Ten Steps to True Mainframe Agility
A Phased Approach to Cross-platform DevOps Success

A BMC Compuware Guide to Mainframe Modernization
Compuware joins BMC to empower the next generation of developers to mainstream the mainframe. With unparalleled mainframe operations management and agile application development and delivery, BMC Compuware provides a mainframe-inclusive DevOps toolchain that increases volume and velocity.

Bring apps that run on the mainframe to market faster with increased quality throughout the process of analyzing, building, testing, deploying, monitoring, and tuning apps and services. Our combined offering accelerates each client’s evolution to an Autonomous Digital Enterprise.
What This Approach Means to You and Your Company

Enterprises must quickly and decisively transform their mainframe practices. The slow, inflexible processes and methods of the past have become intolerable impediments to success in today’s innovation-centric digital markets. IT leaders must therefore bring the proven advantages of Agile, DevOps and related disciplines to bear on the mainframe applications and data that run their business.

But how? And where to begin? Mainframe transformation can seem overwhelming. And no IT leader wants to embark on a “boil the ocean” project that consumes resources and generates risk without a high probability of sizable, near-term concrete rewards.

This eBook addresses these concerns by spelling out a proven, phased approach for measurably modernizing mainframe practices. By following this approach, IT leaders can rack up high-impact, near-term “wins” at each stage along the way, while staying on course towards a vital strategic objective that can be fully achieved in less than two years.

Because every organization has its own existing processes, tools and culture, this mainframe transformation game plan can be modified to fit each organization's specific needs.

No company, however, can afford to further delay mainframe transformation—or embark on such a transformation without a clear plan. To remain competitive in an app-centric economy, mainframes must be as adaptive as other platforms. And enterprise IT must be able to manage DevOps in an integrated manner across mainframe, distributed and cloud.

This eBook lays out a plan for doing exactly that.
STEP 1: 
Determine Your Current and Desired State

Before embarking on the process of mainframe transformation, it’s wise to first be clear about what that transformation will entail. To map out a transformation plan that all relevant stakeholders can understand and buy into, you will need to:

Document and assess your current state.
What tools do your development, QA, and ops teams currently use? What does your software delivery process look like? What are the results in terms of velocity, frequency, person hours, and quality? How consistent or variable are those results? How much institutional knowledge is isolated in the minds of your top SMEs? How well do your mainframe teams collaborate and coordinate with their counterparts working on your distributed and cloud platforms?

Identify impediments.
Many mainframe teams face technical obstacles, such as tools that lock them into slow, waterfall processes. Entrenched habits and work culture—such as insufficient emphasis on speed and collaboration—can also tangibly hinder mainframe transformation. Or you just may be under-funded. Your gap analysis should zero in on these specific impediments preventing you from achieving your specific goals.

Define your desired state.
Prioritize the goals that are most important to your organization. These can include velocity (compressing the time required to go from business requirement to code in production), agility (the ability to make smaller, more frequent changes to application code), efficiency (reducing cost through better use of person hours), integration (better coordination of code changes across platforms), and generational shift (empowering technical staff with mainstream skillsets to assume responsibility for mainframe DevOps). These goals must obviously be achieved without compromising the reliability and stability of core applications.

Define your customized plan.
Once you know where you want to go and what’s currently preventing you from getting there, you can craft a rational, credible transformation plan. Your plan will likely look very similar to the nine steps that follow. Depending on the particulars of your situation, however, you may need to prioritize certain steps or allocate more resources to certain aspects of your transformation.
STEP 2: Modernize Your Mainframe Development Environment

Most mainframe development is still performed in antiquated “green screen” ISPF environments that require highly specialized knowledge and problematically limit new staff productivity. Modernizing the mainframe begins with modernizing this developer workspace.

A modernized mainframe workspace should possess the look and feel of the Eclipse-style IDEs that have become the de facto standard for other platforms. This user-friendly interface will allow staff at all experience levels to move easily between development and testing as they work on both mainframe and non-mainframe applications. Your modernized mainframe IDE will also support a complementary palette of value-added tools as you continue your mainframe transformation through its subsequent steps.
Tools

The enabling technology for Step 2 is **BMC Compuware Topaz Workbench**—along with associated BMC Compuware solutions such as **BMC Compuware File-AID**, **BMC Compuware Abend-AID**, and **BMC Compuware Xpediter**.

