MODERNSYSTEMS

WHITEPAPER

DEVOPS & MODERNIZED MAINFRAME APPLICATIONS

IMPROVE DEPLOYMENT FREQUENCY, LOWER FAILURE RATES, SHORTEN LEAD TIME BETWEEN FIXES, & SPEED MEAN TIME TO RECOVERY

Introduction

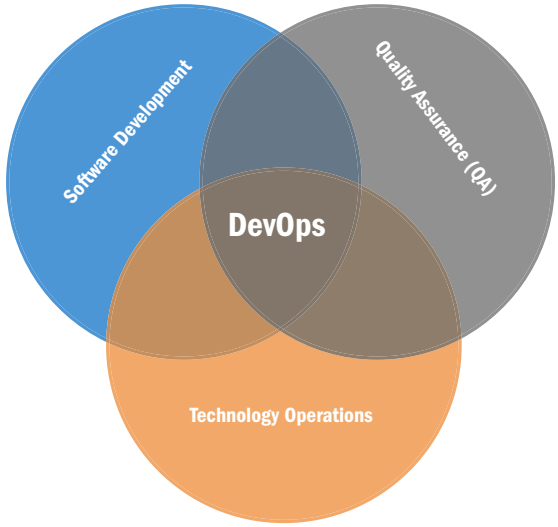
Businesses are under tremendous pressure to create new value for their customers. Coupled with the enormous demand for software-based products and services, the trend of competitive differentiation through technology isn't a huge surprise. However, organizations are finding that traditional approaches to software development and delivery are not sufficient to meet these needs. DevOps, a contraction of development and operations, represents a philosophy that drives collaboration between teams to achieve efficient, competitive, sustained innovation.

A Paradigm Shift Is Forcing Legacy Development Philosophies Into Obsolescence

As technology evolves rapidly before our very eyes touching all aspects of society, one thing is abundantly clear: Things are moving faster. The microcosm of software development is no different. In order to remain competitive and relevant, businesses require speedier development philosophies to bring products and services to market faster. Many early attempts to improve software development focused on better ways of defining and detailing requirements, designing comprehensive architectures, and developing software in a very regimented manner.

Around the turn of the century, a majority of the software development processes that had been developed in the 1980s and 1990s were being criticized as bureaucratic, slow, and overly regimented. In the mid-1990s, in reaction to the difficulties these heavyweight software methods presented, there was a small contingent of thought-leaders tinkering with innovative approaches to development, enabling organizations to quickly react and adapt to changing requirements and technologies. They realized that embracing change, and executing in a manner that not only accommodated this change, but fostered it, would result in a much more successful development strategy. The term "agile software development" emerged from this group in 2001, catapulting the software world into a new era.

Since then, the agile philosophy has been wildly successful. As organizational leaders began to see the successes that these techniques brought to the development world, many wondered how it could expand to include the operational side of the house. These curiosities gave rise to an offshoot of agile development called DevOps. The old view of operations tended towards defining the development side as the creators and operations as the people that deal with the creation after it's implemented. These two being treated as siloed concerns, and the inefficiencies inherent in that approach, is the core driver behind DevOps- to integrate efforts between development and operations. In this way, DevOps can be implemented as an outgrowth of the Agile philosophy.

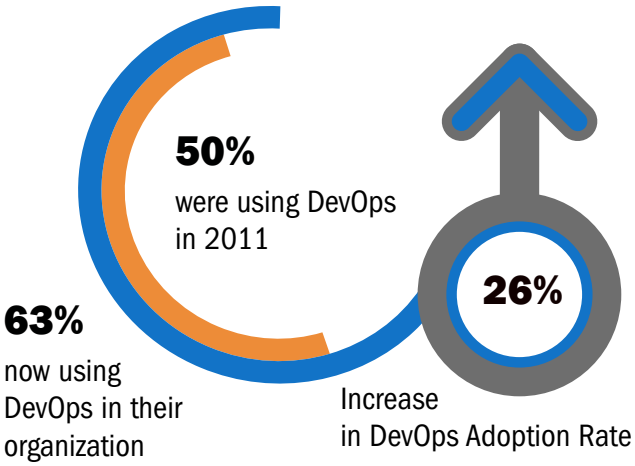


In this whitepaper, we will focus on what DevOps is, why it is such an important advancement in software development and delivery, outline the challenges involved in implementing DevOps in a mainframe environment, and how best to extend DevOps to core applications to maximize efficiency and competitive edge.

