Mainframe DevOps in Action

From Theory to Reality–A Best Practice Guide for Mainframe DevOps
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About This White Paper</td>
<td>2</td>
</tr>
<tr>
<td>Collaboration</td>
<td>3</td>
</tr>
<tr>
<td>Accelerating Delivery without Compromising Quality</td>
<td>3</td>
</tr>
<tr>
<td>Efficiency</td>
<td>9</td>
</tr>
<tr>
<td>Manage Code Changes with Better Processes</td>
<td>9</td>
</tr>
<tr>
<td>Flexibility</td>
<td>13</td>
</tr>
<tr>
<td>Scale Development and Test Capacity while Managing Cost</td>
<td>13</td>
</tr>
<tr>
<td>Conclusion</td>
<td>18</td>
</tr>
</tbody>
</table>
While there is no templated way for IT organizations to deliver software, most development managers would agree that, whatever the approach, delivering new application innovations faster is imperative. Common motivators include staying ahead of the competition, reducing operational costs, aligning IT to better support the business—or a combination of any of these elements.

While time to market, a clear business advantage, underscores everything, doing things better, easier and for less is never far from the mind of the CIO. This is all driving the need to deliver innovations faster. Perhaps this is why recent research suggests 45% of IT professionals are planning to implement DevOps practices in their enterprise organizations?

Unrestricted by process and liberated by the flexibility to swiftly adopt change, newer and smaller companies can move faster using efficiency-enabling tools and modern collaborative working practices from the get-go. However, moving faster is more challenging for longer-established organizations.

They are typically burdened with complex technologies and disparate teams using multiple development processes, a characteristic of most mainframe shops. So, can your organization cost-effectively achieve higher levels of efficiency by retrofitting a new delivery concept to an infrastructure it was never designed for?

The answer is yes. Successful DevOps does not exclusively belong to any particular business profile or development model. Every organization can meet specific business drivers with incremental, percentage improvements. Delivering new services via mobile and web? Streamlining processes to gain new market share or scale geographically? Improving mainframe efficiency and quality to boost profit margins? Priorities might look different, but all are achievable.

How can parallel development, continuous integration and delivery, early and frequent testing—all particular to DevOps, born in the distributed computing world and based on Agile development practices—transfer to the mainframe world, and applications not designed with DevOps in mind? Because, success is imperative. The business needs the whole of IT, not just some of it, to deliver faster, together, to meet evolving customer demands and maintain a competitive edge.
The key to mainframe DevOps success is in quickly identifying and removing major bottlenecks in the mainframe application delivery lifecycle. Three major challenges are the lack of collaboration among development and testing teams, inefficient development and process integration, and limited resource flexibility to scale needed testing.

About This White Paper
DevOps evangelists frequently talk about the ‘DevOps dream’ but are frequently light on practical ‘how-to’ detail. This paper explains how a selection of our customers have identified and removed specific bottlenecks to rapid application delivery on the mainframe. It is based on our global experience in helping mainframe customers take an incremental approach that aligns with business needs and budgets and is achieved at a sustainable pace. This strategy is proven by the success and evidence offered in the use cases.

Throughout the white paper, we explain how customers leveraged products within the Micro Focus® Enterprise solution set to achieve mainframe DevOps results. The following is a quick guide to the products referenced in this paper.

**Enterprise Analyzer**
Intelligence and analysis tools that provide insight and understanding into applications and business processes.

**Enterprise Developer**
modern application development environment (IDE) integrated with Visual Studio or Eclipse that streamlines mainframe COBOL and PL/I development. An efficient toolset for remote development that integrates directly with the mainframe.

**Enterprise Sync**
A distributed software configuration management solution that helps organizations scale parallel development and continuous integration while preserving mainframe source integrity and configuration management. Integrates with Enterprise Developer.

**Enterprise Test Server**
An IBM mainframe application test environment on Windows. It enables IT organizations to confidently test mainframe application change on scalable, low-cost hardware.

