White Paper

Rewriting Legacy Applications for Business Agility

by Abishek Bhat & Anuradha M

TRIGENT
OVERCOMING LIMITS

Background

In the past few decades IT, and software engineering have continued to change with slow improvements in hardware, language, methodology, and infrastructure being intersected by paradigm-shifting innovations. These changes are mostly dictated by business needs with IT budgets being spent on maintaining existing software, and making upgrades to the same. However, upgrades, and migrations deplete resources before actual business benefits are realized which is giving way to the 'application rewrite' approach.

Some years ago, applications were written in popular languages and compiled into formats that were unique to a processor or operating system. These applications were self-contained, remained stand alone, and locked in private data centers. The idea behind this approach was based on the assumption that their use would be long term. In the recent past, all of this has undergone a complete paradigm shift. These heavyweight dinosaurs are limited in the purpose they serve, and as technical innovations increase, have become unwieldy, pulling down thriving businesses.

The development process has also gotten much faster evolving from the traditional waterfall methodology to iterative development models and collaborative DevOps practices with automated and continuous integration and delivery. The development trend has changed from complete ready-to-deploy applications to smaller release cycles of microservices and application programming interfaces (APIs).

Applications belonging to the new school of development are built on standard web services and are intuitive in nature, i.e. answering questions such as 'What is it that a typical user wants from an application?' and 'What does an application have to do to react or respond to the user's requests?' Deployment is more flexible and hardware not so heavy. These applications are written to standards such as Java Platform, Enterprise Edition (Java EE) etc. so they can be easily deployed on combinations of software and infrastructure. Mission-critical applications are now being deployed, managed, and supported by cloud-computing providing the way for reliable, cost-effective enterprise computing. It is thus possible to modernize an existing application or suite of applications in a way that yields smarter faster value for money while staying up-to-date with current technologies to evolve with time.

Setting the Stage for Application Rewrite

Maintaining and upgrading existing legacy systems is a huge challenge for CIOs today and they struggle with the problem of modernizing these systems while keeping their functionality intact. Legacy systems are hardwired by rigid process flows making integration with customer relationship management (CRM) software and internet-based business applications virtually impossible. Also, IT managers are challenged to find developers to work on these applications as the languages too have become dated. Many of these systems continue to provide a competitive advantage by supporting existing business processes and owning tonnes of historical data. Maybe because of this, over 80 percent of existing IT systems continue to run on legacy systems. International Data Corp. estimates that 200 billion lines of legacy code are still in use today on more than 10,000 large main frame sites.

In spite of its undeniable value-add, the need to modernize or rewrite existing applications is being considered by businesses simply because of the cost structure associated with maintaining them. According to industry polls, as much as 85-90 percent of an IT budget goes into operational and maintenance costs of legacy architectures. But modernizing legacy systems from scratch comes with its own challenges. Researchers from the pioneering 1991 DARWIN project at the University of California, Berkeley, listed several deterrent factors:

- Management rarely approves a major expenditure if the only result is lower maintenance costs, instead of additional business functionality.
- Development of such massive systems takes years, so unintended business processes will have to be added to keep pace with the changing business climate, increasing the risk of failure.
- Documentation for the old system is frequently inadequate.
- Like most large projects, the development process will take longer than planned, testing management's patience.
- And finally, there's a tendency for large projects to end up costing much more than anticipated.

In the 1990s, many businesses adopted the incremental approach of modernizing existing applications. In this piecemeal approach, large projects would be split into manageable chunks. But this method continued to prove challenging as during the migration process, legacy systems had to inter-operate with target systems, adding complexity to an already complex process.

Today the migration process is being simplified by tools which help convert legacy code into modern languages which in turn help to break functionalities into components. These components have clearly defined application-programming interfaces (APIs) and the Internet is providing the dictum for modernization. The Web is empowering businesses to save time and money by delivering to customers and partners, business processes which remain locked in legacy systems. In this new scenario too, some challenges continue to haunt the process and these are related to answering some pertinent questions:

- Is it possible to accurately specify a several decade-old application?
- Without an accurate specification, how would one test a re-written application?
- Do you start from scratch and if yes, do you simply dump the existing code which is equivalent to losing years of learning and knowledge?

