APPLICATIONS

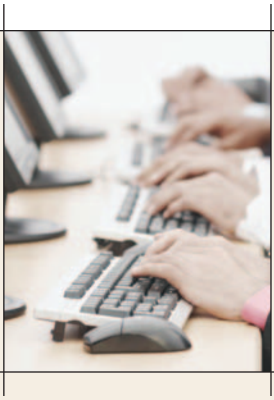
A WHITE PAPER SERIES

MANY ORGANIZATIONS ARE ADOPTING MIGRATION AS A STRATEGY TO RESOLVE THEIR BUSINESS AND IT CHALLENGES. THE GOAL OF THIS WHITE PAPER IS TO PROVIDE BUSINESSES WITH THE SIX KEY STEPS TO SUCCEED WHEN EMBARKING ON A MIGRATION PROJECT.

Six Steps to Migration Project Success

SYNTEL

Consider IT Done®



Six Steps to Migration Project Success

Today's organizations are faced with many challenges in their business environments. Legacy, outdated information technology (IT) systems and changing business processes are among these major challenges as companies address legacy system inflexibility, agility, lack of scalability, lack of wider data access, shortage of skills, high cost of maintenance and unreliability. Couple this with continually changing technologies, and organizations are faced with the need to assess these new technologies and adapt their infrastructures and applications to leverage those technologies.

Migration enhances and protects the long-term investment in an organization's software infrastructure and effectively reduces the total cost of ownership (TCO).

1.

EXECUTIVE SUMMARY

In the quest to address these issues and drive business forward, enterprises are proactively seeking strategies to insure that success. To that end, many organizations are adopting migration as a strategy to resolve their business and IT challenges. However, while they are sold on the idea of migration for its significant benefits, enterprises are uncertain how to begin. The goal of this white paper is to provide businesses with the understanding to select a migration strategy as well as six key steps to succeed when embarking on a migration project. **The six steps are:**

- **Application assessment**
- **Application preparation**
- **Application migration**
- **Post-migration changes**
- **Application testing**
- **Post-migration support**

The intended audience of this white paper is CIOs, IT executives, system integrators, and others who are considering migration as a solution to their business challenges.

2.

WHY MIGRATE?

Prior to arriving at the decision to migrate, the initial approach to a successful project is to identify why the process should be considered at all. This section discusses why a business should consider migration.

Older IT systems—legacy systems such as mainframes and COBOL-based software—are difficult to integrate with newer technologies. Complex mainframe applications are unable to keep pace with, and respond to, today's dynamic business demands. **This presents several issues, including:**

- **Increased maintenance costs of current IT environment**
- **Limited resources with specialized skill sets**
- **Lack of interoperability among disparate systems**
- **Lack of scalability and functionality in the current system**
- **Lack of agility to incorporate new features rapidly**

Migration sets the stage as a beneficial opportunity to move these old mission-critical applications into the 21st century and resolve these issues.

MIGRATION ENHANCES AND PROTECTS BUSINESS INVESTMENTS

Migration enhances and protects the long-term investment in an organization's software infrastructure and effectively reduces the total cost of ownership (TCO). Performance benchmarking ensures that migrated applications perform better than the old applications, resulting in increased scalability, simplified maintenance, and reduced costs. Development efforts are improved through migration-specific tool creation and use, achieving faster time to market.

MIGRATION IMPROVES APPLICATIONS

Migration enables the rejuvenation of existing business systems and leverages application use, offering opportunities that current and future technologies provide. For example, by adopting new service-oriented solutions such as Service Oriented Architecture (SOA), the interfaces of current applications can be updated. The business processes within the applications are represented as reusable components using standards like Web services. New applications can utilize these services, thereby extending their life.