The Micro Focus Enterprise solution set includes:
- Enterprise Analyzer
- Enterprise Developer
- Enterprise Sync
- Enterprise Test Server
We have divided our paper into the three pillars that support successful mainframe DevOps—collaboration, efficiency and flexibility. Real-world customer success stories from around the globe showcase each pillar. The customers are not explicitly named but what is achieved, and how, will be compelling to anyone interested in repeating their success.

Collaboration

**Accelerating Delivery without Compromising Quality**

DevOps is, at its foundation, a cultural approach to delivering applications and services in a way that values time and contributions throughout the delivery chain. An increase in release frequency requires tight alignment and collaboration between line of business, development, testing and IT operations teams.

So, if DevOps is about building interdepartmental bridges, how can members of the development department succeed without a collaborative culture? Unfortunately, this is a common scenario. Mainframe teams are often isolated because they use mainframe-specific development tools and habitual practices that are outdated and overly complex, even to peer developers.

DevOps, in theory, can break down these antiquated barriers between development teams. The challenge is applying theory to practice. Building a ‘collaborative culture’ sounds easy enough. Just do what leading companies do; break down those cubicle walls and watch a collaborative culture grow under the sunlight now shining onto your new, large, open floor plan, right?

According to researcher Vinesh Oommen, in a study of open office plans:

> “In 90 per cent of the research, the outcome of working in an open-plan office was seen as negative, with open-plan offices causing high levels of stress, conflict, high blood pressure and a high staff turnover.”

**Cracking the ‘Collaborative Culture’ Code**

Micro Focus customers have cracked the collaborative culture code by breaking down walls, just not the literal kind. Mainframe application delivery can be improved by up to 40% simply by using modern, intuitive tools and an integrated development and testing environment to boost collaboration. The following use case is from a large banking and credit card institution working with different offshore service providers and in-house z/OS mainframe development, testing and operation teams. Here’s their story:
**Challenge**
To deliver new functions faster and in shorter cycles, the organization had to streamline their development processes. However, disparate development teams each had their own development process, using a traditional ISPF for analyzing, changing and testing applications. In addition, change and configuration management settings in their release management environment, Changeman ZMF, were applied inconsistently. These made it difficult to accelerate application change the business required.

**Solution**
The company used Micro Focus mainframe application analysis tools and its integrated developer environment to reduce delivery iterations, and be more Agile. To establish simple, standardized processes that meant everyone could work in the same, consistent and repeatable way, the organization implemented Micro Focus Enterprise Developer. Using this integrated development environment (IDE), developers improve productivity by taking advantage of modern tooling. Using a common IDE throughout the department also enables greater collaboration between teams because code changes are more visible.

---

Figure 1

Modern development tools give an entire dev team visibility into code changes required, which improves collaboration, productivity and quality.
To identify required changes more easily, Micro Focus Enterprise Analyzer, a mainframe application analysis tool, maps the mainframe application estate. This helps analysts and developers identify application interdependencies and hidden application logic. When making an application change, all of the impacted source code elements are automatically identified, then checked out of the corresponding Changeman ZMF libraries into the off-mainframe Enterprise Developer to perform the work.

As illustrated in Figure 2, once code is changed and compiled locally, developers run debugging scripts and conduct unit testing to verify changes and ensure other changes being fixed in the same environment at the same time will also work without breaking anything else. After all the changes are made and testing is complete, the source elements are checked back into Changeman ZMF.

Fig. 2

Integrating Enterprise Analyzer with Enterprise Developer enables fast, real-time search and discovery so developers can quickly and easily navigate to impacted source lines requiring change.
Results

- Packaging time-consuming, error-prone activities into simple-to-use actions that can be executed from a modern IDE, improved developer productivity.
- The collaborative environment enabled by Enterprise Analyzer and Enterprise Developer makes code changes visible, improving code confidence and quality.
- Waiting for mainframe unit testing resources or resetting test case data is eliminated now that Enterprise Developer enables developers with a local z/OS subsystems on Windows and their own local set of test data.

Simply adopting modern development tools for the mainframe goes a long way toward improving collaboration between development teams and achieving DevOps levels of reliability and speed of delivery.

