MAKING MAINFRAME GIT LESS DAUNTING

Mainframe and Git were words that didn't seem to go together not that long ago. Git's popularity among developers in the distributed world is no mistake. Feature branches, distributed development, pull requests, and a stable community ultimately result in a faster release cycle. These capabilities facilitate an agile workflow where developers are encouraged to share smaller changes more frequently. In turn, changes can get pushed down the deployment pipeline faster than the monolithic releases common with centralized version control systems. And Git's rich collaboration features enable changes by multiple people to be reviewed and merged into one source. Coupled with newer developers already being familiar with using it, Git makes a compelling solution to consider for source code management (SCM).

The concept has arisen that because Git is so successful in distributed development, perhaps that success could carry over to the mainframe. Sharing of practices is a way to provide commonality and breaks down silos. Also, there is a desire to have all assets in one place. But as often the case, meshing technologies between platforms can be challenging.

But it doesn't have to be. BMC AMI DevX Code Pipeline's deploy capability integrates with multiple vendors to provide the unified deploy across platforms. This same focus on integration, and our experience in working with customers using Git over the past 7 years, have made it increasingly easy to provide an option for Git to store your development assets while still leveraging the power of DevX Code Pipeline on the mainframe for your build and deploy.

Here is our advice for those considering Git on the mainframe:

- Explain the rationale – Part of planning for Git is to provide the rationale. Why would teams want to consider moving to Git? Our Git for the Mainframe eBook offers a comprehensive look

at Git and BMC AMI DevX Code Pipeline. It can help developers understand the issues and the benefits to make an informed decision for their teams.

- Gradual Transition – Part of making this easy is to work on your schedule. Teams can move over to Git when they are ready (and some teams may never go to Git - an important point). Resistance is understandable if all applications must move at one time, which can be disruptive. Instead, we recommend a gradual approach where applications are moved to Git when team members are ready.
- Guidance – We have defined procedures to take you step-by-step. Even tens of thousands of programs will be manageable with this application-by-application approach. Our tools can automate the transfer from DevX Code Pipeline to Git. We can even assist you with copying your change history from ISPW over to Git.
- Build – When you have DevX Code Pipeline installed and configured, everything is there for your build and deploy. The relationships are built automatically, so all of your compile parameters and impacts are there to be leveraged.
- Deploy – With DevX Code Pipeline your mainframe deploys will be configured and available to work with Digital.ai, Cloudbees Flow, and HCL Launch. Switching to Git for SCM won't cause any disruptions for deploy.
- Related Tooling – One of the reasons to use Git is to be able to use the related ecosystem of tools. BMC AMI DevX Code Pipeline has a VS Code DevX Code Pipeline extension to take you from editing your code into a mainframe build with one click. We also offer GitHub actions to automate generate, build, and deploy using DevX Code Pipeline on the mainframe.

It is good to have options as you plan your future. With BMC AMI DevX Code Pipeline we understand that and strive to stay current with trends, to be just ahead of where you are. This provides you with what your developers need, when they need it. You will be able to move forward on your initiatives with confidence.

To learn more about pairing Git with BMC AMI DevX Code Pipeline, listen to our podcast, *Git for Mainframe DevOps*.