TYPES OF LANGUAGE/CODE MIGRATIONS

- VB OR ASP TO VB.NET
- VB TO C# .NET
- C OR C++ TO .NET
- C OR C++ TO J2EE
- VB/ASP TO J2EE
- POWERBUILDER TO J2EE
- POWERBUILDER TO .NET
- COBOL TO .NET
- COBOL TO J2EE
- RPG TO .NET
- RPG TO J2EE
- DELPHI TO C#

TYPES OF OPERATING SYSTEM MIGRATIONS

- DOS TO WINDOWS
- UNIX (AIX, SOLARIS, HP-UX) TO WINDOWS
- WINDOWS TO LINUX
- UNIX TO LINUX
- DG UNIX TO IBM AIX
- LEGACY MAINFRAME TO UNIX
- MAINFRAME TO WINDOWS
- C TO C TO UNISYS, UNIX, INFORMIX 4GL

TYPES OF USER INTERFACE MIGRATIONS

- LEGACY CHARACTER-BASED UI TO GRAPHICAL USER INTERFACE (GUI)
- EX-WINDOWS-BASED UI ON UNIX MACHINES TO WINDOWS-BASED UI

Today, Web enabling of legacy applications and data is the goal of many companies, who look forward to leveraging their legacy data assets through migration to a web-based communications model. Web-enabling an application or augmenting an existing Web application provides advanced features such as state management, data access, traceability, and improved application performance.

Migration also enhances the new operating environment. Software assets are consolidated and applications are brought onto a single development platform, resulting in improved system integration, performance, scalability, productivity, integration, reliability, security, and extensibility. In addition, reduced cost overheads are realized by eliminating the need to maintain multiple platforms.

3.

DETERMINE A MIGRATION STRATEGY

WHAT CAN BE MIGRATED?

Migration can be instituted across of range of technology classes, including the following *(see sidebars)*:

- **Language or code migrations**
- **Operating system migrations**
- **Data migrations**
- **User interface (UI) migrations**
- **Architecture migrations, including migration to object-oriented programming (OOP)**

As a result, enterprises can perform migrations whenever a greatly improved infrastructure is desired, including programming languages, operating systems, data, architecture, or any combination of these.

MIGRATION STRATEGIES TO CONSIDER

The next step is to consider the strategies available for achieving migration. We will take a look at complete migration, iterative migration, limited migration, vertical migration, and horizontal migration. Organizations determine which of these strategies to use based on factors such as system qualities, manageability, training, and cost.[1]

Complete migration

In complete migration, all of the components in the application are migrated as a whole. This strategy does not allow for intermediate validation of the migration process or the business rules. The only way to know if the application is suitable for the business requirements is to evaluate it once the migration is completed. Complete migration requires significant effort and is potentially expensive and risky.

This strategy does enable immediate enhancement with new and added functionality. For example, VB 6.0 application functionality is improved when migrated to the .NET environment. Directly integrated into their new .NET environment, these applications gain new versatility, and if deployed on new hardware or new Windows operating systems, will not require further migration or recompilation.

Complete migration is an expensive strategy, but is also the most desirable strategy to adopt since it provides application adaptability to future needs. However, a comprehensive assessment of the application is necessary prior to opting for complete migration; this is discussed in section 4 of this paper.

Iterative migration

An iterative migration strategy allows for a more controlled migration process since the application is migrated component-by-component, with each newly migrated component rolled out as a phase. This strategy is only feasible if the existing application is composed of distinct components. Interoperability techniques are key, as the migrated and un-migrated components must function together. Iterative migrations provide an acceptable alternative to a complete migration, and are often the option for a large-scale legacy application migration.

With this strategy, there is greater control over the cost and progress of the entire migration project. Each phased rollout, or iteration, minimizes risk since the application is returned to a stable production-quality state. Iterative migration also allows the flexibility to migrate only certain portions of the application that have immediate relevance to the business. Improved performance and scalability are immediately realized in the migrated components.