What's the Big Deal?

Agile development process is designed to enable delivery of working software in smaller and more frequent increments, as opposed to the infrequent approach of the waterfall method. High deployment rates will often pile up in front of IT Operations. Clyde Logue, founder of StreamStep, is attributed as saying "Agile was instrumental in Development regaining the trust in the business, but it unintentionally left IT Operations behind. DevOps is a way for the business to regain trust in the entire IT organization as a whole." When code is not promoted into production as it is developed (e.g., Development delivers code every two weeks, but is deployed only every two months), deployments will pile up in front of IT Operations, customers don't get value, and the deployments often result in chaos and disruption.

DevOps is not something you buy, it is something you do. In order to "do" DevOps, you need to connect it to your business in a meaningful way to ensure long-term success. But change is hard, so how do you get your non-technical resources/upper-level management to see the benefits of the collaboration, culture-shift, and communication DevOps brings and invest in it for your business? Consider the results from this 2013 Puppet Labs survey of over 4,000 IT professionals in over 90 countries.



Organizations Who Implemented DevOps Saw:

63%
Improved Quality of Software Deployment

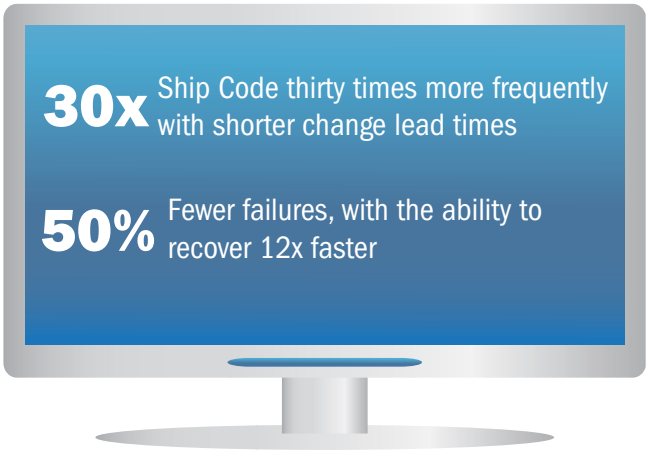
63%
More Frequent Software Releases

61%
Improved Visibility in IT Process & Requirements

55%
Cultural Shift Toward Collaboration/Cooperation

55%
More Responsiveness to Business Needs

38%
Improved Code Quality



The Challenge: DevOps & The Mainframe

While there is explosive growth in the number of organizations implementing DevOps, most focus on systems of innovation to start their journey. They do this because shifting a development philosophy and turning the relationship between development and operations on its head it no small task. However, once the kinks are worked out, most IT organizations shift focus onto other "high touch" environments that could benefit from the nimbleness that comes with DevOps. Mainframes are at the top of that list, but often aren't considered because of the constraints inherent to big iron. However, if these constraints can be overcome, applying DevOps to these systems can significantly reduce the burden of these core systems on the entire business. Here's how:

The Three Tenets

Service Enablement

Service enablement is possible across the mainframe stack in a variety of ways. However, achieving this goal without modernization of the legacy system is difficult at best. Web enablement of the presentation layer, a.k.a. the green screens is the most straightforward transformation. Why choose presentation layer over applications, data, and others? The simple answer is that none of the application source is 'available'. Other reasons may be that the data stores cannot be accessed directly because of security restrictions, or that no stored procedures or SQL exist in the application. Service enablement of the presentation layer is as simple as running and capturing the screens, menus, and fields you want to expose as services. This is relatively fast and for the most part, easy. Application service enablement on the other hand, is more than wrapping transactions as web services. This is all about service enabling the behavior of the system and batch processes. It also includes the business rules, data validation logic, and other business processing that are part of the transaction. Service enabling the application is where you get the biggest bang for your buck, but it is also significantly more complex than enabling the presentation layer.