These products provide complete source code editing, debugging, fault diagnosis, and data browse/edit/compare functionality. Topaz Workbench, in particular, allows developers to write, compile, and run code all from a modern Eclipse-based IDE. It also brings drag-and-drop ease to copying files between LPARs and other common developer tasks. This puts the mainframe development experience on par with other technologies in the enterprise such as Java.

Find more information:
**BMC Compuware Topaz Workbench**

Success Indicators

- **Empirical productivity metrics** such as features delivered, code commit frequency and shorter learning curve for new employees
- **Positive anecdotal feedback** from development and testing teams
- **Ability to recruit and train individuals with no mainframe experience to work on mainframe applications**
STEP 3:

Adopt Automated Testing

Unit testing is central to Agile. Frequently testing small increments of code enables developers to quickly and continuously assess how closely their current work aligns with immediate team objectives—so they can promptly make the necessary adjustments or move on to the next task.

Unfortunately, technical obstacles have historically prevented the kind of automated unit testing that is commonplace in Java from being applied to mainframe development. Now that those obstacles have been removed as described below, reliable automated unit testing can become a mainframe reality.

Of course, effective unit testing requires more than technology. Mainframe developers not accustomed to unit testing must learn how to best leverage the practice to work much more iteratively on much smaller pieces of code. One particularly effective way to accelerate adoption of unit testing best practices across your development teams is to monitor the percentage of code that has been subject to automated testing. By combining automated unit testing with code coverage metrics, you can build confidence among your developers that they can make incremental changes to mission-critical applications without jeopardizing quality. It’s also important to implement controls that ensure unit testing is successfully completed before promoting code. But automated unit testing across all platforms is a fundamental requirement for Agile.

Once you have completed the unit testing phase, you can work through the functional testing phase. Functional testing validates that the implementation works as specified in its requirements. This is different from “the code works correctly,” which is determined during unit testing.

After functional testing, you can then start the integration testing phase. With integration testing, you evaluate if the collaboration between two or more programs works as expected. This extends the functional testing that focuses on testing the specifications of one program to test the interaction between several programs.
Tools

The enabling technology for Step 3 is **BMC Compuware Topaz for Total Test**. The product automatically scans mainframe code and creates appropriate unit and functional tests based upon the program's structure. It can automatically create these test cases for both main programs and subprograms. Test cases can include test data and a set of default test result assertions. This automated test creation empowers developers at all skill levels to quickly, easily and accurately validate and troubleshoot any changes they make to mainframe applications.

Topaz for Total Test now offers functional and integration testing capabilities to extend the use of the solution further into the application development lifecycle. This provides additional levels of test automation to improve velocity to production use of application changes.

Find more information: [BMC Compuware Topaz for Total Test](#)

Success Indicators

+ **More frequent drops of code** updates required by the business
+ **Fewer errors found later** in the DevOps lifecycle (“shift left”)
+ **Lower cost of defect resolution** (since resolution costs are much lower earlier in the DevOps lifecycle)
+ **Tighter synchronization** of in-tandem mainframe/non-mainframe development
STEP 4: 
Provide Developers with Graphical, Intuitive Visibility into Existing Code and Data Structure

As mainframe applications have been expanded and enhanced over many years, they have typically become quite large and complex. They are also typically not very well-documented. This combination of complexity and poor documentation is a major impediment to core transformation goals—including agility, confidence, and efficiency. In fact, the undocumented idiosyncrasies of mainframe applications and data structures almost universally make enterprise IT highly dependent on the personal/tribal knowledge of senior mainframe staff. Worse yet, if a seasoned mainframe developer is no longer available, IT may be fearful of making any changes at all.

