have 10 projects where each project will consist of 10 libraries and each library will have 10 parts. This renders into 10 projects, 100 libraries and 1000 parts. The tests where performed on a network with 10 Mbit/second.

The first experiment shows the importance of cost-based query optimisation (Fig. 3). The performance is improved here because restrictions on the data in the optimised plan are being done immediately after objects have been retrieved from the CAD system to the database to prune traversals further down the hierarchy (selection pushing). A greedy cost-based heuristics (Ranksort [10]) was used to optimise the query plan. *Query 1* was executed with varying data fanout, both unoptimised and optimised. The optimisation is guided by cost estimates tailored for the expensive CAD functions.

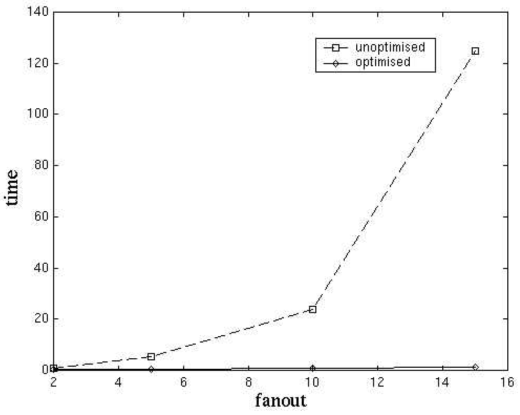

Fig. 3. Optimised vs. unoptimised

The second test, where *Query 2* is evaluated, analyses a query that combines data resident in a materialised function with data that has to be retrieved from the CAD server. *Query 2* selects the library with label 1 and a part stored in it named 'part1'. The performance of *Query 2* is evaluated with two different execution plans. The first plan is generated in a 'cold' mediator with no data materialised. In that plan all data is assumed resident in the CAD system only.

Then a query was executed to incrementally materialise in the mediator proxy objects for all projects, libraries, and parts, except the labels (function *getLabel*).

Query 2 is then reoptimised. In the repotimised strategy the optimiser pulls up the call to the only remaining expensive unmaterialised function *getLabel*, thus minimising the number of costly IIOP messages in favour of access to materialised data.

Figure 4 compares the execution times of the two plans. Notice that the curve with the label 'dynamic reoptimisation' also includes the time to optimise the query. The performance gains by query optimisation are so large in this case that it actually pays off very well to dynamically optimise the query at run-time, rather than using the wrong statically precompiled execution plan.

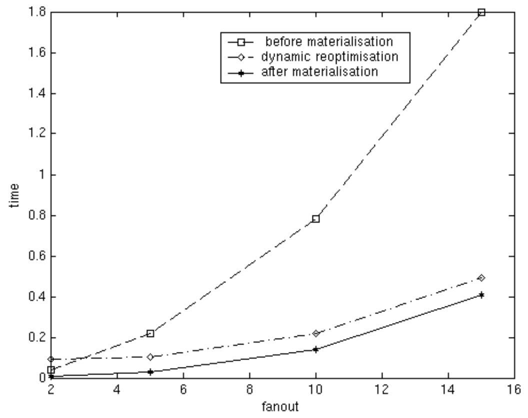

Fig. 4. Query optimised for combining materialised and remote data

5 Conclusions

A general wrapping mechanism for CORBA data sources wraps a commercial CAD system to enable queries to CAD data involving computational methods provided by the CAD system. The architecture provides a very flexible and efficient method to query, e.g., CAD data compared to using a procedural programming language. Furthermore, the architecture enables the use of database query optimisation techniques for scalable and efficient access to CAD data.

To motivate the approach we evaluated the impact of some query optimisation techniques on a number of typical queries to the CAD system. The impact of the query optimisation increases with more data, which is important for distributed engineering applications using large amounts of CAD data.

Ongoing work concerns the use of query transformation techniques to transform query fragments into more efficient direct calls to equivalent CAD function calls. This can be seen as a form of query rewrites in extensible query optimisers[4,13]. Future work includes development of more sophisticated cost models for IIOP methods. For web service support, it should be investigated how to adapt the proposed mechanisms for SOAP-based sources.

Acknowledgement. This project was made in cooperation with the Volvo Car Corporation, Gothenburg, Sweden and the Polhem Laboratory, Luleå, Sweden.

References

1. Avnur, R., Hellerstein, and J.M., Eddies: Continuously Adaptive Query Processing. *SIGMOD Conference 2000*, (2000) 261-272
2. Dogac,A., Dengi,C., and Özsu, M.T.: Distributed Object Computing Platforms. *Communications of the ACM*, Vol. 41 (1998) 95-103
3. Graefe,G. and Ward, K.: Dynamic Query Evaluation Plans. *Proc. of the 1989 ACM SIGMOD Conference*, Portland, OR, May (1989) 358-366
4. Haas, L.M., Freytag, J.C., Lohman, J.M., and Pirahesh, H.: Extensible query processing in starburst, *Proceedings of the 1989 ACM SIGMOD international conference on Management of data*, pp 377-388, June 1989.
5. Hellerstein, J.: Optimization Techniques for Queries with Expensive Predicated. *Transactions of Databases Systems (TODS)*, Vol. 23 (1998) 113-157
6. International Organisation for Standardisation: *Product data representation and exchange-Part 1: Overview and fundamental principles.* ISO 10303-1 (1994)
7. Jungfer,K., Leser, U., and Rodriguez-Tome, P.: Constructing IDL Views on Relational Data. *Conf. on Advanced Information Systems Engineering* (1999) 255-268
8. Koparanova,M. and Risch, T.: Completing CAD Data Queries for Visualization. *International Database Engineering and Applications Symposium (IDEAS 2002)*, Edmonton, Alberta, Canada, July 17-19 (2002)
9. Kabra,N. and DeWitt, D.J.: Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans. *ACM SIGMOD Conf.* (1998) 106-117
10. Litwin,W.and Risch, T.: Main Memory Oriented Optimization of OO Queries using Typed Datalog with Foreign Predicates. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 4 (1992) 517-528
11. Nyström, M.: *Engineering Information Integration and Application Development using Object-Oriented Mediator Databases.* Department of Applied Physics and Mechanical Engineering, PhD thesis, Luleå University of Technology, 2003:04 (2003)
12. Nyström, M.: Multidisciplinary Optimisation with Application to Exhaust System Design. *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-2000-4749,* 6-8 September, Long Beach (2000)
13. Özsu, M.T.,Mu noz, A., and Szafron,D.: An extensible query optimizer for an objectbase management system. *Proceedings of the fourth international conference on Information and knowledge management, CIKM'96* (1996) 188-196
14. Risch, T. and Josifovski, V.: Distributed data integration by object-oriented mediator servers. *Concurrency and computation: Practice and experience*, Vol. 13 (2001) 933-953
15. Sheu, R.K., Liang, K.C., Yuan,S.M., and Lo, W.T.: A New Architecture for Integration of CORBA and OODB. *IEEE Transaction on Knowledge and Data Engineering* Vol. 11 (1999) 748-768
16. Unigraphics Solutions Inc.: *I-DEAS User's Guide v9* (2002)
17. Unigraphics Solutions Inc.: *Open I-DEAS User's Guide v9* (2002)
18. U.S. National Bureau of Standards (NIST): *Initial Graphics Exchange Specification (IGES) – Version 5.1. NISTIR 4412*, (1991)
19. Wiederhold ,G.: Mediators in the Architecture of Future Information Systems, *IEEE Computer*, Vol. 25 (1992) 38-49