Automated Testing

Increasing the frequency of feature implementation with DevOps requires a decrease in the amount of time it takes to validate said features. This means automated testing is a must. Luckily, there are a variety of automated testing applications and services available for all different flavors of mainframe implementations.

The trouble is, business users and the developers responsible for extending the system absolutely must have a solid understanding of the architecture and expected behavior of these ancient behemoths.

In other words, automated testing in legacy environments is a double-edged sword. It is possible, but the likelihood of having a team that understands the systems well enough to effectively implement and digest the results of this kind of testing is incredibly low.

Faster Feature Development & Delivery

Realizing speedier feature development and smoother delivery of applications in a DevOps-friendly way requires a deep understanding of the environment as a whole, a tight interrelationship between users and technical teams, as well as the ability to harness modern technology to achieve the requirements set forth in a competitive way.

As mentioned previously, the statistics are stacked against you when it comes to clear systems understanding. Digging into the past to make sense of code that was written before the internet was invented is a task that is in no way synonymous with efficiency and speed. Furthermore, without effective automation in testing, the time line for deployment increases significantly.

Because these legacy systems and the code that comprises them were created in a bygone era where typewriters and mimeographs reigned supreme, integrating and enabling modern services and functionality is extremely difficult if not costly. For example, mobile technology has created a demand for data unlike anything the business world has ever seen. If you're fortunate enough to enable mobile interaction with your legacy system, you'll still pay a steep price for the increase in MIPS that comes along with it.

Overcoming the Challenge: DevOps & The Mainframe

When it boils down to it, implementing a culture of DevOps against a mainframe environment is incredibly difficult. First, the lack of a service oriented architecture and extensibility make significant time-to-deployment efficiency a difficult goal to achieve. Second, the core concept around DevOps is to connect development with technology operations. Big Iron is notoriously expensive to extend and a difficult piece of infrastructure to manage. When the system itself is the biggest hurdle to realizing a culture of continual experimentation and learning in the name of competitive edge, it is time to replace the system.

Even if your organization isn't ready to rip the bandage off and abandon legacy code-bases, there are options to enable replatforming such as Modern Systems' Application Transparency Platform (ATP), that remove the infrastructure-based constraints that stand in the way of the intimate connection between development and operations. If you are ready to get away from the legacy world, architectural transformation software like Modern Systems' eavRPM can help enable DevOps at much earlier stages than alternatives.

Our Application Transparency Platform (ATP™) automatically transforms databases and data to the relational database of your choice and interprets application code as an ATP Language (ATPL™) on Windows. It allows you to move applications without changes from your current mainframe environment to the less expensive, more robust ATP environment thus eliminating costly maintenance/license fees and reducing mainframe workload. Since ATP directly interprets and executes each legacy command, these applications can execute against the newly reformed relational database without changes to the current language or syntax. ATP also includes a full-featured UI-based development platform that includes application code analysis, flow diagramming, impact analysis, pattern matching, version comparison, and interactive debugging. Wielding ATP, development teams can continue to develop and maintain applications with the support of modern development tools running on modern infrastructure.

The eavRPM Architectural Transformation service enables developers to have the visibility into and make changes to the legacy system. Any necessary adjustments to logical flow are simple, allowing the transition from procedural code to an object-oriented code-base with far less effort than a re-write. Depending on its scope, a complete application can be prototyped in a matter of hours or even minutes. Simply capture lines of legacy code and eavRPM automatically refactors it to C# or Java. Even business analysts who don't have prior programming skills can specify the basic screen processing needed to enhance their performance and experience. They use point-and-click master page definitions and build visual templates with consistent headings, footers, etc. They can also split, merge, or change screens for ease of use. Style sheets are also included as part of the solution to maintain visual consistency in the modernized environment. Database queries and other objects are similarly dropped in place to standardize object access. Elimination of procedural code and the big iron infrastructure is the most complete way to position a transition to DevOps for your core applications.