Phases in Application Rewriting

The IT industry has evolved and the axis has moved from technology-driven to user driven. IT has also transitioned from being a cost-center to a value-center within an enterprise. According to a research report from Robert Frances Group: "Traditionally, IT has been viewed as overhead versus a revenue-generating business unit such as sales…Now, IT executives are realizing that if there is not a shift in thinking and an ability to demonstrate the tangible business value contributed by IT services, their departments will be further scrutinized, and their budgets continually reduced."

Can legacy applications fulfil today's business requirement, help IT to be a value center and be user driven? - Are the questions that need to be answered.

Many companies are following a phased, step-by-step process to achieve application rewrite status.

Lift and Shift

In the first step, these companies are lifting and shifting their existing application components to a modern deployment platform. For example, in application virtualization, an application is packaged with the operating system and run as a virtual machine instead of on dedicated hardware. Lifting and shifting is not as much modernization as much as the foundation for the ensuing process of refactoring the application. Some companies prefer to stop here as they can experience visible improvement in existing application performance. By simply residing on modern platforms these companies experience better speed, flexibility and lower operational costs.

In the lift and shift method, the architecture remains unchanged but the application has a new lease in life sitting on a modern deployment platform.

However, this approach may not work for all applications. For example, applications which are locked into specific vendor-based solutions may need content management capabilities.

Adding layers

Over the last few years, companies have also added layers to their existing applications to add more functionalities. This has helped to bring down costs and increase functionality while retaining the older more complex functionalities.

Adding new layers requires an interface or logic that merely serves the purpose of connecting existing applications to the new layers. In this method, the existing application remains untouched and businesses tend to feel safe with this approach. The architecture of the existing applications remains the same but new functionalities have been added.

Application Rewrite

The two steps mentioned above pave the way for application rewrite. Here, new functionalities retire existing applications, without actually creating new applications from scratch. In this approach, tempting as it may be, the need will not be for rewriting the application from scratch but making a sensible decision on retaining the good and throwing out the unnecessary.

Step-by-Step Application Rewrite Process

Rationalization of Existing Systems and Process

More often than not, application rewrite is triggered by business requirements with immediacy attached to it, making the whole process a knee jerk reaction. However, application rewrite cannot work without a certain rationalization which helps to identify the applications with the greatest risks and highest costs, the transformation process which will give the greatest yield, and finally identifying the changes that can be made relatively quickly reducing cost and complexity.

Because applications are typically used across various areas of an organization they are deeply ingrained into existing processes. As a result of this, gaining a clear and accurate understanding of the depth of information can be difficult. There are, however, proven portfolio rationalization methodologies that leverage automated tools to reduce the time and effort it takes to identify and categorize applications.

This step needs more than technical prowess. It requires a global view of the enterprise objectives, user requirements, and careful consideration of existing and available technologies. Based on this information, an organization can then retire, consolidate, replace, re-platform, re-architect, and finally re-write existing applications.

Data-Driven Approach

A data-driven analysis of existing application code can provide insights to reduce uncertainty and improve overall project planning and predictability. Large, mainstream applications can be daunting to say the least and difficult to determine the time, resources and costs for modernization. These applications could easily have millions of lines of code.

By using specialized tools, algorithms, and statistical analysis it is possible to decompose clustered code to see what functional tasks are performed, and how much code is duplicated or cloned. By doing this deep analysis the effort of rewrite can be reduced considerably.

Stop or Continue?

Rationalization and a data-driven approach can help organizations to decide whether they want to really rewrite their existing applications. In many cases converting an application using automated conversion tools, combined with code analysis, conversion planning, and technical architecture can help to convert obsolete languages and archaic codes into industry-standard data. Existing historical data can also be moved into SQL databases. Conversion tools and methodologies enable development teams to 'continue the code' and enhance an application using a language they are familiar with. Conversion has the added advantage of making testing simple and straightforward.

Modern Frameworks

If the decision to rewrite continues to stand, then enterprises can benefit from application development frameworks to break free from legacy constraints to manage costs, time and effort. Open source frameworks, for example, have proven to be productive and are therefore used extensively by developers. Frameworks and fabrics can abstract and deliver functions as services in a distributed environment in the data center or in the cloud, providing improved availability, security and manageability. They can handle spikes in traffic. They are also valuable in supporting Big Data analytics.

Prioritizing User Experience and Modernizing Software Lifecycles

In the traditional method, software began with the function and technical requirements gathering approach. However, today software is where a business actually operates.

