SEPTEMBER 2018

ca technologies

CA Brightside: Introduction and Common Use Cases

Building a Bridge to True Cross-Enterprise Development

Imagine if you could manage mainframe processes with the same agility as the typical cloud environment, delivering a seamless, simple and secure user experience. Turning that dream into reality would fundamentally redefine how development and operations professionals interact with and work on the mainframe.

From a development perspective the challenge is clear. Bringing agility into mainframe is critical to the future of work. Businesses actually do not have a choice. The majority of teams say are dealing with a skills gap and team productivity. The current generation of mainframers are retiring. And the upcoming generation has very limited z/OS expertise and skills.

Beyond this skills and productivity gap, there is a time to market issue as well. Simply said – today the development process for new mainframe capabilities is too slow. It's too manual. It requires special tools. And it does not integrate into CI/CD pipeline and how the rest of the organization is developing. So, the problem to be solved: The upcoming generation won't necessarily want to do things exactly the same way; but they need the right tools to work.

Introducing CA Brightside

CA Brightside is the first cross-enterprise DevOps solution designed to greatly improve the productivity and efficiency of development teams to control, script and develop for the mainframe like any other cloud platform.

This new developer-friendly solution empowers next-generation developers to more easily apply their experience with modern DevOps toolchains and frameworks, helping to increase their ability to innovate for the mainframe platform.

CA Brightside's automation with self-service capabilities unlock mainframe value to reduce risk and cost by helping to accelerate software development with higher quality and create a fine-tuned cross-enterprise DevOps environment.

CA Brightside opens IBM Z to a new generation of developers:

- Interact with data sets and files: Create and edit mainframe data sets and USS files using the CLI.
- Submit jobs: Submit JCL from data sets or local files, monitor the status, and view and download the output automatically.
- Issue console commands: Submit TSO and console commands to the mainframe directly from Brightside CLI.
- Provision test environments: Exploit z/OSMF to provision on-demand test environments for some services such as CICS.
- Produce responses as JSON documents: Return data in JSON format on request.
- Embed CLI commands in their favorite scripting language: Build local scripts that automate repetitive tasks and enable them to perform scriptable tasks like source builds, continuous integration and delivery.

The Bottom Line:

CA Brightside is designed to help your team:

- Become more effective earlier in the development cycle in hours vs. days
- Develop, test and release code faster with higher quality
- Free up your time through automated, self-service capabilities
- Control, script & develop for the Mainframe like any other cloud platform while reducing risk & cost
- Use familiar, industry-standard, opensource tools so that you could accelerate Mainframe integration into your enterprise DevOps initiatives

Zowe from the Open Mainframe Project (OMP)

Zowe is a new open source framework that bridges the divide between modern applications and the mainframe. It's the first-ever open source project based on z/OS.

CA Technologies is excited to participate with the OMP and Zowe community members to streamline the development process for applications leveraging the mainframe platform. We are committed to the Zowe initiative as it provides simplified and familiar infrastructure services for the mainframe benefiting both experienced and newer developers and will help our customers accelerate the time-to-market as they deploy their mission critical digital transformation strategies.

To learn more about Zowe and OMP, visit: https://www.openmainframeproject.org/projects/zowe

Common Use Cases for CA Brightside

Note: All use cases either originated from customers or were verified by customers to be valid.

Use Case #1: Building Visual Orchestration in a CI/CD pipeline for a mainframe application

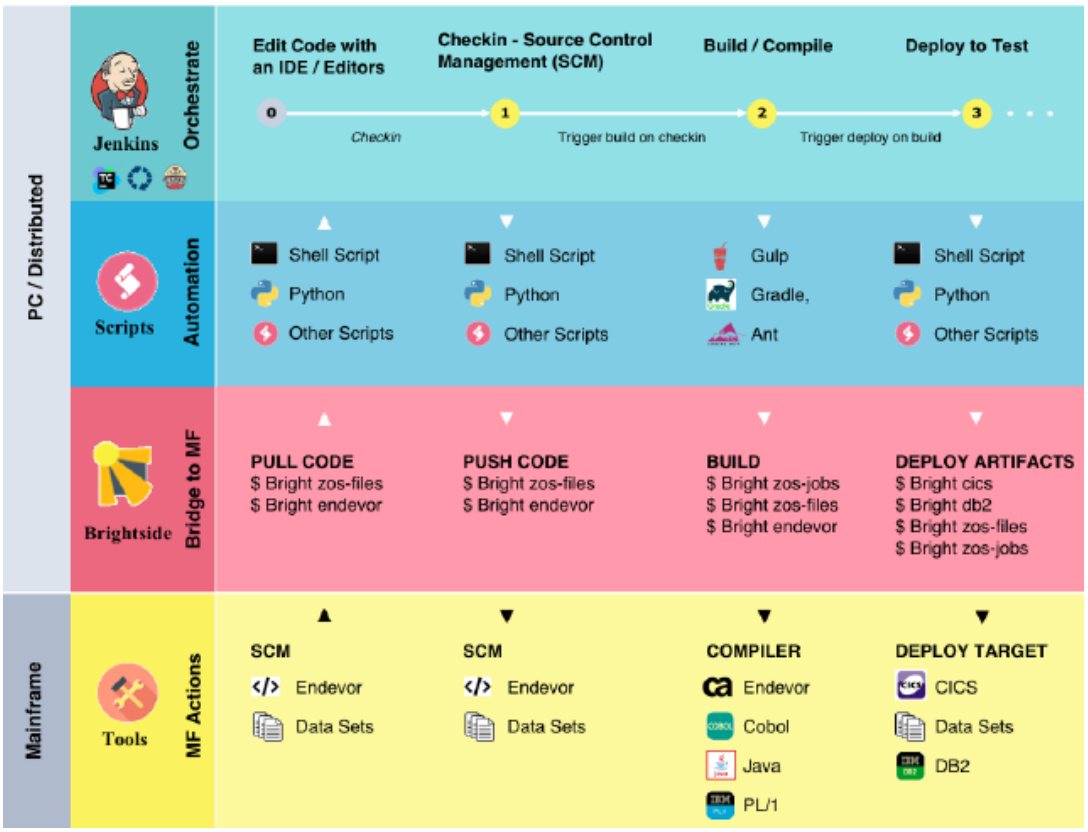
Persona: Ravi (DevOps Engineer)

