DBB Developer for z Systems DevOps Z Development & Test Environment Z Open Development Z Trial

DevOps transformation on IBM Z with Dependency Based Build : a case stuc

francois.dumont
Published on 03/25/2019 / *Updated on 03/25/2019*

2

According to Puppet and Splunk's 2018 State of DevOps Report, successful DevOps transformation results in 7x lower software delivery change failure rate. From this perspective, a large European bank organization presented last month at IBM Think San Francisco its Mainframe DevOps transformation program, focusing on software quality improvement, through a "test early and often" approach, as well as a culture of empowerment for several hundreds developers distributed over several suppliers.

The pain points this mainframe organization met, as well as the imagined solution, is representative enough of the common challenges of DevOps transformation on IBM Z to be shared beyond the audience of Think conference.

The Bank is facing the problem of a large number of developers working in a single development partition, with a single set of DB2 tables and usually obsolete LOAD library making testing difficult is this environment, thus encouraging developers to request deployments to the certification one, creating frequent workflow congestion and incidents.

The first way to address this issue consist in setting up a tool chain that can be provisioned on demand for every service supplier and consumer to develop and test in isolation, while clearly marking out the respective responsibilities through an acceptance mechanism. The second change is about providing new means to drive developments through real-time delivery quality reporting, performed by test automation part of a continuous integration process.

The technical infrastructure realizing this is composed of IBM Developer for z Systems (IDz) installed on a Virtual Desktop. Virtualized LPARs are hosted by a series of IBM Z Development & Test (ZD&T) instances running on Linux Virtual Machines, dedicated for the various build and testing levels.

Selecting test data is key for an efficient Unit Testing strategy. The Bank is using IBM Infosphere Optim Data Management solutions to do so, and that way controls both the right granularity level and pertinence of test data.

Builds for single programs, as well as complete applications are realized by IBM Dependency Based Build, triggered either from IDz or from the Jenkins Build Orchestrator.

Isolation principle for testing is also generalized to development, by enabling true parallel software configuration management with Git, initially loaded from CA Endevor, which will remain the production and deployment repository in the first place. Most of the implementation choices for Git are discussed in a white-paper, co-written by IBM and the Bank, that can be downloaded here.

The key benefits of the solution can be summarized that way :

1. Each service provider benefits from an autonomous ecosystem and organizes its work according to its own organization.
2. There is a real convergence of the Mainframe and Open worlds. It’s even now expected Java developers to come and develop with COBOL, since the main difference is now only the language.
3. Developers and testers have means to find, extract, and load the data of the test cases that they want to run without depending on other people.
4. Modern development environments are provided, similar to the ones from distributed developments world. A minimum productivity gain of 15% is expected.

5. Easy to use and modern Software Configuration Manager helps to meet the business expectations in terms of time to market.
6. Automated testing capabilities on a Continuous Integration Platform for the Mainframe are now available.

IBM and the Bank collaborated efficiently to make this story happen. More specifically, the latest enhancements contained in IBM Z Open Development 1.0.1 and Dependency Based Build (DBB) 1.0.4 (GA 03/15) such as User Build Interface Enhancements, and several other performance improvements result from this collaboration.

It is also to be noticed that DBB 1.0.4 now supports Assembler, compilation input and outputs on USS, and provides automated migration tooling from IBM SCLM to Git.

Now IBM Z Open Development can be evaluated within a zero-install environment including Jenkins and Git running on IBM Z Development & Test with the IBM Z Trial Program.
Z Trial Program is completely free and can be available in two hours for three days.
You can learn Z Open Development capabilities through four scenarios including hands-on tutorials that explain how to:
Load a Git repository and modify the COBOL code
Run a build with IBM Dependency Based Build (DBB) and visually debug the program.
Commit and push modifications into the Git repository and request a Jenkins build.
Write a sample build script using DBB.

Interested in Z Open Development to begin your DevOps journey with Git? Register Z Trial Program for Z Open Development and get started today! Trial Version can also be dowloaded here.

---

by francois.dumont

Twitter

Francois Dumont is an offering manager at IBM, in charge of Rational Programming Patterns, RTC Enterprise Extensions and IBM Dependency Based Build.

2 comments on"DevOps transformation on IBM Z with Git a IBM Dependency Based Build : a case study"

Jerry Edgington • April 02, 2019

Francois, this piece, “Write a sample build script using DBB”, is now in Github, called zJenkins. It series of Groovy scripts using DBB to build many types of source code programs

Reply

Daniel Raisch • April 02, 2019

How do we position IDz and Wazi ?

Reply

Join The Discussion

Your email address will not be published. Required fields are marked *

Enter your comments...

Name *

Email *

Website