Digital models are changing the way people work, goods are sold and bought, and how we communicate. This has made it compelling for businesses to rewrite their business applications to make them intuitive, device agnostic and cost effective.

User experience is not just the attractive graphical interface any longer. It is personal, powerful, first-hand, and is the foundation for forming a relationship with the business. Smart businesses are therefore investing in understanding their customers' needs, distinguishing a product in the market place, and delivering all kinds of new services with the end objective of increased productivity, customer satisfaction and competitive advantage.

To offer a superior user experience which is based on data-driven research requires iterative feedback and testing with real users. It is an ongoing process, with applications that are designed to evolve, continuously capture data and feedback, and incorporate innovations.

Along with legacy applications, inefficient and outdated software development lifecycles can also weigh down businesses, adding unnecessary risk, costs and time. The new, integrated DevOps method can unite people, processes, and technologies to integrate large teams, rationalize and automate processes. By applying Agile principles and simplifying quality assurance integration becomes easier.

The cloud had changed data center operations to deliver low-cost infrastructure on-demand. It is changing the way applications are built, deployed and managed. Service-oriented cloud application platforms, powerful frameworks and middleware are empowering organizations to focus on the business logic that brings unique value to market and deliver faster.

By rationalizing application portfolios to rewrite applications, businesses can benefit from new data models and cloud applications. The first step is to identify the need to change and partner with technology companies that understand the process of rewriting existing applications.

Why Trigent?

Maintaining and upgrading legacy systems is one of the most difficult challenges CIOs face today. While constant technological change has weakened the business value of these systems, they have been developed over many years and financed through large investments.

Legacy applications are monolithic, centralized and synchronized - that rely on costly infrastructure to provide resiliency and performance.

Despite their obsolescence, legacy systems continue to provide a competitive advantage by containing invaluable knowledge and historical data. Modernizing legacy applications that are mission-critical must preserve their functionality during the migration or re-platforming process.

Trigent uses a systemic approach to modernizing legacy application that addresses not only technology re-platforming but also data migration, process redefinition, and integration issues - while taking care of staff training and participation needs.

Trigent will evaluate multiple available strategy to migrate your legacy applications that takes your unique needs and situations into consideration. The approaches could be Re-host (Lift and Shift), Re-Platform (Lift and Reshape) or Re-Architect (Legacy rewrite).

Trigent will rewrite your legacy applications into modern, microservices based cloud-native architecture that has built in resiliency, cloud-scale and hyper-performance using low cost, on-demand infrastructure.

ARCHITECTURE COMPARISON

Legacy	**Cloud**
Scale up	Scale out
Monolithic	Microservices
Tightly coupled	Loosely coupled
Fixed capacity	Elastic capacity
Latency intolerant	Latency tolerant
Web channel delivery	Web, mobile, and voice enabled device channels delivery
Build everything	No undifferentiated heavy lifting
Centralized, relational database persistence	Distributed, replicated, polyglot persistence
Manual fault recovery	Self-healing
Single points of failure	Antifragility, degrade gracefully
Allocated costs	Metered costs
No operational agility	DevOps agility
Siloed development team	DevOps team

About Trigent

Trigent is a CMM Level-4 technology solutions company with its US office at Southborough, MA, and India office at Bangalore. Trigent provides comprehensive solutions for business problems via outsourced software product and applications design, development and quality assurance. Trigent serves customers like Independent Software Vendors (ISVs), enterprises and SMBs in the High Tech, Healthcare, Education, Ecommerce and Manufacturing areas. Trigent's solutions help clients overcome budget, schedule and resource constraints.
To learn more about Trigent visit www.trigent.com

Our Headquarters

Trigent Software Inc.
2 Willow Street, Suite #201
Southborough, MA 01745
+1 (877) 387-4436
sales@trigent.com

Other Location

Trigent Software Ltd.
Khanija Bhavan, 1st Floor
#49, West Wing, Race Course Road
Bangalore - 560001, India
+91 (80) 2226-3000

CERTIFIED ISO 9001:2008 COMPANY

Microsoft Partner
Gold Application Development
Gold Collaboration and Content

amazon web services Partner Network
CONSULTING PARTNER

2016 SOFTWARE 500 COMPANIES

2017 IAOP The Global Outsourcing 100

TOP 10 SiCLOUD SOLUTION PROVIDERS - 2017

TRIGENT
OVERCOMING LIMITS