To overcome this dependency, you have to make it much easier for any new participant/contributor to quickly “read” existing application logic, program interdependencies, data structures, and data relationships. Developers and other technical staff also need to be able to understand application runtime behaviors—including the actual sequence and nature of all program calls as well as file and database I/O—so they can work on even the most unfamiliar and complex systems with clarity and confidence.
**Tools**

The enabling technologies for Step 4 are **BMC Compuware Topaz for Program Analysis** and **BMC Compuware Topaz for Enterprise Data**. Their unique, powerful visualizations reveal underlying program logic and data relationships through dynamically generated, graphically intuitive diagrams. These diagrams show how COBOL and Pl/I programs flow with the associated variables and files, while also enabling developers to play, save, and compare visualizations of application runtime behaviors—without requiring access to current source code files.

Find more information:
- [BMC Compuware Topaz for Program Analysis](#)
- [BMC Compuware Topaz for Enterprise Data](#)

**Success Indicators**

- **Reduction in amount of time** it takes a developer to understand large/complex mainframe applications
- **Experienced developers gaining further insight into runtime behavior of mainframe applications**
- **Developers better estimating project work** required to meet delivery deadlines
Successful mainframe transformation demands rigorous, reliable, and early detection and resolution of quality issues. There are three primary reasons for this. First, mainframe applications often support core business processes that have little to no tolerance for error. Second, in transitioning from waterfall to Agile delivery cycles, continuous quality control reduces costs and prevents even relatively minor errors from adding friction that undermines the goal of faster, more streamlined application updates.

Third—and of particular importance at this moment in the history of the mainframe—a new generation of developers with less mainframe experience and expertise are being called upon to maintain and evolve mainframe applications. These developers must be supported with quality controls and feedback above and beyond the automated unit testing adopted in Step 3.

Every effort must therefore be made to rigorously safeguard application quality as the mainframe becomes more agile. Continuous Integration (CI) is especially important in this regard, since it ensures that quality checks are performed continuously as your code is updated.

Also, with the right quality control tools and processes, you can do more than just catch and fix individual issues early in the cycle. You can also capture KPIs that give you clear visibility into individual, team, and project quality metrics so you can quickly pinpoint issues requiring additional coaching and training—allowing you to continuously improve development performance and productivity.
Tools

As with Step 3, a core technology enabler for Step 5 is **BMC Compuware Topaz for Total Test**—which provides a single point-of-control for developers to manage their testing activities, while also tracking code coverage to ensure consistent conformance with testing best practices prior to code promotion.

Topaz also provides integrations with **SonarSource’s SonarLint** and **SonarQube**. SonarLint integrates into the Topaz Workbench environment to give developers on-the-fly feedback on any potential new bugs and quality issues they may inject into their code. With SonarLint, even mainframe-inexperienced developers can quickly be alerted to application quality issues such as unbalanced or unmatched working storage/data items and sections of code that are too complex. SonarQube is a feature-rich dashboard for tracking quality issues, code coverage from automated tests and technical debt—all of which are useful for capturing the kinds of KPIs needed to ensure deliverables are being created with sustainable and high-quality methods.

**Jenkins**, the open source automation server, is also typically important for this step since it provides Continuous Integration functionality. Jenkins can also automatically drive execution of any essential quality checks—such as static code analysis, automated unit and functional tests and measuring code coverage—that you want to ensure are performed on every change to code.

**BMC Compuware zAdviser** further aids in the use of KPIs to drive improvements in mainframe DevOps performance by both capturing developer behaviors and benchmarking those behaviors against metrics captured from across a broad range of other organizations engaged in mainframe agility initiatives.

Find more information:
**BMC Compuware zAdviser, SonarQube, SonarLint, and Jenkins.**

Success Indicators

+ **Increasing code coverage** exercised by automated tests
+ **Positive trends in quality** and reduced technical debt metrics
+ **Reduced number of error-related cycles** between testing and development teams
+ **Quantifiable improvements in the performance and productivity of new mainframe developers**
STEP 6:
Initiate Training in and Adoption of Agile Processes

At this point in your journey, you should have the right development environment in place—so your development teams will be ready for actual training on Agile development methodologies. Once completed, you’ll be able to start shifting your process from a traditional waterfall model with large sets of requirements and long project timelines to a more incremental model. The goal is to have developers for mobile, web, and mainframe components collaborate on a single Scrum team. Teams are focused on stories and epics that capture specific units of value to your business versus technical tasks in a project plan. By estimating the size of these stories and assigning them their appropriate priority, your teams can start engaging in Agile processes that allow them to quickly iterate towards their goals.

