

Legacy Modernization: Realities and New Options

Prepared by: eCube Systems. LLC
January 2010

www.ecubesystems.com



eCube Systems

Executive Summary	3
Introduction	4
Why are Legacy Systems Like Horseshoe Magnets?.....	4
The Current Environment	5
The ROI Bell Curve	6
Extending the Bell Curve	6
Modernization Options: Extending The ROI Curve	7
Standardized Integrated Maintenance Environment.....	8
Conclusion	8
Appendix: Real World Case Study.....	9

Executive Summary

When addressing the challenges of legacy systems, Enterprise IT managers often reach for well-known solutions including user experience modernization, application connectivity, architectural modernization, database/application modernization and application re-engineering. Unfortunately, the benefits vary widely depending on circumstances.

In reality, bottom-line costs, business needs and stubborn technical realities often limit a solution's applicability. If partially implemented, the result is a complex composite of legacy, contemporary and half-modernized systems.

eCube Systems, a leader in enterprise modernization, provides all of the customary modernization services to clients in the US and Europe. Now eCube delivers a new, game-changing offering -- a standardized Integrated Maintenance Environment. (IME)

eCube's IME encapsulates specialized skills, proprietary processes and specific knowledge of your environment into an Integrated Maintenance Environment, designed to help less-experienced developers manage tasks that previously required specialized skills and years of experience.

Called NXTware Remote, this universal maintenance platform (based on the Eclipse Open Source IDE) allows cross-functional teams to productively maintain and develop legacy, contemporary and modernized systems.

By modernizing the development and maintenance process, NXTware Remote is proving to be an effective modernization tool, lowering the costs and complexity of legacy systems.

Introduction

Why are Legacy Systems Like Horseshoe Magnets?

Whether they know the answer or not, this is a riddle that every IT manager with a few years of experience has lived through. The scenario is a familiar one. A new technology gains broad acceptance as the next great technical "savior" within the software industry -- guaranteed to save IT organizations time, money and effort.

The promising technology could be a new architecture, language, data representation or middleware, but in any case, its reviews are "positive." In no time, it attracts the support people, organizations and existing technologies. Budget soon follows, money seems magnetized to anything connected to it.

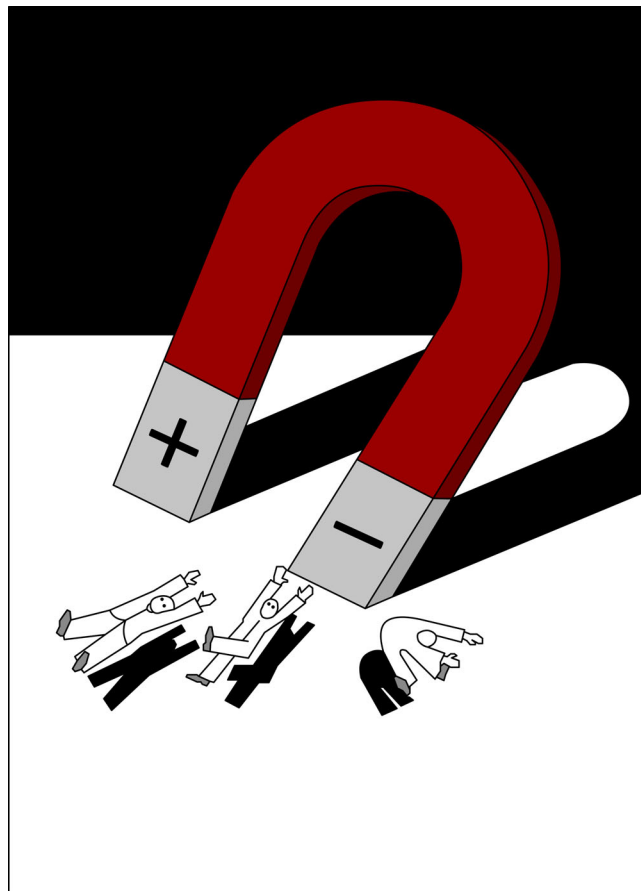
Soon after, things change. The technology worked, *nearly* as well as promised. It managed to save time, money and effort. It helped build important critical applications. But industry trends change quickly, soon there is a new technological hero.

