FAST AND FEARLESS CHANGE FOR MAINFRAME APPLICATIONS—WITHOUT REPEATING RISKS

Just like the *Fast and Furious* movie franchise, it seems like mainframe modernization is a predictably repeated story that never runs out of sequels in most enterprises, even if the plot is pretty much the same every time.

Organizations delay changing and evolving their application estates because developers fear they will break systems they don't understand, and IT operators fear what will happen if live mainframe production environments are disrupted in any way.

Regardless, enterprises must realign their most critical core systems to keep up with the pace of change required to become more agile and responsive to customer needs, or they will risk losing customers. Instead of fearing change on the mainframe, we should fear not changing fast enough.

Breaking the calcification of COBOL

As an early programming language, COBOL isn't particularly complex in and of itself. There's a rather limited instruction set, and when most COBOL code was written, there was no need to even think about object orientation—much less extended libraries and frameworks or a distributed cloud architecture.

The calcification that prevents refactoring doesn't come from COBOL, it comes from the fear of change: the risk of layering on each additional procedure and workaround, in order to add just one more feature to the monolith.

As new code is appended one function at a time over the years to adapt to change, mainframe-

based applications start to accrete technical debt and become more unstable and harder to change without failures.

Worse still, at this point, nobody knows the intent of the original builders, as they have likely already moved on or retired from those roles long ago, without leaving behind adequate documentation.

Rather than having to check out, review, and rebuild the whole program every time a change is needed, what if we could instead refactor mainframe programs into separate logical components? Then, even an individual developer could check just one component of the whole program out, and develop and test a new feature in isolation, away from the entire monolith.

Finally, we may be able to realize the dream of agile development on the mainframe, without the fear of failure.

Refactoring monoliths with BMC AMI DevX Code Insights

Calling up mainframers from retirement is probably not feasible, and even if an organization has retained COBOL developers, they are likely to be in short supply. Enterprises need a way to convey the intent of those original, expert builders into a system that is more object-oriented and understandable to modern development teams.

BMC has released **BMC AMI DevX Code Insights**, a solution that allows developers to scan an entire mainframe application for business logic and structure, and then extract a limited section of the monolithic program for isolated development, testing, and replacement.

The extracted section here is a *subprogram* with discrete inputs and outputs, a truly modular component that developers can work with in parallel to other teams working on their own subprograms. Subprograms can even be called on via APIs and exposed to external services.

Rather than needing to check out the whole stack, subprograms undergoing development and testing can be inserted into teams' DevOps, and GitOps automation scripts and delivery tools of choice. This enables shift-left testing practices, a core tenet of modern agile development that discovers bugs and errors earlier, when they are far less labor-intensive and costly to fix.

Figure 1. Runtime Visualizer feature within BMC AMI DevX Code Insights.

Of course, even if we can isolate certain features of the monolith as subprograms, that doesn't mean the original business logic was simplistic, especially if it was a section of the application that was updated several times with complex IF/THEN/GOTO statements and looping returns.

Here's where a little sensory enhancement can save the day. BMC AMI DevX Code Insights includes a Runtime Visualizer feature that charts out a call-by-call working model of the running mainframe

application, including inputs and outputs, files and database access, which can be replayed any time or watched in real time.

Rather than uprooting the whole tree trunk, developers can just snip off the logical branch they need to work with.

Closing the DevOps feedback loop with BMC AMI zAdviser Enterprise

Another core tenet of DevOps is continuous measurement and optimization of the entire software delivery and operations capacity of the organization. To achieve this goal, companies are looking at key performance indicators (KPIs) for measuring the quality, velocity, and efficiency of mainframe development activities in relation to the organization's broader digital transformation efforts.

DevOps Research and Assessment (DORA) metrics, based on a long-running research project started by Google and several other leading tech vendors, are becoming a commonly accepted framework for measuring success. This model sets out four key metrics:

- **Deployment frequency:** How often an organization successfully releases to production
- **Lead time for changes:** The amount of time it takes for a commit to get into production
- **Change failure rate:** The percentage of deployments causing a failure in production
- **Time to restore service:** How long it takes an organization to recover from a failure in production

Notably, these DORA metrics can be considered independent of the underlying technologies that DevOps teams are working on, so it is completely reasonable to expect we should be able to realize similar KPIs for mainframe development and operations.

BMC is addressing this continuous improvement challenge with its new **BMC AMI zAdviser Enterprise** analytics solution, which uses machine learning (ML) to collect and process telemetry data from live mainframes, as well as third-party IT service management (ITSM) and source code management (SCM) dashboard data, and events from within the continuous integration and continuous deployment (CI/CD) pipeline.

Incoming BMC AMI zAdviser Enterprise data is processed by an applied artificial intelligence (AI) model that summarizes the KPIs of an organization's mainframe activity and provides a unified progress report and recommendations for improvement. Mainframe developers can zero in on individual application delivery events to figure out exactly when and where an anomaly occurred.

While the metrics are based on DORA, there are certain measures that are esoteric to mainframes. For instance, if an abend occurs in a particular IBM® CICS® region during pre-production testing, the developer would want to know whether that is a bug that is reproducible in production and get a recommendation about prioritizing the urgency of a resolution based on the potential impact or risk.

To tie it all together, BMC AMI zAdviser Enterprise also offers a refactoring candidates dashboard that highlights the programs exhibiting the most checkouts and problems, so that BMC AMI DevX Code Insights users can identify the next best program to start work on.

The Intellyx Take

"I must not fear. Fear is the mind-killer. Fear is the little-death that brings total obliteration. I will face my fear. I will permit it to pass over me and through me. And when it has gone past I will turn the inner eye to see its path…"

Frank Herbert, Dune

When mainframe teams gain insight into the levers they can use to break up a complex system into manageable parts, they soon realize there was no reason to fear change or slow down continuous improvement, just for COBOL's sake.

New mainframe modernization solutions like BMC AMI DevX Code Insights and BMC AMI zAdviser Enterprise are rapidly transforming the mainframe into a true first-class citizen of the DevOps lifecycle.