Although each tool offers a unique approach to modernization and DevOps enablement, both effectively reduce the time and effort involved in evolving legacy applications and the shops that support them into the agile world. Implementing DevOps in a shorter time frame will enable greater cost savings and competitive advantage. Alternatively, enabling DevOps with a grassroots effort takes an exponentially greater amount of time and frustration, which is the most common reason core applications and the mainframe are overlooked.

Overcoming the Challenge: DevOps & The Mainframe

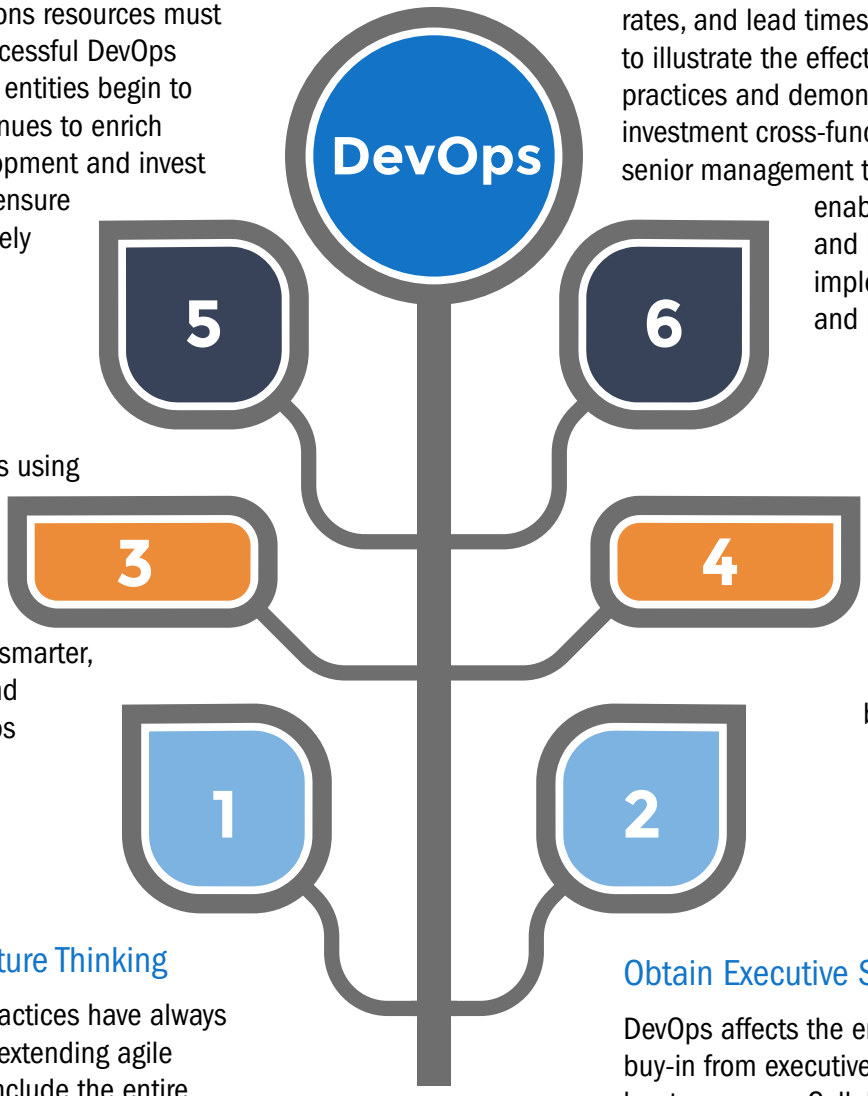
Building a DevOps state of mind requires more than just giving developers root, installing a configuration management tool, or using a specific source code repository." All aspects of the people, process, and technology play a role in- and are impacted by DevOps.