Suddenly, yesterday's promise of technical redemption is not that attractive. Compared to the latest "next big thing", the old system is getting negative reviews. Now a "legacy system," it has a high cost of ownership, requires specialized skills, vendor are reducing support, etc.

It becomes difficult to find people to work on these old applications – and nearly impossible to get budget to start new projects based on the suddenly unpopular technology.

Somehow, you've gone from the positive side of the horseshoe to the negative side, in a single application lifecycle.

The story doesn't have to end that way...



The Current Environment

Valuable applications built on older technologies present challenges to all types of companies and IT organizations. These legacy systems present issues that are difficult to avoid, tricky to solve and too expensive to ignore.

The shifting technology landscape, as referenced in the Horseshoe Magnet story, continues to add complexity. With every "next" generation technology added to their portfolios, software vendors are forced to discontinue or minimize support for an older offering.

Companies running strategic business operations on these older platforms can not change so quickly. Often, two new generations of technology may gain broad acceptance before companies can make effective decisions on what to do about their oldest legacy systems and increasing maintenance costs.

At the same time, each generation of new technology broadens the legacy issues that IT managers must address. With each passing year, the legacy skills that IT managers need to run their businesses become more and more expensive, in turn driving maintenance costs higher.

Inevitably, companies that have invested in prior generation technologies must confront increasing gaps in their ability to sustain and leverage these systems. Without continued maintenance, functional improvements and support for contemporary development languages and service-oriented architectures, technology managers are faced with increased operational costs, additional risk and an inability to clearly extend technology value.

In these situations, there are a limited number of available responses:

- Prematurely retire working profitable applications and invest in replacing them with new applications completely re-engineered from scratch;
- Buy expensive, off-the-shelf-applications and attempt to customize them to duplicate the functionality of the existing system;
- Do nothing, allow the value of the investment to diminish and ride out the company's initial investment for as long as possible;
- Engage in some type of incremental modernization process

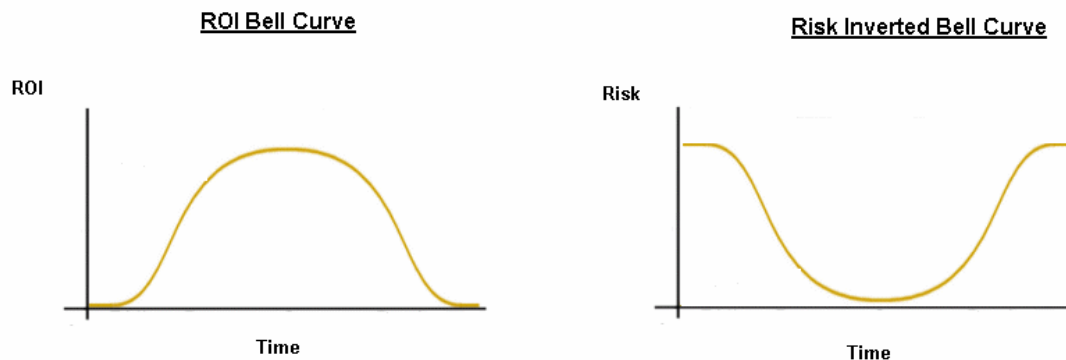
Clearly, technology managers expect that new expenditures will increase business capability and productivity -- not just replicate an existing system.

At the same time, technology management professionals understand that doing nothing only allows the legacy gap to widen, adding complexity and cost to the eventual long-term solution.

The ROI Bell Curve

Before discussing what can be done to modernize legacy applications, it's important to understand the application lifecycle as it applies to return on investment (ROI) and risk.

The bell curve is familiar model. Most things in life follow its pattern in some form or fashion. The models below illustrate two bell curves. One measures ROI, while the other, inverted, measures risk.



