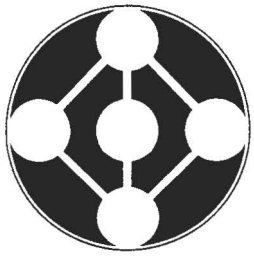

Alternative Technologies

Enabling Productive Co-existence

David McGoveran
Alternative Technologies
13130 Highway 9, Suite 123
Boulder Creek, CA 95006
Telephone: 408/425-1859
FAX: 408/338-3113

I. Introduction

Corporations wishing to move to an RDBMS (relational database management system) environment have developed a variety of strategies, each with its benefits and costs. Among the more recently developing strategies is the notion of integration rather than replacement. This strategy involves a policy of co-existence that can breath new life into legacy applications in a variety of ways. However, it cannot be used productively unless the RDBMS provides the proper facilities. This article will discuss some of the key issues involved.

II. Traditional Strategies

Over the years, IS (information systems) has developed a number of strategies for managing technology change. I will describe five of these "traditional" strategies here. For convenience, I have given them names: direct replacement, staged migration, anticipated migration, competitive replacement, and extinction.

The most severe and risky policy is that of *direct replacement*. The idea is that a new application is written to replace the old one. As soon as the new application is tested, the replacement phase begins. This phase may be realized through either a cutover or parallel operation. Both techniques have benefits and costs.

Staged migration is a technique similar to direct replacement, but in which separate modules are replaced rather than the entire application. Replacement occurs selectively, forming ever growing islands of newer technology. Typically, these islands are interconnected by an integrated RDBMS data model and possible through distributed computing techniques as well. If the RDBMS supports distributed database management, the databases

associated with the individual islands can often be integrated with relative ease.

Anticipated migration refers to a technique that is possible if the designers and developers of the older application planned for eventual replacement. In particular, the application architecture must be highly modular, and the interfaces and functionality well-defined. This permits selective replacement of individual modules. Unfortunately, it does not address the problems of data store conversion.

Competitive replacement takes a somewhat Darwinian view of the software evolution process. For a particular application requirement, several different groups may be given the task of creating a new system. These then undergo trial adoption by the user community and "compete" to be selected.

Another technique that is sometimes used is that of *selective extinction*. Certain applications have a naturally limited life expectancy, usually due to changes in business practices. For example, an application may be designed to support specific products sold by a company. Rather than modify the existing application to support new products, an entirely new application is designed. As the company moves away from the older product, the older applications cease to be useful and are effectively "extinct".

III. Integration and Interoperation

A. An Evolving Technology

Each of the strategies described above for handling migration has serious costs and risks. While legacy applications continue to be selectively replaced, a new imperative has appeared. Enterprise IS is beginning to consider a co-existence strategy, which Oracle Corporation terms *productive co-existence*. The goal of productive co-existence is to adopt new software systems and technology where possible, to leave the more deeply entrenched legacy applications in place, and to establish a framework for cooperation between the systems. The recently released ORACLE7 (ORACLE Version 7) was designed to support a co-existence strategy.

The strategy of co-existence attempts to address the deficiencies of past systems, especially their rigidity, by developing highly adaptive new systems. These new systems are made possible by certain new technologies, discussed in more detail below.

The key benefit of this strategy is that many of the migration techniques used in other strategies can be selectively implemented without introducing significant conflicts. Each legacy application (and indeed individual sub-systems) can be evaluated separately as a candidate for migration. On the one hand, if migration or replacement is not deemed cost-effective, the application can remain in place and new applications can take

advantage of it and its data. On the other hand, if either migration or replacement seem to be cost-effective, the transition can be made at any time so long as the application is relatively independent of other systems.

Despite the attractiveness of co-existence, there are risks associated with it. Without the appropriate enabling technology and without adequate planning, the co-existence may not be productive. For example, if existing systems are non-relational and an RDBMS is implemented, forcing these applications to store data directly in the RDBMS can be extremely disruptive. Typically, performance will suffer unless significant portions of the existing application are rewritten to take advantage of relational capabilities.

Many of the enabling technologies for productive co-existence depend on a good RDBMS. The framework for implementation of the strategy must recognize this fact and promote proper use of relational technology. If this is not done, new applications will suffer from poor return-on-investment, hidden costs, and even loss of data integrity.

B. Overview of RDBMS Requirements

Not all RDBMS products meet the needs of the productive co-existence strategy. If an RDBMS is to be the key component of an enterprise IS strategy, it must be reliable, scalable, cost effective, and open.

A *reliable* RDBMS is, first and foremost, predictable. It is not necessarily one that is fault tolerant, although high availability is certainly one aspect of reliability. End-users must be able to rely on the RDBMS to deliver timely data, without corruption. Administrators must be able to rely on the RDBMS with respect to capacity planning and other database administration tasks. Data integrity features, non-conflicting data definition, data manipulation, security, and maintenance operations, transaction isolation, and high concurrency are key aspects of reliability.

