December 1998

Data Modeling: Moving to an Object-Oriented DBMS

Michael Chilton
University of Arkansas

Data Modeling: Moving to an Object Oriented DBMS

Michael A. Chilton
University of Arkansas

Abstract

Databases were designed originally to eliminate redundancy in data storage while speeding data retrieval in a specific problem domain. However, because these databases were constrained to specific problem domains, it was impossible to develop new applications without changing the structure of the database. Relational databases were developed to overcome this problem by creating a data storage structure that was independent of the application. This data independence allowed new applications to be developed without concern for the ways in which data is stored, unlike the previous systems. Although these relational databases continue to work well with most applications, the advent of newer technologies has placed demands on data storage that go well beyond their capabilities. As the power of application software has grown, breakthroughs in database technology have been achieved to meet these new demands. Organizations now face the decision of which technology to use, and this choice is not an easy one. What is needed is a model that can be used to structure the decision process, taking into account those issues that affect the outcome of the decision. This paper seeks to construct such a model by enumerating and discussing the issues and providing some insight as to how an organization might incorporate it into their decision process.

Introduction

Currently, relational databases are defined and manipulated in one language (the query language) and the data itself is analyzed separately in the programming language. Each serves a separate function and must be tied together at the interface through separate but equal data typing. This results in what is called an *impedance mismatch* and there are five types of problems associated with it (Rao, 1994):

1. Sets of data which are returned from a database using SQL cannot be manipulated directly by the programming language;
2. Data types must be consistent across the junction between the programming language and the database, but it cannot be checked by either language;
3. Queries can only be performed on persistent objects (data stores), not on transient ones;
4. Query statements must remain separate from programming statements (and are usually embedded within the program, separated by a delimiter of some sort); and
5. The syntaxes do not match and in fact are not even similar.

In order to deal with these data typing problems, two database technologies have been developed, the object oriented database and the object-relational database. These OO databases make it possible to store a complete object as it exists in main memory at run time without having to change its structure. This is true regardless of the complexity of the object (Rao, 1994) and it allows for better performance due to a more realistic representation.

Object-relational databases have also been introduced in order to take advantage of the strengths of the Object Oriented model and the relational database and reduce their inherent weaknesses. The following features are considered necessary in an object-relational DBMS (Hunter, 1997):

1. support for an object model, including attributes, methods, and the characteristics of inheritance, polymorphism, and encapsulation,
2. the ability to support 1:1, 1:many or many-to-many relationships with enforced integrity,
3. an application programming interface (API) that allows the programmer to treat the database as though it were an enhanced relational DBMS,
4. support for SQL3,
5. Indexes, constraints, triggers and security,
6. a table interface, and
7. rich data types (e.g., audio, video, time series, etc).

Decision Criteria

There are several considerations which must be evaluated when deciding which database technology to use in an organization. These considerations fall into four basic categories: 1) what exists in the firm currently, 2) what the future needs are for the firm, 3) what the importance of new applications is, and 4) what resources are available. Options that the firm has are network, hierarchical, relational, object oriented, and object relational database management systems. This paper will discuss only relational, object oriented, and object relational, since network and hierarchical DBMSs are older technologies that do not allow for data independence, which is considered to be a critical factor for a DBMS.

Current and future needs of the firm are critically important. The following figure should assist in understanding current and future needs as they relate to the choice of a database. It has been adapted from the literature where specific applications have been identified for new technology such as that found in Khoshafian (Khoshafian, 1993).