One of the applications that Ravi supports wants to configure an automated pipeline to perform a build of the application, and deploy it to a test environment.

Solution: Ravi chooses Jenkins as the CI/CD orchestrator to automate the pipeline. Ravi is proficient with shell scripting, so he uses Brightside CLI commands in a few simple shell scripts to automate the build on Endevor and to deploy the Endevor build artifacts to a live test CICS environment. He integrates the shell scripts easily into Jenkins with a click of a button and has a working pipeline.

Benefits:

- The application team is happy that when they make changes to the code, Ravi's pipeline automatically performs a build and deploy.
- Ravi used his tool of choice for CI/CD orchestration and is happy that he has this flexibility and freedom.
- Ravi uses his favorite scripting language to automate mainframe actions, and is happy that he does not need to learn Mainframe specific languages.
- Ravi automates the pipeline in the same way as he would with any other cloud platform. He's happy that his skills are current and easily transferrable to other platforms.



Use Case #2: Static Analysis of COBOL/PL1

Persona: Michelle (New Mainframe Application Developer), Ravi (DevOps Engineer)

Michelle has been working on a code change for her CICS transaction programs. She wants to run the code through static analysis to ensure that she's following proper coding conventions. Her code is on the mainframe and she wants to make it have an easy way of passing it static analysis tools and get the results back.

Solution:

Michelle chooses to write a script that use Brightside commands to pass code from MF to static analysis tool. Going forward she just runs this script every time to do static analysis on her MF code.

Benefits:

- Michelle is happy that she can gain confidence in her code changes following language conventions
- Ravi is happy that he was able to include static analysis in his pipeline for the code change

Use Case #3: System and Integration testing for mainframe apps

Persona: Michelle (New Mainframe Application Developer)

Michelle has been working on a mainframe app called Acme. Acme has lately been encountering errors in the production environment. Michelle generally performs rigorous unit testing of modules she changes before she pushes her changes. With the recent errors she believes this is not enough and wants to incorporate system and integration testing as part of her development process.

Solution:

Michelle chooses to implement system and integration tests using a javascript based modern testing framework called MochaJS by using Brightside's APIs and CLIs.

Benefits:

- Michelle is happy that she can gain confidence in her code changes at a system level before pushing her code to the next level of the pipeline.
- Michelle's manager and architect are happy that she is able to use an open source testing framework that is used and maintained by 10 million + developers across the world. This means Michelle's company doesn't have to spend valuable $ maintaining these testing frameworks on their own.

Use Case #4: Automating the maintenance and release upgrade of z/OS vendor software

Persona: Sysprogs

The process of patching and upgrading vendor software can be repetetive and error prone. Automating this using using orchestration tools like Jenkins would streamline and improve the vendor software upgrade process. In addition, this vendor software upgrade pipeline could be integrated with the pipelines of customer applications that rely on vendoro software changes.

Solution:

Brightside's CLI can be used to automate vendor software maintenance and upgrades from Jenkins.

Benefits:

- Fast maintenance/upgrade time
- Use familair scripts
- Potential for consistent process
- Ability to trigger other pipelines

Use Case #5: A Non-GUI modern interface to mainframe services

Persona: Sysprogs, Operators, Developers, Admins

Users are tired of having to navigate ISPF panels, browser GUIs or IDE GUIs to interact with the mainframe. They want to easily issue commands interactively on a local terminal and either query or take action on mainframe services.

Solution:

Users issue commands from their terminal of choice to interact with all mainframe services. This complements the GUIs that may already available and gives power users another interface into the mainframe.