A *scalable* RDBMS is essential in any business with changing needs and workloads. Scalability is often understood in terms of *linearity*. For example, if the workload doubles, performance should be reduced to no less than about fifty percent. Similarly, adding twice as much computing power (memory and CPU) should roughly double performance or perhaps the number of concurrent users. An RDBMS can be scalable either horizontally or vertically. Horizontal scaling refers to the ability to add (or remove!) database servers. It requires support for distribution, portability, and cross-platform interoperation. Vertical scaling refers to the ability to increase (or decrease!) database server computing power, either within a given family of computer hardware and operating systems or, if necessary, using a completely different platform. RDBMS vendors have usually focused on upward scalability. Downward scalability is just as

important, since it permits the company to downsize more easily and to eliminate unnecessary computing resources.

A *cost effective* RDBMS can have an important beneficial impact on all the costs of data processing. The many benefits of relational systems are often lost because the overall cost of ownership of the RDBMS is too high. Ease of use, ease of maintenance, and ease of application development are all important in this regard since they reduce costs due to training, day-to-day operation, and maintenance. In addition, both improved performance and the elimination of data corruption errors should be goals in moving to relational applications. All too often, mis-design of relational applications leads to poorer performance than was found in non-relational applications. The RDBMS should encourage proper design of relational and distributed (e.g. client/server) applications by providing the appropriate facilities.

A key requirment of an enterprise RDBM is that it be extensible or *open*. An extensible RDBMS is achieved in a variety of ways, but most often it means the RDBMS supports for open systems. Open systems support has different meanings for different companies. For some, it means that integration and interoperation with their products is possible. For others, it means that public standards are supported. For still others, it means that proprietary standards are followed and made available to customers.

C. Enabling Technology

Among the enabling technologies for productive co-existence are gateways, TP monitors, standards, and distributed computing. These technologies are improving at a rapid rate, and most RDBMS vendors offer some degree of support for each of them. ORACLE7, its options (the Distributed Database Extension, the Procedural Database Extension, and the Parallel Server Extension), and Oracle's new gateway products were designed to address many of these requirements.

There are several types of database *gateways* offered by RDBMS vendors, each of which may be require a separate hardware platform or either software only. Gateways may support either uni-directional or bi-directional data access between two types of systems. Perhaps the two most common types of gateway are the standard RDBMS gateway and the custom gateway. Standard gateways are provided by the vendor and provide access to foreign data. Typically the data is managed by other vendor's RDBMS products. There is tremendous variability in the degree of transparency and compatibility which such gateways provide. Gateways may provide either direct connection from an application or access via the native RDBMS. Custom gateways are generally supported through vendor-supplied gateway development kits. Custom gateways permit the end-user to obtain access to proprietary and non-database data sources. This capability is extremely important for productive co-existence, since it enables integration of RDBMS-

based applications with legacy applications at the data level.

When existing environments are transaction-based, new technology cannot be integrated without some means of coordinating transactions between applications. Because traditional RDBMS products manage transactions through internal services, it is important to support external transaction services, known as TP monitors. Both UNIX-based and proprietary TP monitors are available. Most UNIX-based TP monitors support a call-level interface standard known as XOPEN/XA. If the RDBMS supports TP monitors, newer applications can be integrated with older applications via the TP monitor at the transaction level. This means that older application modules can be selectively and more readily replaced by newer software without disruption of the existing transaction profiles.

Ideally, the goal of open systems support is low-cost extensibility leading to systems with a longer life expectancy. As long as limited functionality and slower innovation can be tolerated, low-cost extensibility can be achieved best by support for public or *de jure standards* such as the ANSI SQL standard. In principle, competition between vendors can lead to lower cost and easily interconnected applications. When higher levels of functionality are required, users must look to proprietary standards and the innovations of a particular vendor. If the vendor is large enough, such proprietary standards may become public by default (*de facto standards*). Of course, either de jure or de facto standards may restrain the industry from creating truly innovative, more efficient, and more cost-effective solutions.

Among the most confusing of enabling database technologies is that for *distribution.* Part of the confusion stems from the difference between process distribution (distributed computing) and data distribution (distributed database). It is important that an RDBMS product support both. An RDBMS can support distributing computing through its architecture and through certain features such as database stored procedures (either local or remote). The most common distributed computing architecture supported by RDBMS products is client/server. A client/server architecture permits application-specific code to be separated from the RDBMS code and data, possible across multiple client platforms. Support for remote stored procedures permits database processing to be shared across multiple server platforms, but not necessarily within the context of a single transaction nor with each server having access to the same data.