Removing Barriers to Innovation

There’s more to collaboration than ‘just’ the productivity benefits of delivering faster. According to Forrester, “Ultimately it’s not about delivering old code faster; it’s about removing barriers to innovation in core products and processes.” (Kurt Bittner and Robert Stroud, “Digital Transformation Needs Mainframe DevOps,” Forrester report, 2016)

For many customers, innovating core products and services requires better alignment between mainframe ‘systems of record’ resources with distributed and virtualized, ‘systems of engagement’ resources. New mobile and Cloud-based applications must work well with core mainframe applications to deliver winning innovations. Game designer and author, Jane McGonigal⁹, comments “Collaboration isn’t just about achieving a goal or joining forces; it’s about creating something together that would be impossible to create alone.” She explains collaboration requires three kinds of concerted effort: cooperating, coordinating and co-creating. The most important being co-creating because without it, collaboration is not possible.

So how does that work in the real world? This story highlights how a European managed service provider used a unified development environment to foster co-creation across teams. The result led to achieving new value from their mainframe application services for public and private sector customers in more than 16 countries.

---

⁹ www.janemcgonigal.com/meet-me/
Challenge
The service provider needed to improve productivity between their traditional mainframe team running COBOL, CICS, DB2 and JCL and a Java team working in an Agile, continuous integration environment in developing and testing highly complex mainframe applications. Access to the host mainframe for compilation and testing was constrained and inefficient, mainframe-based tooling impacted project timelines even more. These challenges made prioritizing, planning and delivering customer enhancement requests difficult.

Solution
Implementing a modern mainframe development environment using Enterprise Developer was the first crucial step toward mainframe DevOps. This enabled the service provider to successfully move its traditional mainframe COBOL development to the Agile process used by the Java development team. Establishing a continuous COBOL application integration platform ensured that changes are merged with the application baseline several times a day.

Using Enterprise Analyzer helped this customer to quickly identify the scope and complexity of required application changes so development teams can more accurately estimate the time and resources needed to deliver exceptional services to their customers.

Results
- Visibility and continuous control from analysis and development tools supports collaboration, quality and innovation when co-creating and managing composite applications.
- The complete mainframe application portfolio is managed faster.
- Less dependency on mainframe for development and testing reduces mainframe MIPS consumption—a saving that can be passed onto clients or re-invested in additional mainframe production workloads.
- The ability to test more frequently improves application quality.
- The boost in collaboration improves developer skills and enables easier on boarding for new staff—a benefit that led to a new consultative service offering.
Enabling DevOps Levels of Collaboration

If DevOps is about achieving better business results faster, leveraging more from your core business process and systems of record, the mainframe depends on adopting a collaboration culture across distributed and mainframe development teams.

Micro Focus Enterprise Developer, a modern IDE, unifies distributed and mainframe development teams.

If DevOps is about achieving better business results faster, leveraging more from your core business process and systems of record, the mainframe depends on adopting a collaboration culture across distributed and mainframe development teams.
DevOps Levels of Collaboration: A Self-Assessment

- Do you find it hard to align mainframe related activities with other distributed, i.e. systems of engagement activities?
- Is the communication between teams seamless, or do change requests get ‘lost in translation’?
- Can your Java or .Net developers access mainframe source code to build composite applications or services?
- Can you see enough of your mainframe environment to understand the effects change might have on interdependencies?

Efficiency

Manage Code Changes with Better Processes

Since the mainframe has been around for decades, the technology and processes embedded in mainframe development environments, deployment runtimes and configuration management use a legacy approach. From concept to customer, mainframe application development and release processes are non-continuous, slow and error-prone. Mainframe shops aspiring to deliver faster innovations are time-bound by a linear workflow that does not adequately support multiple development teams in delivering multiple, simultaneous releases.

Branching, merging and labelling is a best-practices method to the challenges imposed by parallel development activities and has been used in DevOps practices as a version control tool. But mainframe development environments are tightly coupled with their SCM tools and these environments are untouched by the emergence of advances in SCM tooling. Developers must manually merge simple and complex code lines. This often leads to broken builds, re-work and frustration. So the challenge is straightforward: to support parallel development and manage, coordinate and orchestrate multiple streams of change easily and quickly without incurring more mainframe resources.
The following use case demonstrates how a wholly owned subsidiary of a large global insurer was able to achieve continuous integration on the mainframe while running the organization’s core insurance policy applications supporting multiple brands. The application portfolio is made up of COBOL, CICS, Batch, DB2, SAS, REXX and Web Services and is maintained and managed through the CA Endevor Software Change Manager (SCM) platform.