Benefits:

- Do not have to take your hands off of the keyboard
- Can use local commands like sed/grep or other bash commands on data from the mainframe
- Makes mainframe more like a cloud platform

Use Case #6: Mission-critical web application is not meeting SLA requirements

Persona: Mindy (Web / Cloud Developer)

A web application's server code has been stressed to its maximum and has started missing SLAs lately. Mindy works on this application and has determined that the slowdown can be alleviated if the web server is moved closer to the data system of record. The system of record in this case is the mainframe with web services in CICS and DB2. Mindy wants to experiment with running the web application on the mainframe to increase the app's change of making SLA commitments.

Solution:

Mindy issues a few Brightside CLI commands from her terminal to deploy the web application to the mainframe and she is now ready to experiment with it.

Benefits:

- Mindy had heard that the mainframe can be difficult to interface with but she was pleasantly surprised that she can deploy to the mainframe platform in the same way as she would to AWS or other cloud platforms.
- CTO is happy that the mainframe infrastructure they pay a lot of $ is being considered for more workload •If the experiment succeeds, then the Services leadership is happy that this mission-critical application is now able to conform to its SLA commitments.

Use Case #7: Automating the build of COBOL/PL1 on the Mainframe using modern build tools like Gradle

Persona: Developers

Developers make code changes and want to perform a build to either check for errors or to test the changes. Code and build logic may reside in an SCM like Endevor or in PDS members.

Solution:

Brightside's CLI can be used to automate the build of mainframe code using modern build tools like Gradle or Gulp.

Benefits:

- Reusable
- Easy integration with CI/CD tools
- Easily understandabeable build logic
- Less need for MF specific domain knowledge

Use Case #8: Use Brightside to build plugins and other applications

Persona: Sysprogs, Operators, Developers, Admins

People build small standalone tools and plugins for IDEs/editors on their own to make their lives easier. This is not easy when it comes to building your own custom tools for mainframe.

Solution:

Brightside's CLI as well as the Node.js APIs can be used to quickly build plugins for IDEs/editors or even stand-along GUIs.

Benefits:

- Fast development time
- Brightside handles credential management
- Brightside handles security
- Brightside handles connection details to mainframe services

Use Case #9: Populate a test environment with data

Persona: Developers, DBAs, DevOps Engineer

When starting work on a project or just experimenting, peopel often want to create either sythetic data or use masked real data in their test environment. Small experiments may just need some simple synthetic data created and populated, while large projects may require complex data from a tool like TDM.

Solution:

- VSAM:Brightside provides a means to create new table data as well as use data from an existing VSAM to populate a new VSAM (requires file master plus plugin)
- DB2: Brightside provides commands to run SQL commands. Brightside also allows exporting data from an existing database and inserting it into a new database.
- IMS: N/A
- IDMS: N/A
- Datacom: N/A

Benefits:

- Enables quick testing of corner cases
- Promotes independence with creating test data
- Allows test data generation and population to be part of automated CI/CD

Use Case #10: Automate the verification of an IPL

Persona: Sysprogs

After an IPL, many variables need to be checked to ensure that an IPL was successful. Automating the verification of these variables after an IPL and producing a report is useful.

Solution:

Use Brightside's zos-console, zos-tso and zos-jobs groups to automate the verification from modern scripting languages like Python or testing frameworks like Jest. Execute these task orchestrators like Jenkins to provide a visual of the tasks, produce and store reports.

Benefits:

- Parsing and autoamtion can be written modern scripts
- Visualization and reports of the verification is possible
- Reports can be stored for future use

Use Case #11: Write Performance Tests for Mainframe Applications

Persona: AppDev, Performance Tester

When code changes are made to my CICS application, I want to ensure that the performance of the transactions have not degraded.

Solution:

Use existing performance testing frameworks like Blazemeter, and use Brightside to query services like CICS and supporting services like SYSVIEW on the mainframe to obtain performance indicators.

Benefits:

- Use existing performance testing frameworks
- No 3270 screen scraping is needed

Use Case #12: Copy Datasets from one LPAR to another LPAR without shared DASD

Persona: Any MF user

Often one might have to copy some datasets from one LPAR to another without having any shared DASD. It may be for the purpose of testing or to roll out new maintenance to a new LPAR.

Solution:

Use Brightside to copy datasets from the source datasets to your PC or a server and copy it from there to the target LPAR.

Benefits:

- Brightside commands use HTTPS which is secure
- Able to move datasets easily without having to start an FTP session

Use Case #13: Automating the submission of a dynamically modified batch job

Persona: Batch Operators

In special cases, a scheduled job may fail. Automate the dynamic modification of the faltering JCL for known cases and automatically resubmit the job.

Solution:

Use a framework like mochaJS to automate downloading the JCL, modifying it dynamically and re-submitting it and validating the output all using Brightside commands or nodeJS APIs.

Benefits:

- Ability to react to faltering batch jobs

By delivering a cloud-like experience for the mainframe, application delivery teams can make significant strides forward. Managers no longer have to recruit for mainframers with decades of experience, but can hire the next generation of developers. Barriers to innovation come down as developers and DevOps architects can respond with more agility to customer and business demands. Costs and cycle times decrease through self- service provisioning for mainframe assets. Developer and architect satisfaction and productivity skyrocket.

Learn more. Visit us online today at **ca.com/brightside**

CS000-SPK_0918