RDBMS distributed database support requires that each user or application perceive multiple databases as a single logical database, regardless of physical location. Database authorities have proposed different ways of defining distributed database support. Chris Date has identified twelve objectives for distributed database support. These are local autonomy, no reliance on a central site, continuous operation, location independence, fragmentation and replication independence,

distributed query processing, distributed transaction management, hardware independence, operating system independence, network independence, and DBMS independence. Similarly, Mike Stonebraker has identified seven rules based on transparency. These are update, retrieval, scheme, performance, transaction, copy, and tool transparency.

IV. The Right Stuff

A. What strategies does the RDBMS support?

Once an enterprise IS organization has chosen a migration strategy, success will often be dictated by how appropriate the tools are to the job. The RDBMS is an especially important tool, and it should clearly support the selected strategy. If the wrong RDBMS is currently in use, it may be worthwhile making a transition to a better suited product early on, particularly if there are strong business or financial reasons for adopting the strategy. Determining if an RDBMS provides adequate support for a particular migration strategy can be difficult. Few RDBMS vendors understand the issues and most follow no coherent statement of direction in this regard.

Regardless of migration strategy, an RDBMS must provide a complete set of application development tools, be reliable, provide scalability, be cost effective, and provide low-cost extensibility through open systems support. If it is to support either a direct replacement or a staged migration strategy, the primary issue will be potential return-on-investment. For a staged migration strategy, modularity will be an additional key requirement. This means that features such as import/export utilities, performance, and availability will be issues. For an anticipated migration strategy, the key issues are reliability and openness since the application designer must be able to anticipate the application requirements which the RDBMS will be able to support or else be able to extend the RDBMS to support these requirements. Support for standards becomes important in terms of predictable behavior of the RDBMS. Both competitive migration and legacy application extinction strategies demand that the RDBMS have an extremely flexible architecture and meet the reliability, scalability, cost effectiveness, and open systems requirements described above.

B. Some questions to ask

Of all the strategies described here, the most demanding is productive co-existence. The RDBMS must support not only migration, but integration and interoperation as well. Here are a few questions to ask RDBMS vendors when selecting an RDBMS for this purpose.

- How true to the relational model is the RDBMS? Does it simultaneously support transaction, batch, ad-hoc, and decision support workloads?

- Does the vendor have a migration, integration, and interoperability strategy? If so, what is it and how do the vendor's products and product plans complement this strategy?

- What is the vendor's understanding of migration problems? Does the vendor's product support more than one migration strategy?

- What features does the vendor say support migration? Are application development tools included? What direct experience has the vendor had with migration efforts?

- Is the RDBMS designed to make migration easy (i.e., low cost), without sacrificing response time, throughput, data integrity, and transaction support?

- Are standards supported throughout the product and, if so, what is the impact on vendor innovation and product ease-of-use?

- Does the RDBMS provide a means to access or convert legacy data to relational form? Are CASE (especially data modeling) tools available?

- How well can the RDBMS integrate with non-relational applications? Are TP monitors and gateways supported for this purpose?

- How reliable is the RDBMS? Are both performance and availability predictable? Is integrity strongly enforceable?

- How scalable (both horizontally and vertically) is the RDBMS? Does it provide both distributed computing and distributed database support? Is it portable?

- Can the RDBMS interoperate with other vendor's relational and non-relational products? How open or extensible is the RDBMS and at what functional and financial costs?

- Finally, is the RDBMS cost-effective? Can it really offer significant short-term and long-term return-on-investment?

V. Conclusions

In this paper we have reviewed the various strategies for migrating legacy applications to new technology. An alternative to pure migration has been introduced, that of productive co-existence, and have argued that this strategy has much to offer. Finally, we have examined the critical role played by the RDBMS in migration efforts.

Productive co-existence, as a superset of the various migration strategies, may well be the only reasonable approach to solving the obsolescence of applications. At the very least, it does not assume that every legacy application needs to be replaced in order to take advantage of newer technology. Ideally, it extends the window of opportunity to recapture past investments by expanding the functionality of legacy applications at a low cost.

Of course, productive co-existence does place a heavy burden on RDBMS functionality. The RDBMS becomes the center of migration, integration, and interoperability efforts. As such, it cannot be less than the very best product available for the job. Chosen incorrectly, the RDBMS can lead to weakening and eventual disarray of the entire IS infrastructure. Chosen correctly, the RDBMS will enable and strengthen efforts to move into the next century of computing.

References:

D. McGoveran, "An Overview of Migration Strategies," c. July, 1992, Alternative Technologies, Boulder Creek, CA.

D. McGoveran, "Application Migration and Portabilty," Database and Application Development in the '90s Conference Proceedings, October 1-4, 1991, Database Associates, Chicago, IL.

D. McGoveran, "ORACLE7 Database Evaluation Report," Database Product Evaluation Report Series, c. June, 1992, Alternative Technologies, Boulder Creek, CA.