**Challenge**

Demands for faster delivery of changes to core applications drove the adoption of parallel development practices. A challenge—delivery bottleneck—emerged at the point of code convergence into a single path to production through CA Endevor. This environment could not easily be extended to match the needs of parallel development at scale. This led to manual and error-prone retrofitting of code changes, especially for large, long-running mainframe application enhancements.

**Solution**

The insurer uses Enterprise Developer coupled with Enterprise Sync. Application sources maintained in CA Endevor are synchronized automatically with source control to a distributed environment where developers manage source change in parallel development streams, using more efficient tooling. This solution was implemented over three months and in three phases, which meant that the distributed Micro Focus tools had to work with the long-established mainframe processes.

During the first phase, developers transitioned from antiquated mainframe development tools to modern Eclipse-based equivalent technology that accessed sources directly on the mainframe. This improved adoption of the modern tools while the development team transitioned into the second phase. This extended the management of sources in CA Endevor to a distributed environment using Enterprise Sync. This source code replication enabled development teams to set up multiple, concurrent source code baselines in a distributed environment and automatically synchronize parallel development activities with CA Endevor. Now, developers use modern source control management tools to effectively manage code changes, merges and retrofits. They are more efficient and parallel streams of development are managed more effectively and with greater flexibility.
Equipping developers with modern and efficient development tools enabled the insurer to implement code changes faster and with greater confidence.

By synchronizing mainframe source control with a distributed environment, developers can use modern tools to automatically detect code conflicts.

**Results**

- Equipping developers with modern and efficient development tools enabled the insurer to implement code changes faster and with greater confidence.
- Better visibility to code change enables earlier detection of quality issues across the development lifecycle.
- By synchronizing mainframe source control with a distributed environment, developers can use modern tools to automatically detect code conflicts, view differences, and perform tool-assisted merges. Retrofitting changes is faster and less error-prone.
Overcoming the limitations of their mainframe delivery bottlenecks, the insurer can now scale parallel streams of development more efficiently while also preserving valuable system change and configuration process settings and mainframe source code integrity.

**Enabling DevOps Levels of Efficiency**

Effectively scaling to DevOps levels of efficiency—managing change faster, without time-consuming or costly rework—mainframe organizations should leverage modern development tools. Micro Focus tools replicate mainframe source code to a distributed SCM platform. Using modern tools, development teams can work collaboratively and deliver more software releases, faster and without restriction. Development teams work in parallel without constraints and with greater confidence in the software releases they deliver.

This unique, low risk approach ensures code changes are automatically synchronized with the mainframe software change and configuration manager (SCCM). This ensures that the mainframe SCCM environment remains the primary system of record.
DevOps Levels of Efficiency: A Self-Assessment

- Is your rate of software delivery sufficient for the business?
- Can your developers use parallel development processes for mainframe application delivery?
- Can you practice continuous integration—are your individual changes integrated early to prevent larger problems later in the delivery cycle?
- Are your development teams having to rework code manually to manage changes along the path to production?
- Is achieving higher level of mainframe efficiency or quality important to the organization?

Flexibility

Scale Development and Test Capacity while Managing Cost

Traditional mainframe delivery methods are based on a sequential, waterfall method for planning and resource allocation. Precious mainframe resources, specifically machine and human, are also time-bound, creating a rigid framework where flex and scale is difficult. Missed delivery milestones mean long delays until mainframe resources are available for dev and test again. This was the case for this large general insurer in the Asia Pacific region. They had a significant technical inventory in their IBM mainframe, including COBOL, Batch, CICS, IMS TM/DB, DB2, MQ, CA Endevor SCM. The core insurance policy application supported multiple brands and companies. Here’s their story:

Challenge

Business demand for new application functionality had increased threefold, adding pressure to deliver new releases faster. Their mainframe lacked the capacity to accommodate the additional test environments required to deliver quality at pace, and acquiring additional mainframe infrastructure to support increased testing was not an option.