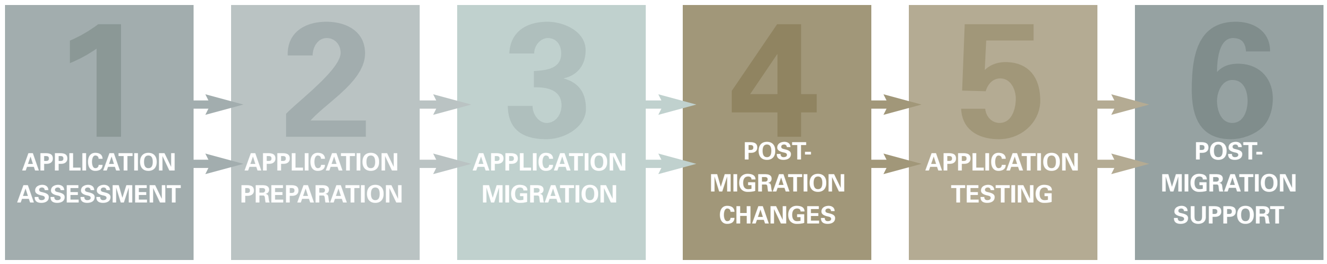
Limited migration

Limited migration is different from the iterative migration process in that only a component of the application is migrated. The migrated portion is then modified to interoperate with the un-migrated part of the application. Interoperability is the key issue in this type of migration. A business may not need to migrate the entire application, so this type of migration allows an organization to port only the components of the application that are actually required.

Vertical migration

Vertical migration differs from the other migrations in that the process is performed tier-by-tier. Vertical

Figure 1: The Six-Step Migration Process



migration involves isolating and replacing a portion of an application through all n-tiers. The developer determines which component of an application has the least interaction with the other components and performs the migration. The migration is then completed on all tiers for a particular module prior to proceeding to the next module.

This strategy is advantageous when portions of an application are well isolated from other portions of the same application. In these instances, the isolated components share little state information with the rest of the application and can undergo easy migration with minimal impact on the rest of the system.

Vertical migration is also an effective option when ActiveX Data Objects (ADO) recordsets are used between tiers. Many applications pass disconnected ADO record sets from the data and business tiers to the presentation tier. They then iterate through the recordsets and generate HTML tables. This type of application is well suited to a vertical migration because migrating vertically minimizes the work involved in achieving interoperability with ADO.

Horizontal migration

Horizontal migration involves replacing an entire tier of an application without immediately migrating the other tiers. For example, a developer may choose to initially replace the ASP code within a Web-based presentation tier or replace the COM code within the middle tier as the initial migration step. For a business migrating to the .NET environment, the migration is performed a single tier at a time, taking advantage of the features of the .NET framework specific to a particular tier. In this instance, no application code is modified and no operations are affected on another application tier.

Horizontal migration is beneficial for infrastructures containing large numbers of servers, large amounts of shared code, heavy ASP application or session state use, and complex middle tiers.

4.

A MIGRATION ROADMAP

MIGRATION: A SIX-STEP PROCESS

The actual migration process is divided into six distinct phases: application assessment, application preparation, application migration, post-migration changes, and application testing ***(see Figure 1***).

STEP 1: ASSESS THE APPLICATION FOR MIGRATION

Once a business has determined that a migration is a viable solution, the first step is to ascertain which current applications continue to fulfill current business needs. Companies that omit this assessment process delay the inevitable need to retain or eliminate certain applications. This prolongs the duration of a project and reduces many of the key benefits of migration, such as improved and more efficient business processes.