The ROI bell curve demonstrates how an application moves through a period of low ROI during the development and early stages of deployment, to a period of high ROI where the application is said to be in its "sweet spot." The productive period is invariably followed by a swoon in ROI, where the costs of operation and system aging start to diminish the system's return on investment.

The inverted risk bell curve follows in a similar way with periods of high risk interrupted by a period of low risk operation in the middle. These patterns are inevitable for most systems and applications. However, these cycles can be managed. The best strategy is to overlap two application bell curves so a second system reaches its "sweet spot" while the first system is ending. This pattern ensures that companies can avoid two simultaneous periods of low ROI while bridging from one application lifecycle to another.

Of course, managing application lifecycles in this fashion requires solid, long-term budgeting -- something that not all companies can consistently provide. This reality would seem to lock organizations into expensive periods of operation where they maintain the end of one application lifecycle while ramping up a second.

There are other solutions to this legacy modernization challenge.

Extending the Bell Curve

It is possible to extend the "sweet spot" of both the ROI and risk curves through a process we call Enterprise Evolution. Enterprise Evolution applies a variety of modernization disciplines to an application or system, making it possible to continue to operate at high levels of performance and to affordably meet new and ongoing business needs.

Modernization Options: Extending The ROI Bell Curve

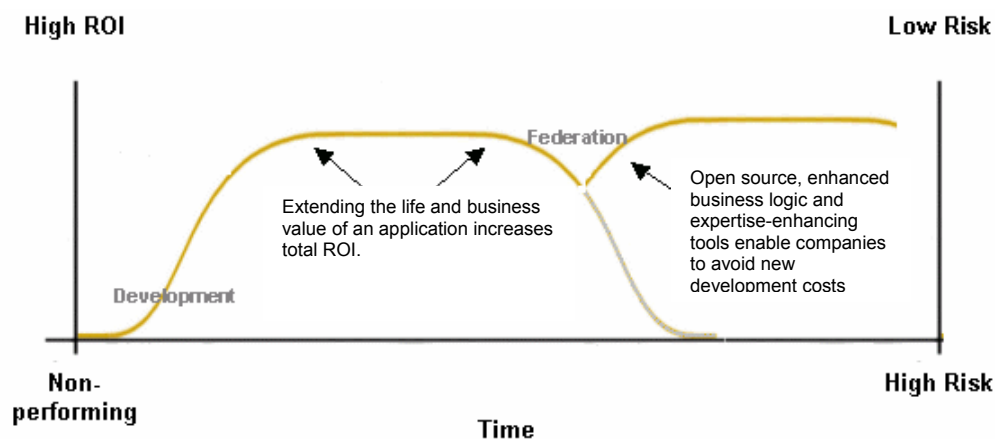
The ROI bell curve can be extended by applying one or more of the standard modernization disciplines as part of the Enterprise Evolution process.

These disciplines include:

- Hardware Modernization (actual or virtual operating systems)
- Application/software modernization
 - o User interface modernization
 - o Connectivity and integration
 - o Architecture modernization
 - o Database modernization
 - o Application modernization
 - o Application re-engineering
- Development and maintenance process modernization
 - o Standardized Integrated Maintenance Environments

eCube specializes in the Enterprise Evolution process and helping clients apply the proper modernization discipline to their environment. Careful analysis is required to determine the best discipline or combination of disciplines to apply in any give situation. Powerful methodologies, such as eCube's ARM Process (Assess, Remediate, Modernize) have proven to effectively define a problem space and identify the best solutions.

Additionally, ARM enables a phased approach to application transformation, delivering evolutionary transformations that avoid the high cost of pure "big-bang" modernizations. Solutions like ARM ensure that every modernization dollar spent provides real business benefit, even if the project is delayed or canceled.



A second facet of Enterprise Evolution is the federation of application life cycles, which in effect, removes both the low ROI / high risk slopes of development and retirement. This is accomplished by federating old application business logic with open source and new modern solution logic. The federated model provides a dynamic means to reduce the costs associated with both enterprise maintenance and new development.