Solution

The solution was to implement a mainframe application delivery model using Micro Focus Enterprise Developer and four distributed test regions running under Micro Focus Enterprise Test Server. The result? The increased test capacity enabled the development team to test application change faster and more reliably without relying on mainframe resources.

The process instigated by Micro Focus means a complete test environment is built automatically when a developer commits a code change, or as part of a scheduled process.
1. Sources from a specified level within CA Endevor are synchronized down to a source depot on a Windows build machine running Enterprise Developer.

2. The build machine drives the change process using meta data from CA Endevor SCM processor groups. Application load modules are built in the same way as on the mainframe—essentially the mainframe test environment is emulated in a Cloud.

3. Built load modules are then published along with test data and region configuration to an artefact repository.

4. These artefacts are then deployed automatically to a cloud instance of a Windows-based server running Enterprise Test Server to start a test region.

5. This triggers automatic test scripts from an IBM test automation tool. These run through a set of automated functional and integration test scenarios.

6. Results are harvested and made available to QA teams for investigation. The Cloud test server instance is then decommissioned.

---

Testing that supports core functional and integration work now runs 50% faster in a distributed test environment than the mainframe.

---

Fig. 6

Testing that supports core functional and integration work now runs 50% faster in a distributed test environment than the mainframe.
Enterprise Test Server delivers easily provisioned, ‘on demand’ test environments that do not consume valuable mainframe resources. They are available wherever they are needed, delivering the flexibility and scalability needed to enable mainframe DevOps.

**Results**
- Automatic provisioning of new test regions is cut from six weeks to two hours, while reducing the need for mainframe testing capacity positively impacts on expenditure.
- Testing that supports their core functional and integration work now runs 50% faster in the distributed test environment than the mainframe.
- On-demand testing results in faster feedback and reduces the time to find and fix issues, increasing QA team confidence.
- Today, test server capacity has expanded to 35 virtual mainframe test environments running on Amazon Web Services (AWS) instances. Adding further test environments, each comprising over 30 applications, takes only 2.5 hours to provision.
- The next incremental step in the journey to mainframe DevOps will be decommissioning integration and test regions on the mainframe, so these valuable resources can be dedicated to production workloads.

**Simulated Mainframe Environments for Early Testing**
As parallel development efforts begin to scale, organizations must now test new capabilities quickly.

Increasing test capacity on the mainframe can be challenging; test resources are often limited and priority is given to production workloads. Off-loading early test cycles on a low cost commodity platform removes mainframe resource bottlenecks and accelerates application delivery cycles.

Enterprise Test Server delivers easily provisioned, ‘on demand’ test environments that do not consume valuable mainframe resources. They are available wherever they are needed, delivering the flexibility and scalability needed to enable mainframe DevOps. This was the evidenced by this UK-based global bank with a technical inventory including COBOL, IMS DB/DC, Assembler, DB2, MQ and CA Endevor.

**Challenge**
All UK banks are subject to a Structural Reform Program as part of regulatory changes introduced in 2014. To achieve the specified requirements, application development and maintenance needed to become more Agile—which meant more frequent application testing.
The challenge was that non-functional testing teams had limited access to mainframe test regions. This led to using customized, abbreviated test routines during testing cycles to cope with capacity issues and the time-to-market demands of the business. Unfortunately, these place-holder routines increased the risk of production failures.

**Solution**

The solution was to establish a continuous synchronisation process and a ‘mirror’ of the mainframe components that make up a mainframe application testing environment. The mirrored repository includes mainframe source, database metadata, data, and infrastructure configuration—replicating all relevant levels along the software supply chain.

The mirrored repository is stored on Enterprise Developer, a Windows-hosted build server underpinning a fully automated testing environment.

Jenkins, a self-contained, Java-based continuous build tool, controls the test environment provisioning process. Parameters entered via the Jenkins provisioning profile portal are applied to the relevant infrastructure templates. The resulting configuration file defines a mainframe emulated test environment running on distributed architecture. An automated build process is executed using a Micro Focus build server to produce and deploy the executables for testing.

