Is It Time to Migrate from the Mainframe?

Your mainframe system had a lot to do with the success of your organization. It provided breakthroughs in processing speed and improvements in handling and reporting transactions, tracking inventory, generating invoices, processing payments, and supporting your customer relations. But the world has changed, demanding fast response to changing conditions at lower and lower cost. You’re finding that your mainframe environment is now getting in the way of your organization’s goals. It’s time to think about another migration.

It’s unlikely that the systems you are using today are the ones that your organization used when it first introduced automation, and it’s helpful to think about why the previous migrations were undertaken. Those reasons that led to migration in the past are the same as the ones that will lead you to move off the mainframe in the near future.

As legacy systems reach end-of-life, they do less and less to help grow the business while costing more and more. They become unaffordable, not just in terms of how big a slice of the budget they require, but because they divert funds from new initiatives essential to business competitiveness. Moreover, they begin to risk the health of the business in a number of other ways as well:

- **Your legacy system can’t be made responsive to the changing business environment.**
  The majority of legacy systems have been changed beyond the point where anyone on the technical staff really understands all the details of how they work. Any change must be preceded by an extensive analysis, and it’s common to find that changes affect not only the primary software, but also the various middleware and front-end components that have been added to try to satisfy previous changes. It’s not that the current system cannot be changed, but it can’t be changed in the time and at the cost that makes the change viable for your organization. What’s worse, each change makes the system increasingly brittle.

- **The cost of caring for your mainframe is constraining your development budget.**
  No IT organization wants to sit still, but no IT organization has infinite budget. When push comes to shove, keeping the current systems going will take priority over new developments that can generate internal efficiencies or competitive advantages. You’re in a situation where the inevitable migration is simply being delayed as the cost of maintenance rises.

---

Legacy Migration to Microsoft® .NET

Is this the time that you should start decommissioning your mainframe-based applications and migrate them to the Microsoft .NET environment? For many IT executives, this question raises concerns about the costs, benefits and risks of moving from a familiar environment, even when it’s clear that the systems that were so valuable to the organization in the past are now functioning as obstacles to progress.

This paper looks at common migration concerns, and it shows how tools and processes available today make migration to .NET a viable choice for the majority of today’s mainframe applications.
The skills required to maintain your mainframe software are becoming scarce. If you are using mainframe COBOL or RPG with IDMS data files, you’ve probably noticed the “deer in the headlights” look in the faces of prospective employees. Legacy mainframe skills simply aren’t being taught anymore.

Preserving Your Investment
Even with the pressures of cost, functionality, and staff skills, you may be reluctant to leave your mainframe environment because of the risks you perceive. The path of design recovery, specification and reprogramming in a new language for a new environment is littered with cost overruns, functional breakdowns and more than a handful of total failures. Many organizations decide that their only rational choice is to pick a packaged application and then try to either have it modified or to change their business processes so the system can provide support. The risk of business disruption is real, and it can be frightening to think about it.

There is a better alternative - Legacy Migration.

Legacy Migration takes your current code and prepares it to run in a cost-effective, extensible environment. This approach:
- Preserves your investment in the logic that supports your organizational practices.
- Bypasses the expense and risk of requirements recovery from your current code.
- Retains the value of your staff’s knowledge and skill, while at the same time providing an opportunity for them to learn and apply today’s advanced techniques using powerful visual tools.
- Breaks the technology constraints of the mainframe environment with its specialized interfaces. Mainframes and their system software were never designed to operate in an interconnected world.
- Lowers the costs of computing for technology, support services and staff.
- Shrinks the time, cost and risk of moving from the mainframe to a distributed, “internet-conscious” environment.
- Lets you take advantage of the new tools and capabilities of a modern environment at a pace that can be coordinated with your organization’s capacity for change. You don’t get "thrown in the deep end."

Legacy Migration should be at the top of your list of strategies for moving your organization’s IT forward if:
- Your systems have been designed and enhanced to support your organization’s specific operating practices.
- You are pleased with the functionality you have today, but are concerned about your ability to move forward because of obsolescence, cost and enhancement constraints associated with your mainframe environment.

