Understanding the Business Aspects of Software Migration

> **An IT shop considering migration to a new architecture can be making a "bet your business" decision**.

To many IT shops faced with replacing an old platform, the idea of application migration equates to the conversion of their existing software. Their primary concerns tend to be technical ones pertaining to the difficulty of moving software and data and the compatibility of old and new hardware. But application migration is much more than simply moving code and data; it is an investment in a company's future. Moving applications from one platform to another represents a decision which impacts a company's bottom-line. For a small enterprise, switching platforms may actually make or break the company. For larger corporations, migrating without adequate research and planning may cost millions in lost revenues and result in lost jobs. Before a single line of code is converted, an organization needs to examine its business and the role its software applications play within it. Once the business requirements of an enterprise are thoroughly understood, a migration strategy can be developed which meets present and future needs.

Before discussing migrations and conversions, we need to clearly define some terms.

- **Conversion:**   Rewriting an application in a new programming language.

- **Migration:**   Transfer of applications between dissimilar operating systems.

- **Port:**   Transfer of application source code and command procedures with minimal modification between similar operating systems.

Part of developing a migration strategy is the selection of a migration solution. There is no single best way to move from one platform to another. A company must

> **There is no single best way to move from one platform to another.**

consider its needs and resources when planning a migration and then choose the options that are the best fit. Migrations often involve a portfolio of software applications and different applications may require different approaches to ensure overall success for the project. In some cases, a single option may fulfill the requirements of a migration plan. In others, use of multiple options may be required to adequately migrate applications and keep the company running. In all cases, there is no substitute for adequate research into the needs of the business and careful planning of a strategy to fulfill those needs.

There are seven basic options to consider when planning a migration. These options are *virtualization, package replacement, new development, re-engineering, coexistence, conversion*, and *do nothing*. Any single migration option can be used for the entire project, or the options can be mixed to achieve the best results. Before choosing one or more options, a careful analysis of present and anticipated business needs should be carried out and weighed against the pros and cons inherent in each option. This will help measure the relative effectiveness of each migration alternative.

Now, let's take a closer look at the trade-offs involved with each migration option.

Virtualization

This approach consists of replacing legacy hardware with virtual hardware. Essential a box swap,

> **Plus:** *Simple and quick. Existing applications remain intact and unchanged.*
>
> **Minus:** *Existing applications remain intact and unchanged*

virtual hardware improves system reliability without impacting end users. Virtual hardware can be a quick fix to ailing hardware and provides a long term solution to support critical legacy applications.

The primary downside to virtualization is it does not modernize the software applications. Any support or stability issues with the applications will remain. If modernizing applications is a primary, then hardware virtualization will not be effective. Hardware virtualization's strong suite is improving reliability at a reasonable cost.

Package Replacement

> **Plus:** *Minimal development effort.*
>
> **Minus:** *Limits functionality. Offers no competitive advantage.*

This approach consists of replacing existing software applications with

pre-packaged commercial applications; in other words, buying off-the-shelf. Package replacement is the simplest and often the best solution. It is quick, provides standardized applications, and reduces overhead by relieving the IT staff of conversion and maintenance responsibilities. When properly implemented, replacement of applications with packaged solutions reduces risk and migration effort.

The primary risk when implementing a packaged solution is that the packages selected may not adequately meet company needs. In this scenario, a packaged solution may cause more problems than it solves, wasting money and resources and emphasizing the need for proper business analysis before selecting a solution. Furthermore, packaged solutions can also be costly, so cost versus return must be taken into consideration before selecting a packaged solution. Packaged software tends to be limited in functionality, as its supplier must compromise between providing a wide range of functions and the cost of maintaining the software. Packaged solutions also make the company dependent upon a third party for support and updates, so research must be conducted to ensure that the supplier is reliable. Packaged solutions generally require end-user and IT staff retraining and a conversion of data from the old system to the new one.

A final point to consider is that a popular packaged solution may be used by many of a company's competitors. Using the same software does not give a company a competitive advantage. At best, it simply levels the playing field.

New Development

> **Plus:** *Addresses current and future needs.*
>
> **Minus:** *Expensive and time consuming.*

New development consists of throwing out the old applications

and starting with a fresh slate. It requires a thorough analysis of present and future business needs followed by the development of new software applications. The new applications can be designed and built based upon present and future business strategies and new operating procedures, not upon existing applications and operating procedures.

New development allows a company to take full advantage of modern hardware, software, programming tools, and programming techniques. It addresses current and future needs

(provided it is properly planned), allowing construction of software systems which help an enterprise and can grow with it, rather than forcing an organization to conform to the restrictions of its software applications. Development of new applications allows a company to take advantage of the knowledge and experience it has acquired over the years. Well planned and executed development of new software can put an organization on the leading edge of information technology and may give it an advantage over its competitors.

Unfortunately, new development is often the most expensive migration option. It is time consuming, requiring extensive studies, planning, and preparation. Key to successful implementation of a new development strategy is accurate anticipation of future needs. If software is developed for needs which do not emerge, valuable time and resources will have been wasted. New development requires that IT staff and end-users be fully retrained on the new system. Finally, few companies have the luxury of simply stopping production while they develop new software systems. Extra staff and resources may be necessary to develop new applications, further driving up migration costs.