NXTware Remote: Driving Innovation in Legacy Modernization

eCube Systems, a leader in enterprise modernization, provides all of the customary modernization services outlined above, but now delivers a new, game-changing offering -- a standardized Integrated Maintenance Environment (IME) called NXTware Remote.

This solution applies the federation model to extend the ROI bell curve. Rather than focusing on modernizing hardware or software, NXTware Remote modernizes the process. This approach allows IT organizations to create cross-functional teams that can efficiently maintain and develop legacy, contemporary and modernized systems--without the specialized or legacy skills these systems often require.

NXTware Remote encapsulates specialized skills, proprietary processes and custom utilities in its open and integrated maintenance environment. Using the ARM Process, eCube incorporates the specifics of your development environment and engineering process into NXTware Remote. With this built-in expertise, less-experienced developers can manage tasks that previously could only be done with specialized skills and years of experience. Basic engineering skills can be translated across platforms, operating systems and even decades, to drive greater productivity and reduce legacy maintenance costs.

NXTware Remote support a wide array of development languages, remote operating systems and engineering requirements. Its powerful wizards guide engineers through maintenance, testing and development processes they know little about. Tools such as code-assist and custom library awareness make new team members more productive, faster.

Conclusion

Modernizing the development and maintenance process is proving to be an effective tool in lowering the cost and complexity of legacy systems. IT managers now have additional tools in their modernization toolbox. As they work to reduce risk and extend ROI, eCube is uniquely positioned help companies achieve their business and technical goals with NXTware Remote, Enterprise Evolution and the ARM Process.

Contact eCube Systems for more information at www.ecubesystems.com.

Appendix: Real World Case Study

eCube Case Study

Cross Functional Development Increases Productivity on OpenVMS

▶ Challenge: Lower the cost and increase the speed of developing batch programs supporting critical business operations	▶ Tools: Java Eclipse IDE NXTware Remote for OpenVMS
▶ Applications: Core business operations	▶ Language Platform: Solution transitioned from C language to Java
▶ Target Platform: OpenVMS Integrity and Alpha Servers	▶ Client: Mutual Insurance Company

OVERVIEW

- The business provides diversified insurance services such as life, salary, health. With over one million customers, the company is a leading mutual insurance company (MIC) and a known leader in the group insurance industry and investment and retirement sectors.
- Solution lowered MIC's cost of operation by enabling an under-utilized pool of Java developers to take over the programming of batch processing applications on OpenVMS. These batch applications were historically written in the C language on HP VMS servers. In the process, the solution improved MIC's business performance, reduced information technology (IT) operations costs and eliminated the high costs and risk associated with moving away from the OpenVMS platform.

SITUATION

Batch processes still play a very important role in IT services at MIC, where real-time applications are updated by batch processes on a regular basis. Batch processes are used because they are still the most economical way to automate computer interactions and access expensive computing time. It is not necessary or efficient to update every aspect of a system and its data after each transaction. Additionally, new compliance rules and changes in underlying IT infrastructure make it important to have access to batch-oriented workflows that can be scheduled or triggered by business conditions.

Historically, MIC's batch programs running on OpenVMS platforms have been written in COBOL and the C language. However, over the years, the number of COBOL and C programmers in the organization has diminished. Moreover, as these skills become scarcer, it became more expensive to develop, locate and/or hire resources with these skills.

Concurrently, the company had found it easy to staff and develop projects based on the Java language. In fact, within the IT organization, there were under-utilized software engineers with these skills. These developers were very successful engineering new applications using Java and the Eclipse Integrated Development Environment on Windows and Linux-based computers. However, these engineering resources did not have the skills to work on the OpenVMS platform and its command line and non-graphical environments, nor were they interested in learning to do so.