In determining the feasibility of current applications, businesses need to examine several decision drivers. These drivers include project priorities and goals, application business value, development environment and resource skills, application complexity and architecture, and quality assurance. **To determine the application's value to business, the following queries need to be answered:**

- **What functionality does the application possess that other applications or third-party tools cannot reproduce?**
- **What types of data and data transmission protocols does the application support?**
- **What are the application's basic input and output types, different interface points, and external dependencies?**

TYPES OF ARCHITECTURE MIGRATIONS

- LEGACY TO WEB ENABLEMENT
- CLIENT SERVER TO N-TIER
- CLIENT SERVER TO WEB SERVICES
- CLIENT SERVER TO SOA
- MIGRATIONS STRUCTURED TO OOPS (OBJECT-ORIENTED PROGRAMMING, SYSTEMS)

TYPES OF DATA MIGRATIONS

- SYBASE TO ORACLE
- SYBASE TO MS SQL SERVER 2000
- MS SQL SERVER TO ORACLE
- DB2 TO MS SQL SERVER
- DB2 TO ORACLE
- LEGACY FILE-BASED SYSTEM TO DB2
- SQL SERVER 6.5 TO SQL SERVER 2000

> *For most organizations today, waiting to create a migration strategy is not a viable option. Current business and IT challenges—outdated IT systems and changing business processes—require proactive resolution.*

- **Does the application handle legacy file formats or high value business transactions?**
- **How would removal of the application impact the organization?**
- **What is the current TCO for the application? Would TCO improve if the application was ported to the new environment?**

The next step in the assessment establishes the application's code quality in terms of design and source code. This step helps the migration team understand the code complexity and, in turn, helps determine the cost, effort, and schedule for migration. In addition to assessing code quality, other metrics for this stage of the assessment include examining the development environment and developer skill sets.

To determine the application's code quality, the migration team establishes the application's size, usage, complexity, dependencies, and overall stability. **The following are important queries to consider:**

- **What is the application's size? How many lines of code, forms, user controls, modules, classes, and data source types exist?**
- **What are the application's functions, properties, and types?**
- **How complex is the application? What application features are not supported, resulting in a potential manual migration?**
- **Does the application depend on other tools or use an internal mapping that generates internal functional dependencies?**
- **Is the application currently undergoing enhancements or code changes?**

The final parameters in the application assessment examine the development team's skills and the development environment. This helps determine if the migration adheres to estimates in terms of effort and schedule. Familiarity with the code base and new environment are essential to performing a successful migration.

In the event the migration team is unfamiliar with the application, the extra time that would be required to gain familiarity with the code base needs to be factored in during estimation. Any lack of code level knowledge, insufficient documentation, and lack of development skills in the new environment increase the risks involved in a migration.

STEP 2: PREPARE THE APPLICATION FOR MIGRATION

Once the assessment is completed, the second step prepares the application for migration. **Three main initial conditions are required before migration commences:**

- **Provide all the relevant application documents and baselined source code to the migration team**
- **Supply functional experts to the migration team for accurate understanding of the project**
- **Provide application source code that has not undergone separate enhancements as the migration begins**

With these conditions met, the application preparation phase begins. The entire application is rebuilt in the parent environment from the given source code. The migration team then executes it and runs the application test cases to determine that the application source code provided is the correct version for migration. With the correct source code version established, developers proceed to the migration.

STEP 3: MIGRATE THE APPLICATION

In the actual migration process, the prepared application is migrated to the new environment using migration tools developed specifically for this purpose. Migration tools provide many benefits, among them supplying the migration team with the ability to:

- **Consider resource-consuming elements, constructs, and features**
- **Identify incompatible porting issues in the application's code, build, and production environment**
- **Remove dead code and obtain recommendations for improved coding style**
- **Analyze application components and component relationships**
- **Shorten migration timeframes by eliminating manual rewriting of unsupported code**
- **Simplify the migration process via migration wizards**

With the use of a migration tool, an upgrade report is generated which will identify what application features are not upgraded automatically; these result in the need for manual migrations. This is accomplished in the next phase of the migration process.

STEP 4: PERFORM POST-MIGRATION CHANGES

Some applications simply cannot be migrated automatically and may require significant manual work. Based on the report findings from the prior step and the desired functionality in the new environment, the developer will need to change the code in the new environment. The objective is to write code for the new platform to obtain the same functionality in the migrated code as found in the original application.