Invest In People

Developers and Operations resources must build new skills in a successful DevOps environment. As the two entities begin to collaborate, identify avenues to enrich appropriate skills development and invest the time and money to ensure these skills are adequately nourished.

Measure Success

Measuring deployment frequency, change fail rates, and lead times allows the organization to illustrate the effectiveness of new DevOps practices and demonstrate a return on investment cross-functionally and to the senior management team. It also enables a system of checks and balances against implementing new processes and philosophies.



Know Thyself

Assess the legacy systems using tools such as our Portfolio Analysis service to ensure a complete understanding of the environment and to drive smarter, lower-risk decisions around modernization and DevOps implementation.

Identify Waste

Use the legacy systems assessment to identify and map stages and handoffs and measure the time it takes to move through a process. Once identified, begin removing, automating, or eliminating components appropriately.

Encourage Big Picture Thinking

Agile development practices have always focused on code. By extending agile beyond the code to include the entire delivered service, the organization will be able to simplify complex systems and modularize services to allow the extention of more rapid and incremental changes to business operations.

Obtain Executive Sponsorship

DevOps affects the entire business. Early buy-in from executive leadership is the key to success. Collaboration with middle management will ensure the organization as a whole commits to DevOps.

A New World Order: DevOps on Core Apps

It is pretty clear that DevOps provides an optimal deployment environment in any given system and the case for bringing DevOps into the mainframe world is compelling. However, what tangible differences will taking the plunge into agility bring?

More Resources to Work On It

It is no secret that the talent pool that is capable of maintaining these ancient mainframe systems is shrinking at an astounding rate. Consider this from a 2013 Computerworld Survey:

ComputerWorld's IT Survey identified a **massive resource gap** facing the mainframe world. Moving to a distributed system mitigates risk around increasingly rare legacy system resources.

46%
are noticing a COBOL programmer shortage

50%
said age of average COBOL staffer is over 45

22%
said the age is 55 or older

64%
said their organizations still use COBOL- more than any modern language except for Java/JavaScript and Visual Basic

nearly 75%
said COBOL represents more than half of all internal business application code

The evolution of software development from a procedural landscape to object-oriented design has opened a rift between the skills required to maintain legacy systems and the skills being utilized by today's developers in the design and implementation of modern software.

By eliminating all or part of the legacy system through modernization, you open it up to an endless pool of resources (probably already in-house) now capable of maintaining and extending it. Furthermore, the right tool such as eavATP will enable the business to leverage legacy skills (COBOL, Natural, etc.) while allowing procedural code and object-oriented code to coexist until the business feels more comfortable with a complete shift away from procedural codebases.

More Service Enablement Capabilities

Cutting-edge software and systems developers are not clamoring to enable COBOL or Natural coverage in their products. They're focusing interoperability with far more modern systems that are grounded in modern code-bases. While service enabling the mainframe is possible as we discussed earlier in this whitepaper, its capabilities are incredibly limited and extremely difficult to bring to fruition.

By modernizing the mainframe environment, your organization enables compatibility with modern tools that were completely out of reach in the past. These core applications are positioned to handle whatever the competitive world has to throw at it far into the future.

More Tools & Capabilities for Automated Testing

In much the same way that modernizing the mainframe opens it up to greater service enablement capabilities, the expansion of options for automated testing- and faster deployment- is practically unbounded. However, having lots of new and more functional automated testing options is just the tip of the iceberg. Undergoing the process of modernization in combination with exposing the underpinnings of the system to a new generation of developers naturally expands knowledge of the system itself. This bridges the gap between the business user, the developer, and the infrastructure needs necessary to support expansion or revision as business demands change. When in-house teams approach testing and functionality with higher visibility, cross-functional efficiencies are inevitably realized.

Now is the Time to Re-think DevOps

The upside to mainframe modernization for DevOps enablement is obvious. With tools such as eavRPM and eavATP, realizing that upside has never been easier. To learn more about mainframe modernization, DevOps application on core apps, and leaving big iron behind, visit us at modernsystems.com.