Why Migrate to .NET?
What could seem further apart than your mainframe and Microsoft Windows? Actually, they aren’t that far apart at all. Servers using Intel processors can provide up to 1000 MIPS of computing power. The industry standard TPC-C benchmark shows that Windows-based servers can process almost 800,000 order entry transactions per minute while performing background tasks, and the top ten performance/cost leaders tested by the TPC all use Microsoft Windows 2000 and Microsoft SQL Server. There are companies that run all their data processing on Windows servers, including one with $40B annual sales.

Microsoft developed the .NET environment to be connectable and scalable. Your mainframe wasn’t designed to be either. At its heart, the extensibility of .NET breaks away from the primary feature of mainframe environments - scarcity. Mainframe operating systems, accounting and support tools are all built with the underlying belief that cycles are scarce. As a result, prodigious effort is spent on tuning and balancing the operational partitions of the system as it goes through its daily work, and all resources are carefully accounted for and charged.

In the mainframe world, adding processors, memory or storage is a significant undertaking. In the .NET world, change is accepted as a normal condition, and the processes of adding to, subtracting from or reorganizing the IT environment are straightforward.

Another reason to consider .NET is cost. Mainframe technology resources typically price out at one to two levels of magnitude more than server/Wintel technology of equal capacity. Xephon’s December 2002 study, “Inside the Mainframe Processor Market” provides the following information:

"Mainframe processors deliver a level of profitability unheard of in any other manufacturing industry - gross profit margins of anything from 50% to 90% are typical for IBM in this area. The price the user pays is determined not by normal supply and demand issues, or even by the existence of competition, but by other factors unique to this industry."

In the mainframe world, processor cost leads the costs of operating system and support software licenses as well, with the effect that mainframe users pay premium prices for everything.

Microsoft’s .NET is an evolutionary step into the future. You don’t need to overhaul your existing servers, desktops and networks to adopt it. .NET-connected software is built on Web service standards, which enables both new and existing applications to connect with software and services across platforms, applications, and programming languages. In applications such as Microsoft Excel,
.NET-connected software allows analysts to monitor up-to-the-minute production tallies across numerous suppliers, enabling everyone in the supply chain to effectively tailor production to match demand. For example, Newport News Shipbuilding used .NET-connected software to build applications faster, and to connect with various partners. The company improved its time to market by 19 percent.

Microsoft Visual Studio®, .NET and the Microsoft .NET Framework empower developers to quickly and easily create cutting-edge Web services and applications, building on their existing skill sets. Through support for multiple programming languages, developers are free to use the programming language of their choice in building Web services. Seamless deployment, in addition to the ability to use existing Web services, presents substantial savings opportunities for the corporate IT department.

In addition to improving developer productivity, Visual Studio .NET helps alleviate one of the greatest scarcities in the world: skilled programmers. Applying rapid application development techniques to Web applications and services increases developer productivity, saving both time and money. Finally, by supporting any programming language, these tools tap the broadest developer talent pool (only about 10 percent of the world’s developers know Java), take advantage of existing skills, and let people use the tool most appropriate for a specific task.

Tools for Migration
Migration from the mainframe no longer requires a big budget and long schedule to recover business rules, develop specifications, write new code and test everything before you see any benefits. Because the move to .NET is evolutionary, tools exist that help you run your current software on Windows-based application and data servers with minimal manual modification of the code you have running today.

If you are running COBOL applications on your mainframe:

- You can run your source code through Fujitsu’s NetCOBOL compiler and directly produce executable Microsoft Intermediate Language code.
- You can use Fujitsu’s NetKicks software to translate your CICS code and produce Active Server Pages that can be accessed with any Internet browser.
- You can take your Batch Processing JCL and translate it to execute under control of Fujitsu’s zBatch control system.
- You can move your DB/2 database over to Microsoft’s SQL server, or if you’re still using a hierarchical file system, you can use Fujitsu’s proven data migration process to restructure it into relational form (which you always wanted to do, anyway).
- You can use Microsoft Systems Architecture as a template that helps you lay out the optimum mix of servers for cost-effective Windows computing.
- You can apply the Microsoft Operating Framework to monitor and control changes to your new environment.