Common situations that necessitate manual work are:

- **Applications unable to update to their most current version prior to migration (e.g., VB 5.0 or earlier applications upgraded to VB 6.0 before migration to VB.NET)**
- **Certain features not upgraded automatically (e.g., DDE, OLE, DAO, and RDO data bindings in VB 6.0)**
- **Distributed n-tier applications with several layers of objects communicating through COM**
- **Web applications using DHTML, Web Classes, or ActiveX documents**
- **Projects using ActiveX controls or ActiveX DLLs**

STEP 5: TEST THE APPLICATION

In this final phase, the newly migrated application is subjected to rigorous testing using the same test cases applied earlier when validating the source code provided. Apart from functional testing, stress, volume, and load tests are carried out to ensure scalability and performance levels are achieved. Fine tuning and/or optimization is conducted after each round of testing to achieve the desired performance levels. Following successful testing, the application is released for production. Alternatively, further enhancements are executed based on the business needs for the application.

STEP 6: POST-MIGRATION SUPPORT

Once the migrated application is deployed, additional business and user needs may be identified, requiring technical team, developer, and/or system support intervention. Issues may involve system configuration or optimization, or the application configuration parameters may require fine tuning. Ensuring close attention to this final phase and leveraging the knowledge capital of migration team members during this phase reduces risks during the field testing. It is an essential step to ensure mission critical applications operate smoothly.

5.

MIGRATION BEST PRACTICE CHECKLIST

The migration best practice checklist below provides an at-a-glance overview of the steps required to launch and drive a migration initiative.

The Migration Best Practice Checklist

❑ Establish costs and benefits of the migration in advance

❑ Evaluate current resources

❑ Define the scope of the migration

❑ Start with simple projects initially

❑ Use applications best suited for the current operating environment “as is”

❑ Understand how the application is going to be modified

❑ Analyze the current application

❑ Ensure a complete understanding of the migration tool that will be used

❑ Prepare the code being migrated prior to the actual migration

❑ Upgrade module-by-module

❑ Test each module as it is being upgraded before continuing migration

❑ Review the upgrade report generated by the migration tool

❑ Use stored procedures as much as possible

❑ Use tools like source code analyzers and compatibility test tools to identity issues in advance

❑ Define technical changes needed due to migration

❑ Write new test cases for the migrated applications to gauge existing functionality along with performance and scalability tests

6.

CONCLUSION

For most organizations today, waiting to create a migration strategy is not a viable option. Current business and IT challenges—outdated IT systems and changing business processes—require proactive resolution. By empowering the rejuvenation of existing business systems and application uses, migration offers opportunities that both current and future technologies provide. Companies that carefully embrace and incorporate the six strategic steps outlined in this paper leverage the power of migration and drive the changes that equate to business success, now and in the future.

References

[1] Mainframe Migration Alliance: Top 5 reasons cited by mainframe customers to MMA, www.mainframemigration.org.

about SYNTEL:

Syntel provides custom outsourcing solutions to Global 2000 corporations. Founded in 1980, Syntel's portfolio of services includes BPO, complex application development, management, product engineering, and enterprise application integration services, as well as e-Business development and integration, wireless solutions, data warehousing, CRM, and ERP.

We maximize outsourcing investments through an onsite/off-shore Global Delivery Service, increasing the efficiency of how complex projects are delivered. Syntel's global approach also makes a significant and positive impact on speed-to-market, budgets, and quality. We deploy a custom delivery model that is a seamless extension of your organization to fit your business goals and a proprietary knowledge transfer methodology to guarantee knowledge continuity.

SYNTEL
525 E. Big Beaver, Third Floor
Troy, MI 48083
phone **248.619.3503**
info@syntelinc.com

visit Syntel's web site at www.syntelinc.com

Consider IT Done®