Re-engineering

> **Plus:** *Reduces development time while taking advantage of new hardware and software.*
> **Minus:** *May inherit limitations and errors from old applications.*

Application re-engineering uses existing applications as templates to design and engineer replacements. It represents a compromise between converting existing applications as they stand and replacing them with entirely new applications. Application re-engineering reduces the time spent in software planning and development as the old application serves as a template for the new. Re-engineering does not reduce the time spent in a business needs analysis. A full needs analysis is always necessary to determine which migration option best suits the requirements of the organization.

Application re-engineering allows an application to be redeveloped to take advantage of the new computer system's features and functions. It permits improvements to the application as it is redeveloped, often allowing IT shops to eliminate some of the backlog of user requests. It reduces development time by re-using code from the old application wherever possible. Re-engineered applications can reduce end-user retraining requirements by redeveloping the applications with user interfaces similar to those of the old applications.

On the downside, application re-engineering can be expensive and time consuming. Like new development, it requires time for planning and analysis. Also like new development, it may require additional resources to maintain existing software while the replacement applications are developed. Re-engineering using existing applications as a template presents the risk of inheriting limitations in design and functionality in the new applications. Re-using old code in the new applications may also transfer or introduce errors into the new code.

An area that is often overlooked when re-engineering an application is testing. Each modification and enhancement made to the migrated application may require additional time for testing on the new system. Substantial modifications to an application may drive up the cost of the re-engineering effort by making testing more difficult and time consuming.

Coexistence

There may be one or more applications on the current platform that, for a variety of reasons, are not worth moving to

> **Plus:** *Retains investment in applications and reduces impact on operations.*
> **Minus:** *Problems with old applications will remain in converted applications.*

the new system. A particular application may be near the end of its useful life, or it may only be used by a single department that has marginal need to integrate with the rest of the corporate portfolio of applications. In these cases, it may make sense to leave an application where it is or move it to a smaller model of the current computer system and continue to support it.

A coexistence solution has the advantage of requiring little or no migration effort. It does require the company to devise a means of sharing information between the old system and the new one. It also has the disadvantage of forcing the company to maintain two platforms and two operating systems, which adds to overhead costs. In cases where an organization is replacing outdated equipment, coexistence is often used as a short-term solution, providing more time and flexibility in planning and implementing a migration strategy. In cases where a company requires a wide variety of applications which are not all available on a single platform, coexistence can make good sense.

Conversion

Conversion of an application is the physical movement of code and

> **Plus:** *Little or no migration effort.*
> **Minus:** *Must support two platforms.*

data to a new platform with minimal change. Conversion has the advantage of retaining the investment in current applications and minimizes end-user and IT staff retraining. Depending upon the compatibility of the chosen solution, conversion can be a quick and cost effective means of moving to a new platform.

Conversion can also be very messy and has the distinct disadvantage of allowing past sins to revisit migrated applications. If the code being converted is not clean, modular, structured, and documented, it will remain so on the new system, retaining existing problems and shortcomings. The old adage of "garbage in, garbage out," definitely applies to code conversion. Older applications are often designed to take advantage of a specific operating system and architecture. Transferring the code to a new system with a different architecture often results in a degradation of performance. Conversion can also be very expensive. If an application is closely tied to the source architecture or the target system differs significantly from the source system, code conversion efforts can require substantial manual effort. This results in increased migration time and cost and a higher risk of errors in the converted code.

Code conversion is a solution that needs to be carefully considered within the context of the needs of the business. Conversion of applications which do not meet current needs or which restrict the ability of an enterprise to expand and change represent a bad investment of time, money, and resources.

Do Nothing

The final option to consider in migration is the "do nothing"

> **Plus:** No cost, no pain.
> **Minus:** May lose competitive edge, staff, and incentives.

option. Continue to operate on the existing platform. Migrating to a new platform simply to gain access to the latest and greatest software applications and hardware technology is a luxury few companies can afford. If it makes sense to stay with the existing platform for another year or two, do so. Unless support costs are exorbitant, this solution will be the least expensive.

However, failing to modernize applications may result in the loss of a company's competitive edge. It may also result in the loss of key IT staff who regard the company as a dead end to their careers. Many hardware manufacturers offer incentives to move from an old platform to a new one. These incentives do not last forever and failure to move now may result in a higher migration cost later.

Conclusion

Any of the above

> *A full needs analysis should always be carried out to determine which migration options best suit the requirements of a business.*

options can be considered when deciding whether to migrate an application. However, only a thorough analysis of business needs will determine which option or combination of options will best achieve a successful migration solution. A company must understand the role its software and computer systems play within the context of its business practices and strategic direction before it can determine whether a migration is necessary. It must determine the goals of a migration before developing a plan to achieve them. Most companies will elect to migrate to new platforms and applications because their existing ones restrict their organizations in some way. Only careful planning will ensure that the new solutions do not impose new restrictions or carry over old ones. Migrating to a new platform can make or break a company and the most successful migration solution is the one that helps the bottom line while paving the road to future growth and expansion.

About the author:

Mr. Bruce Claremont has a degree in Computer Science and has been delivering migration solutions since 1983. Mr. Claremont has extensive programming, project management, and system management experience. He has worked all sides of the fence, as a customer, software engineer, system manager, delivery specialist, project manager, and business owner. He founded Migration Specialties in 1992. Migration Specialties delivers OpenVMS, Tru64 UNIX, software migration, and VAX and Alpha hardware emulation services. More information about Migration Specialties products and services can be found a www.MigrationSpecialties.com.