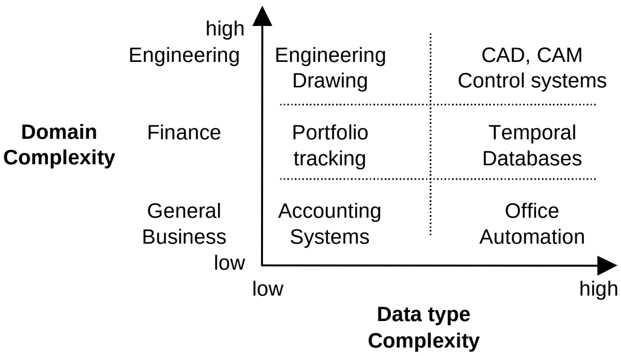

Figure 1. Database/Domain Requirements

Figure 1 displays domain complexity against data type complexity and can be interpreted as follows: domain complexity increases in the upward direction such that general business applications are less complex than those used in finance (or economics), and these are less complex than engineering applications. The table is annotated with examples in each category to demonstrate this increasing complexity. Thus a general business application such as found in an accounting system is less complex than a financial application which might track a portfolio of investments. At the highest level is an engineering drawing application which might create blueprints of some structure or manufactured product, and require storage of that drawing in digital format. Next, data type complexity is shown in the horizontal direction and increases to the right. Again, examples are provided to clearly show the comparison within each domain. Office automation using multi-media data is clearly more complex than an accounting system. Likewise, temporal (or time series) data such as derivative financial instruments are more complex than a portfolio tracker, and CAD/CAM applications are more complex than an engineering drawing system. As one moves upward and to the right, the demands placed on both the software application as well as data storage both become increasingly complex.

In order to use this table effectively, an organization can locate where their requirements for data storage are at the moment and attempt to predict where they might be in the future. If the firm finds itself in the lower left quadrant, the demands here are easily met by relational DBMSs. As the firm moves to the right yet remains in the business domain, the object relational DBMSs become a more appropriate choice due to their extensibility (Hunter, 1997, Vermeullen, 1997). In this case the richer data types of office automation can be added to existing databases allowing a firm to preserve its system while gradually adding new features (Hunter, 1997). This also allows integrating object oriented languages (such as Java) for web applications into existing database structures.

As the firm moves upward and to the right, however, the performance characteristics and the demands of these kinds of systems are clearly handled best by object oriented DBMSs (Khoshafian, 1993). CAD/CAM and knowledge based systems are stretching the limits of traditional programming and database technology—it is important in these cases that an OO DBMS be used.

Next, a firm must ask just how risky this new database technology is. This is a common question asked frequently about a number of different strategic proposals in a firm, not just information systems. Alternatively, the firm must also determine how important the application and the performance is. Jacobson and Lindstrom (1991) have developed a matrix to help answer this question. An adapted form is shown in figure 2. Business value can help provide an approximation of risk, but the lack of empirical evidence in regard to the reusability of software and consequent reduction in developments resources make it nearly impossible to judge risk directly. Additionally, there are no real measures of cost versus return regarding information systems, and so this matrix represents (at least for the present) a suitable alternative.

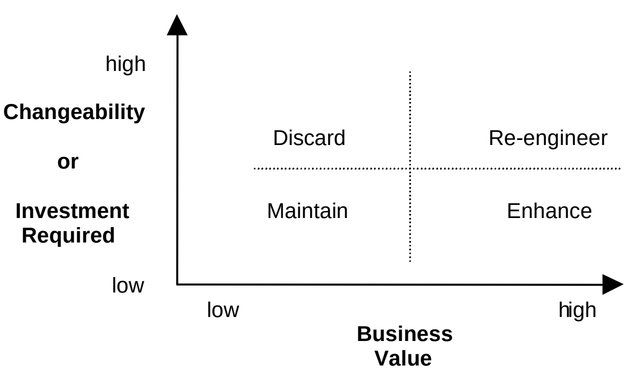

Figure 2. Decision Matrix

Figure 2 plots the amount of difficulty a firm might experience in changing an existing system against Figure 2 plots the amount of difficulty a firm might experience in changing an existing system against the value of the system to the firm. The harder the system is to change, the more likely the firm would be to replace or discard it completely. The greater value it has to the firm, the more likely the firm would be to replace, re-engineer or enhance the system.