SOLUTION

The MIC had three basic choices:

- Replace the OpenVMS systems
- Hire new C and COBOL programmers
- Cross-functionalize: Find a solution that enabled their Java engineers to work on OpenVMS

After conducting an assessment of software assets and transition options, it was deemed too expensive, risky and time consuming to replace and port 20 years of work on OpenVMS. The assessment process determined that it would cost nearly a million dollars to port and test the existing code -- and even more to implement the replacement infrastructure and servers. This did not take into account parallel costs of running and maintaining the production systems.

Hiring new programmers presented other problems. In addition to the cost and difficulty of locating and hiring new C/COBOL developers, there was the fact that it would take time for new programmers to learn the business and become productive. Most importantly, the company at the time was trying to reduce head count, not add new developers with specialized legacy skills.

The approach preferred by the Director of Infrastructure and Integration and recommended by the assessment was the third option: Find a solution that enabled their Java engineers to work on OpenVMS. It was agreed that the ideal solution would combine modern tools that their developers already used. In their environment Eclipse was the IDE tool of choice. The final requirement was the ability to develop remotely on the OpenVMS Alpha/Integrity servers.

PROCESS

The IT leadership at MIC was determined to implement the most cost-effective and functional solution. After speaking with HP, they were introduced to eCube and its NXTware Remote Product. NXTware provides Eclipse-based Java developers a means to develop code on local workstations (Windows or LINUX) then remotely compile, run and debug code on OpenVMS Alpha/Integrity servers.

There was some skepticism among the OpenVMS administrators and caution within management, so a Proof-of-Concept was arranged. A team of three Java developers were provided with NXTware Remote Studio, the Eclipse plug-in, while an Alpha server used for development and testing purposes was installed with a NXTware Remote Server component.

Management then specified new batch program business logic to be developed. Without any training in OpenVMS development languages, operating systems, nor the use of telnet, FTP or ANT, these Java developers were able to develop, compile, execute and debug business logic on OpenVMS. The developers and the OpenVMS administrators both gave the product high marks. Some of the evaluators suggested small modification to better suit their process. Before the completion of the PoC, these changes were implemented by eCube, tested and accepted by the client.

Soon afterwards, the MIC agreed with eCube to a site license that would allow them to expand their use of OpenVMS to Java developers. At the same time, it identified and closed new internal business based on these capabilities.

ROI

The MIC reaped benefits in improved development productivity through the use of cross functional development teams, while eliminating the need to hire new developers skilled in OpenVMS.

Productivity:

- increased development output by a factor of three in initial tests and projects; The team immediately went from two developers writing batch programs to a flexible team that could expand to more than six on demand.

Business operations:

- reduced operational risk by eliminating unnecessary system and architectural change
 - were able to increase productivity without increasing head count
 - improved ability to meet business needs due to a more agile and responsive development team
-

TECHNOLOGY

Application Platforms NXTware Remote
Operating System Windows/LINUX and OpenVMS
Development Language Java

IMPLEMENTATION

Services Provided Executive Assessment
Other options evaluated Hiring new C/OpenVMS developers;
re-building existing capabilities and porting of logic
Actual Costs software and consulting 7 -10% cost of
other alternatives
Implementation Team size One
Implementation time One week, in addition to a three-week PoC



eCube Systems

eCube Systems offers a family of middleware evolution products and services that maximize return on technology investment by leveraging existing technical equity to meet evolving business needs. Fortune 1000 companies and government agencies turn to eCube Systems to reduce risk, extend ROI, and increase productivity as they consolidate existing capabilities and evolve legacy systems to contemporary SOA platforms, such as ESB and Web Services.

eCube Systems, LLC, is headquartered in Montgomery, Texas, with marketing offices in Boston, MA and R&D in San Mateo, California, USA. For more information, visit us at <http://www.ecubesystems.com> or contact eCube Systems by email at ev_sales@ecubesystems.com or by telephone: 866.493.4224 x3.



eCube Systems

© eCube Systems, L.L.C. 2007

Headquarters
550 Club Drive, Suite 272
Montgomery, TX 77316 USA
T: 866.493.4224
F: 936.449.4880
ev_sales@ecubesystems.com