The move from large-scale waterfall projects to Agile represents a significant change in work culture for most mainframe teams. Training in Agile processes and work culture is therefore a must. Technical Leadership roles and Product Owners, in particular, need in-depth training and coaching. However, all team members should have at least some formal introduction to basic Agile concepts—especially if they’ll be expected to read Scrum or Kanban boards.

You may want to build your initial Agile mainframe team by choosing select mainframe developers and pairing them with Agile-experienced developers from other platforms. You may also want to consider how you’ll factor conformance to Agile values such as transparency, knowledge sharing and ideation into your developer performance reviews.
Tools

Enabling technologies for Step 6 include modern collaboration platforms and Agile project management tools. **Atlassian Jira**, for example, is an Agile task management tool that supports Agile methodologies such as Scrum and Kanban. It enables you to plan, track, and manage all Agile development activity, so you can keep your teams on track and continuously improve Agile adoption in terms of speed, efficiency, quality and—most importantly—ongoing alignment with your most urgent business needs.

**Atlassian Confluence** complements Jira by providing a centralized, well-organized web collaboration space where your Agile teams can easily and flexibly share ideas and product requirements as well as provide process and status updates, etc. This type of collaboration supports the required culture of innovation.

It’s typically wiser to leverage these kinds of popular best-in-class tools than it is to adopt a monolithic approach that requires you to perform all SDLC activities within a single vendor’s solution set—since going forward you’ll want to avoid vendor lock-in and ensure your ability to take advantage of the latest innovations in Agile management.

Find more information: **Atlassian Jira** and **Confluence**.

Success Indicators

+ Target percentage of **dev/test staff completes Agile training** with goal of 100% trained
+ First delivery of artifacts from initial Agile teams and discovery of technical and cultural obstacles to broader Agile adoption
+ **Evidence of cross-team collaboration** and business participation in Agile process
STEP 7: Leverage Operational Data Across the Development, Testing and Production Lifecycle

To ensure that your applications will perform optimally in your production environment, it’s not enough to just write good code. You also need to understand exactly how your applications behave as they consume processing capacity, access your databases and interact with other applications.

One good way to gain this understanding is to leverage operational data continuously throughout the DevOps lifecycle. This provides dev and ops teams with a common understanding of the operational metrics/characteristics of an application throughout its lifecycle, helping them more fully and accurately measure progress towards team goals. Early use of operational data can also dramatically reduce your MIPS/MSU-related costs by allowing you to discover and mitigate avoidable CPU consumption caused by inefficient code.
Tools

The enabling technologies for Step 7 are Abend-AID and BMC Compuware Strobe. Abend-AID provides source-level analysis of application failures—sparing developers the time-consuming work of manually cross-referencing numerous pages of memory dumps, source listings, and application code. With Abend-AID, dev/test staff can quickly pinpoint bad statements, identify data issues, and isolate many other types of application problems.

Strobe pinpoints application inefficiencies such as bad SQL statements, wasteful Db2 system services that cause excessive CPU consumption, slow data retrieval and other issues that add cost while undermining the end-user experience. By automatically identifying these issues, Strobe enables even novice dev/test staff to contribute to faster delivery of better-performing applications.

Find more information:
BMC Compuware Abend-AID
BMC Compuware Strobe

Success Indicators

+ Early detection of avoidable CPU consumption
+ Reduction in abends in production
+ Reduction in average cost per error and mean time to resolution

For more information on transforming your operations team, check out a companion pieces in our DevOps guidebook series entitled 9 Steps to Agile Ops.
STEP 8:
Deploy True Agile/DevOps-enabling Source Code Management

Traditional SCM environments are inherently designed for waterfall development and are thus incapable of providing essential Agile capabilities—such as parallel development work on different user stories, quick compare and merge of different versions, and conditional pathing of code promotion.