The model for making the choice of a database technology proceeds through each of the four considerations. It begins with an assessment of what the firm currently has (the installed base) and what the firm will need to remain competitive in the future. When making these assessments, the decision-makers should look at all possible alternatives regardless of cost, and formulate a "shopping list." Options might include using a structured programming language, SQL-92 and either a relational or object-relational DBMS or using an object oriented language, SQL-3 (or OQL) and either a relational, object-relational or object oriented DBMS.

Once these alternatives are delineated, the firm next needs to determine the importance of the application and the DBMS. Questions such as the need to eliminate problems of impedance mismatch and reduce maintenance costs through code re-use need to be analyzed. Also, can the objective be accomplished using object-relational techniques and thus provide a more gradual transition to new technologies? Which applications need what technologies? Is it necessary to integrate data storage across all applications or can some functions (such as engineering design) be isolated from other functions (such as inventory)? Is integration necessary or can the firm stand some separation?

Finally, the resources available need to be determined. How much can the firm afford in terms of time, labor and money? Will cost savings be achieved for a greater investment today that can be recovered in the future? What level of investment is the firm willing to make today?

Once these questions are answered, the organization can then select a DBMS that will meet its needs. This selection process is not an easy one, but this decision methodology presented here should help guide the firm through the process in such a way that what will be known are the capabilities and limitations of the chosen technology.

References

1. Andleigh, P.K. and Gretzinger, M.R., *Distributed Object-Oriented Data-Systems Design*, Prentice-Hall, 1992.
2. Bancilhon, F., Delobel, C., and Kanellakis, P., editors, *Building an Object-Oriented Database System, The Story of* O_2, Morgan-Kaufmann, 1992.
3. Basili, L., Briand, L.C. and Melo, W.L., "How Reuse Influences Productivity in Object Oriented Systems," Communications of the ACM, (39:10), October 1996, pp. 104-116.
4. Blaha, M.R., Premerlani, W.J., and Rumbaugh, J., "Relational Database Design Using an Object-Oriented Methodology," Communications of the ACM, (31:4), April 1988, pp. 414-427.
5. Bryant, T. and Evans, A., "OO Oversold," Information and Software Technology, (36:1), 1994, pp. 35-42.
6. Carrier, M., "JAVA With and Edge," Object Magazine, (7:6), Aug 97, pp. 50-53.
7. Chorofas, D.N. and Steinmann, H., *Object-Oriented Databases*, Prentice-Hall, 1993.
8. Hunter, S. K., "Cutting to the Chase," Object Magazine, (7:6), Aug 97, pp. 32-41.
9. Jacobson, I. and Lindström, F., "Re-engineering of Old Systems to an Object-Oriented Architecture, SIGPLAN Notices, (26:11) 1991, pp. 340-350.
10. Khoshafian, S., *Object-Oriented Databases*, Wiley & Sons, 1993.
11. Khoshafian, S. and Copeland, G.P., "Object Identity," *Proceedings of OOPSLA-86*, 1986; also appeared in S. Zdonik & D. Maier (eds), *Readings in Object-Oriented Database Systems*, Morgan-Kaufmann,1990.
12. Pepper, J., "What's Corporate America Spending on Objects?" Object Magazine, (6:12), February 1997, pp. 62-65.
13. Rao, R.R., *Object-Oriented Databases*, McGraw-Hill, 1994.
14. Smith, H.A. and McKeen, J.D., "Object-Oriented Technology: Getting Beyond the Hype," Data Base for Advances in Information Systems, (27:2), Spring 1996, pp. 20-29.
15. Vermeullen, R., "Pushing the Limits," Object Magazine, (7:6), Aug 97, pp. 42-49.
16. Weinberg, R., Guimaraes, T. and Heath, R., "Object Oriented Systems Development," IS Productivity Improvement, (7:4), Fall 1990, pp. 18-26.