And, after you’ve migrated you can start taking advantage of powerful development tools like Microsoft’s Visual Studio to make those function extensions your business needs but couldn’t afford to make on the mainframe.

The Business Case
The two reasons you should consider moving to .NET - lower costs and greater flexibility - form the basis for the legacy migration business case. The combination of these two can be a compelling argument. You should be able to present substantial cost reduction targets. While each situation needs to be analyzed on a case-by-case basis, successful migrations typically result in operational savings of between thirty-five and ninety percent. Two examples are attached as appendices to this paper. The cost savings come from lower-cost computing technology, reduced fees for software licenses and maintenance, and reduction in the headcount that maintained all the disparate pieces of the mainframe environment, each of which required specialized skills.

Fujitsu has specialized migration cost analysis tools that can be used to help you determine the Total Cost of Ownership for your .NET environment, and we’re prepared to help you make your case.

The big benefit comes from the .NET environment itself. Your programmers, working in COBOL, can create Web Services that can provide up-to-the-minute data and functionality to your suppliers, customers and internal users. The opportunities to streamline your supply chain no longer depend on complex middleware structures and customized interfaces. Your customers don’t need to settle for day-old order status or waiting for a customer service rep to assist them. Industry after industry (and government is included) is setting up information exchange standards based on the Extensible Markup Language (XML), which is the fundamental message mechanism of .NET. These standards are not a natural interface for the mainframe.

If customers can directly place and track their own orders, what will this do to the structure and cost of your customer service operation? If your suppliers receive an up-to-date picture of your order pipeline and inventory, what will the effect be on your warehouse costs?

And if you can’t provide Web Services access to your information, what will your competition do to your market share?

Getting Started
Fujitsu and Microsoft offer a joint workshop on Legacy Migration that can be tailored to your organization’s specific situation. This is a high intensity, two-to-three day event that requires one to two weeks of preparatory activity. It addresses the business aspects and the technical details of migrating from your mainframe to .NET, and it provides detailed information on technical topics that are important to your organization: scalability, security, reliability and performance.

Talk to your Fujitsu or Microsoft representative to schedule a workshop.
Success Stories

California City School District

The Problem
- Significant portion of total IT spend on maintenance of existing environment
- Time consuming, expensive development platform
- Looking for a less costly solution

Legacy Environment
- Amdahl 5890 Mainframe, 4500 Users
- 19,000 JCL Jobstreams
- 2,300 Batch Programs, 4 Million Lines of Code
- 500 CICS Programs
- 1M Lines of Code
- 10,000 Data Files

New Environment
- Database Servers
  - 2 Quad Xeon 500s Production
  - 2 Quad Xeon 500s Failover
- Production Servers
  - 2 Quad PIII 500s Batch Servers
  - 2 Dual P1-133s Online Job Submission, Batch Scheduling

Result
- Overall processing time reduced significantly
- Batch program execution 30% faster
- Extension and enhancement of legacy COBOL and CICS code
- Significant cost reductions for hardware, software and operations and development

“This was a project that some people said couldn’t be done… We were able to leverage our initial investment in COBOL into a cost effective, efficient system that can serve us well into the 21st century.”
- Paul Gustafson, Systems Development Coordinator

California County Court System

The Problem
- Poor response times during peak operation of case management application
- Chargeback costs escalating +$60,000/month
- Initial migration effort to J2EE failed due to lack of scalability

Legacy Environment
- IBM ES9000
- OS 390
- COBOL and CICS - Case Management Application

New Environment
- Dell PowerEdge 2550 Dual PIII 1.1GHZ
- Dell PowerEdge 4500 Dual Xeon 900MHZ

Result
- Dramatic reduction in staff costs
- Systems costs reduced from $720k annually to $20k total
- Maintained transactional integrity and improved performance

“By porting the application to NETCOBOL we improved the performance and maintained the transactional integrity of the original application. Our costs are now a fraction of what they were before.”
- IT Manager