But to truly enable Agile and DevOps on the mainframe, your SCM must do more than just provide automation, visibility, and rules-based workflows to your SDLC. It must also integrate easily and seamlessly with other tools in your end-to-end toolchain. Chances are that your development, test, and ops teams will want to use some combination of Jenkins, XebiaLabs, Slack, CloudBees and/or other popular tools. You must therefore be able to move data easily between these tools—and trigger automated actions, messages, and alerts between them. Ideally, your new SCM will do this with industry-standard REST APIs and webhooks, which provide the simplest means of doing so and give you the greatest flexibility to allow your next-gen developers to work with whatever their personal tools-of-choice happen to be at any points in time.

The shift from waterfall-based SCM to Agile-enabling SCM is a pivotal moment in any mainframe transformation, and it should be carefully planned to avoid disruption to current work in progress. It is, however, an absolutely essential shift if your goal is to increase the speed and frequency of new mainframe code drops, optimize developer productivity, and simplify end-to-end management of your SDLC.
Tools

The enabling technology for Step 8 is **BMC Compuware ISPW**. ISPW is a modern, end-to-end Agile SCM and release automation solution that enables mainframe application developers at all skill levels to fulfill business requirements, optimize code quality, and improve developer productivity through mainframe DevOps. It provides automated change management that eliminates manual steps and empowers your teams to iterate quickly through the development, test, and QA cycles. ISPW is fully integrated into the Topaz Workbench environment, so developers have full access to all of their tools from a single, intuitive interface. It also integrates with other popular tools, including Jenkins, as described above.

Find more information: [BMC Compuware ISPW](https://www.bmc.com/products/compuware-ispw). BMC Compuware also offers **SCM Migration Services** to speed and de-risk this critical conversion.

### Success Indicators

+ Developers no longer working with “personal libraries” and different Agile teams working on different stories in parallel
+ Developers with different mainframe skills levels are able to quickly understand the scope of their changes before they begin to code
+ Reduction in code approval delays
STEP 9:
Automate Deployment of Code into Production

Agile development alone is insufficient for full digital business agility. To keep pace with today's fast-moving markets, your business must also be able to quickly and reliably get new code into production. That means automating and coordinating the deployment of all related development artifacts into all targeted environments in a highly synchronized manner. You'll also need to pinpoint any deployment issues as soon as they occur, so you can take immediate corrective action.

And, if such corrective action is not immediately evident or doesn't immediately produce its expected remediative effect, you have to be able to quickly and automatically fallback to the previous working version of the application. This automated fallback is, in fact, a key enabler of rapid deployment—since it is the primary means of mitigating the business risk associated with code promotion.
Tools

The enabling technology for Step 9 is also ISPW. ISPW complements its core SCM capabilities with advanced mainframe deployment features that empower you to rapidly move code through the deployment process—including test staging and approvals—while also providing greatly simplified full or partial fallbacks. As such, it offers a significantly superior solution to traditional “homegrown” scripts, which do not provide essential capabilities such as fallback, progress visibility, and auditability of deployment processes.

ISPW also provides visualization that enables DevOps managers to quickly pinpoint deployment issues in order to both solve immediate rollout problems and address persistent bottlenecks in code promotion.

ISPW features a mobile interface that enables mainframe DevOps managers to respond immediately when alerted that code changes are ready for approval. This anytime/anywhere mobile management eliminates a common cause of mainframe code promotion delays.

Find more information:
BMC Compuware ISPW

Success Indicators

+ Faster rollouts of applications into production
+ Reduction in code promotion failures
+ Reduction in personnel effort and system down time caused by need to revert after failed deployment
STEP 10: Enable Coordinated Cross-platform Continuous Delivery

Mainframe applications and data increasingly serve as a back-end resource for multi-platform customer- and employee-facing applications that include mobile, web and/or cloud components. DevOps teams must therefore be able to fully synchronize the delivery of new, approved code across all platforms. These deployment controls should also provide unified, cross-platform fallback, and progress reporting.

This is the target state of enterprise DevOps at the completion of Step 10: A de-siloed environment where the mainframe is “just another platform”—albeit an especially scalable, reliable, high-performing, cost-efficient, and secure one—that can be quickly and appropriately modified as needed to meet the needs of the business by whichever staff resources are available to do so.
Tools

The enabling technologies for Step 10 include ISPW REST APIs and integration with tools like Digital.ai Release, CloudBees or CloudBees Flow. ISPW integrates with distributed tools to provide a single point of control for all changes across z/OS, Windows, Unix, and other platforms. REST APIs are especially important for ensuring full flexibility to mix and match best-in-class tools and avoid vendor lock in.