Data is then imported from the mirrored baseline using the database tooling in Enterprise Developer, whether this is VSAM files, IBM DB2 or IMS databases.

Finally, a mainframe instance is started under Enterprise Test Server and a notification is sent to the testing team that the application testing environment is ready for use. As the bank has discovered, the testing environment can also be purpose-specific, limiting it to just the executables and data required for the use cases under test. This is achieved during the continuous synchronisation process by applying application intelligence extracted from a repository created and maintained by Enterprise Analyzer.
The solution paved the way for the mainframe application development team to adopt DevOps practices including Agile development, continuous integration and continuous deployment to keep pace in the fast-paced world of enterprise application development.

Results

- Using on-demand testing environments enables the non-functional testing teams to test front-end business applications faster, because they are not restricted by mainframe testing capacity resource issues.
- Because Enterprise Test Server can scale to meet test requirements, the on-demand testing environment replaced abbreviated test routines. This improves the quality of the testing, reducing the number of defects found in later test stages of the application delivery lifecycle.
- The solution paved the way for the mainframe application development team to adopt DevOps practices including Agile development, continuous integration and continuous deployment to keep pace in the fast-paced world of enterprise application development.

Enabling DevOps Levels of Flexibility

Because resource priority on the mainframe is given to production workloads, mainframe capacity particularly for development and testing is seldom available. Adding new capacity to the mainframe environment to support DevOps levels of development and testing practices comes with an increased cost to the business—and potentially, another bottleneck. Low risk, distributed testing tools for mainframe environments can help a capacity-strained mainframe team improve application delivery flexibility while maintaining quality.

![Diagram](image)
DevOps Levels of Flexibility: A Self-Assessment

- Can you easily scale dev and test capacity while managing mainframe costs?
- Does mainframe resource availability mean you can only test code changes at certain times of day, or during fixed periods in the week?
- Do you want to take advantage of automatic test provisioning in a virtualized or Cloud environment?

Conclusion

Developers are under pressure to get more done despite limited mainframe time, test resources, staffing and the organizational drive to contain mainframe cost. Faster time-to-market shines a critical light on delivery issues, particularly when business leaders perceive the mainframe to be inflexible and outdated.

Enabling IT to respond faster to change while preserving core business processes and data is a both a strategic win-win and a financially pragmatic way forward. Alternatives, such as application rewrites, are costly, time-consuming and the consequent loss in mainframe IP can reduce an organization’s ability to deliver unique services. Moving faster on the mainframe is the preferred option. It requires greater flexibility, cross-team engagement and of course, modern, integrated tools.

Although the customers profiled here have different setups they can all point to significant improvements in efficiency, collaboration and flexibility—or a combination of all three. They can justifiably claim to have achieved mainframe DevOps, or certainly a similar uptick in performance.

Finding the right fit for your organization is a question of identifying the areas where delivering faster will offer the most value to your business. This will look different to your organization than others in the marketplace. It may be leveraging efficient development tools and processes. Alternatively, you might regard simplifying the process for on-boarding new team members as a strategic must-have. Very few IT organizations will argue against meeting the faster pace of change that IT must now operate.
As referenced in the recent Forrester white paper, digital transformation and mainframe DevOps are very closely aligned. A company contemplating application modernization will be taking similar steps to one assessing their processes and IT infrastructure for mainframe DevOps. Both can expect significant performance improvements in key areas.

Our white paper is a snapshot of Micro Focus customers’ experiences of achieving mainframe DevOps levels of efficiency. Your journey towards improved collaboration, efficiency and flexibility will be driven by your strategic priorities—we have already identified maintaining or increasing market share, reducing costs, ensuring IT better supports the wider business aims—or a combination of all three. Very few organizations could not use the cost and efficiency savings that a well-executed mainframe DevOps strategy will deliver.

But the starting point is the same for everyone. Ask us about our Value Profile Service™, our free consultation and review offering. It is where we assess your IT infrastructure for mainframe DevOps potential and plot your journey to application modernization. As our white paper can confirm, only a series of achievable, sequential steps prevent you from moving from theory to reality.