XL Release is a highly advanced application release automation solution that eases the planning, automation, and analysis of cross-platform software releases so you can streamline code promotion, maintain full visibility into release progress, pinpoint release problems and address any chronic bottlenecks in your DevOps processes.

Find more information about our integrations: Digital.ai Release, CloudBees and CloudBees Flow

Success Indicators

+ Ability to work on related code on multiple platforms in parallel
+ Increased communications and collaboration between previously siloed developers with different skill sets
+ First successfully automated cross-platform release fallout
Summary

**STEP 1:**
Determine Your Current and Desired State

**STEP 2:**
Modernize Your Mainframe Development Environment

**STEP 3:**
Adopt Automated Testing

**STEP 4:**
Provide Developers with Graphical, Intuitive Visibility into Existing Code and Data Structure

**STEP 5:**
Empower Developers at All Skill and Experience Levels to Deliver High-Quality Code with Less Friction

**STEP 6:**
Initiate Training in and Adoption of Agile Processes

**STEP 7:**
Leverage Operational Data Across the Development, Testing and Production Lifecycle

**STEP 8:**
Deploy True Agile/DevOps-enabling Source Code Management

**STEP 9:**
Automate Deployment of Code into Production

**STEP 10:**
Enable Coordinated Cross-platform Continuous Delivery
The Ongoing Pursuit of Digital Excellence

The evolution of your mainframe doesn’t stop once you achieve the desired state of agility and cross-platform integration of your DevOps workflows. In fact, you’ll probably want to build upon that achievement to further enhance your company’s digital agility and efficiency over time.

One especially compelling way to do that is by providing your IT service management (ITSM) team with a unified environment for both mainframe and non-mainframe applications. This unified ITSM model will become increasingly useful as more of your company’s digital value proposition is based on code that traverses multiple platforms—from back-end mainframe systems of record to customer-facing web and mobile apps.

**BMC Compuware Topaz Connect** provides this kind of cross-platform ITSM integration by unifying third-party ITSM solutions such as ServiceNow, BMC Helix, Tivoli, and CA. Through this integration, ITSM staff can track processes for mainframe applications in the same manner as they do for other hardware and software platforms. If an ITSM change request requires a code change, the specifics of that change are automatically communicated to ISPW. And as those modifications are delivered, your ITSM environment can track the progress of the workflow right through to deployment.

You can also use a solution like **Digital.ai Release Dashboard** to both maintain real-time insight into your active projects across platforms and track KPIs that help you pinpoint opportunities for further improvement—whether it’s the time it takes your smallest incremental changes to get through QA or the types of technical issues that seem to be driving the most chatter in your collaboration tools.

**zAdviser** can also help you pinpoint opportunities for continuous improvement—especially as its benchmarks reveal how your peers at other organizations keep raising the bar for mainframe agility over time.

---

**The Transformation Imperative**

IT leaders are under intense pressure to deliver on many fronts. They have to deliver new AI-enabled systems. They have to fulfill the escalating mobility expectations of customers and employees. They have to safeguard the enterprise from an ever-intensifying range of threats. And they have to do all this within extremely challenging capex and opex budget constraints.

Mainframe transformation, however, is central to the success of all these efforts and more. If your core systems of record aren’t agile—and if you’re not fully prepared to extend the useful life of those systems of record well into the next decade, even as you lose your current cohort of mainframe veterans to retirement—then your other efforts can only deliver limited benefits. The performance of your business will ultimately be constrained by the constraints of your mainframe environment.

The good news is that best practices, modern tools and committed partners are now available to assist you in your efforts. All you need is a decision and a plan. Then you can start.
About BMC
From core to cloud to edge, BMC delivers the software and services that enable over 10,000 global customers, including 84% of the Forbes Global 100, to thrive in their ongoing evolution to an Autonomous Digital Enterprise.

BMC—Run and Reinvent

www.bmc.com