Jenkins

The coded business

open source

wipro

When a large banking services provider was faced with a DevOps muddle, streamlining the processes and decentralization of tasks helped them simplify their DevOps platform. This resulted in considerable reduction (around 20%) of platform maintenance overheads.

In times of agile development, DevOps has a vital role to play. While there is no single tool which can address all business and development issues, there are potential approaches that can streamline the automation of these tasks. As enterprises grow, mundane tasks can become maintenance nightmares. Hence, there is a pertinent need to optimize existing DevOps platforms to sustain the organization's business velocity.

The Jenkins approach will bring agility into pipeline management by allowing businesses to create their own Jenkinsfile for the pipeline plugin. This will help them manipulate it exactly the way they want to.

Jenkins: An overview

Jenkins is a widely used open source continuous build and integration tool. It has over a thousand plugins for a variety of supporting tools and technologies and a strong community. In Jenkins, continuous build/integration/testing or deployment-related tasks are designed and developed into a set of Jenkins jobs. A group of jobs associated with a specific delivery lifecycle of a project is called a Pipeline. Pipelines manage the continuous delivery lifecycle of a project.

Pipelines evolve during the project lifecycle and have to be managed by operations teams in collaboration with development teams. It is a continuous process which quickly becomes complex as the number of dependencies for a project increase and as the code base grows considerably over a period of time.

Jenkins has introduced 'Jenkinsfile' which can be treated as a 'make file' for pipeline creation and execution, similar to 'pom' file for a Maven project. All jobs that are required to be executed for a given project can be coded into this Jenkinsfile as Groovy statements. Moreover, this file is versioned along with the project creating the pipeline-as-a-code concept.

Pipeline as code

The traditional way of implementing a pipeline requires modification or creation of jobs using the Jenkins console or to modify the Jenkins job template through various custom methods. While this method is simpler to use, it comes with certain long-term limitations:

- Not scalable as the number of projects increase
- Concerned Ops team needs to define and manage these templates
- No intuitive direct association or correlation of job definitions with the respective project
- Versioning of these job definitions cannot be in sync with the respective project versions
- Job definition ownership resides with multiple teams (Dev and Ops)
- Complexity in pipelines results in manageability of overheads

Success factors

In the new approach, the pipeline steps are implemented as groovy instructions without the need for explicitly defining the same using the Jenkins console. This approach provides some interesting advantages:

- Ability to define complex workflow tasks with human interaction capabilities using simple groovy instructions
- Complete job creation ownership resides with the developers of the project
- Pipeline lives with the project and hence it is version-able along with the project
- Better control of pipeline execution at runtime with fine-grained access to each task through groovy instructions
- Operations team needn't manage these pipelines anymore making the pipeline autonomous
- Inheritance of pipelines/groovy instructions from external sources such as files/projects

However, we have to keep in mind that the new approach is supported only by Jenkins version 1.65 and above and is mandatory to have the Jenkinsfile as part of every project which goes through continuous build and delivery. Jenkinsfile also encapsulates entire pipeline details in the form of Groovy instructions. Furthermore, the naming convention of the Jenkinsfile cannot be changed, and pipeline plugins are mandatory to leverage these features.

With the focus on microservice based architecture, the Jenkinsfile approach gives more control and flexibility to design a pipeline. It also provides fine-grained control on the usage of Jenkins slaves against a specific section of pipeline.

The Jenkinsfile approach

On the project side, availability of a version control system such as Github or SVN is needed. Respective project webhooks are configured to trigger Jenkins jobs. Git user credentials for the corresponding project are configured in Jenkins server by creating a file named Jenkinsfile in the root directory of the project (alongside pom file in java project or app.js file in a node.js project).

The pre-requisites for the Jenkins server are the availability of pipeline, multibranch, organization and workflow plugins pre-installed. As part of the build process, once a commit triggers a webhook (e.g., in GitHub), Jenkins job will be triggered. Jenkins will look for the Jenkinsfile in the root folder of the project. Apart from having Jenkinsfile, there is a need to have one of the following Jenkins jobs:

- **Multibranch pipeline:** To build multiple branches of a single repository automatically
- **Organization folders:** To scan and discover available repositories(e.g.Github) associated with an enterprise and automatically creating managed Multibranch Pipeline jobs for each of the projects

The enhanced approach

Motivation

A project that needs to undergo continuous build and integration processes using the Jenkinsfile approach should have the multi-branch pipeline job pre-created in the Jenkins server. This step adds a dependency to the build process and then triggers the build pipeline through the Jenkinsfile.

Approach

In order to eliminate the dependency of creating a multi-branch step through Jenkins UI, this job is created at runtime through Jenkins API. While creating the main Jenkins job the subsequent build task is passed as a parameter to the main Jenkins job. Any parameters that are required for the subsequent Jenkins jobs are passed as parameters to the main Jenkins job. Within the Jenkinsfile these job parameters can be accessed as "${ParamName}" where ParamName is the parameter name passed to the main Jenkins job.

Conclusion

It is evident that organizations wanting to leverage on the approach can benefit immensely. The Jenkins approach is likely to bring in agility to the pipeline management by allowing businesses to create their own Jenkinsfile for the pipeline plugin and manipulate it exactly the way they want it. This will give more control and options than in the UI alone. Thus, flexibility and isolation that it offers to the project pipelines will help in managing continuous deployments across platforms, and create a structured way of managing DevOps tasks in a complex environment. Wipro's OpenApp platform leverages the Jenkinsfile approach to enable pipeline as a code capability to the stacks and services deployed on the platform.

About the author

Sreekanth Nyamars
Lead Architect - Open Source COE,
Service Transformation

Sreekanth is an Open Source Lead Architect at Wipro. Sreekanth's focus and interests are building solutions through the use of open source technologies, particularly DevOps and PaaS platforms. His experience includes the implementation of integration platform solutions for both large payment service providers and telecom service providers. He is a certified Java Enterprise Architect.

Wipro Limited
Doddakannelli, Sarjapur Road,
Bangalore-560 035,
India

Tel: +91 (80) 2844 0011
Fax: +91 (80) 2844 0256
wipro.com

Wipro Limited (NYSE: WIT, BSE: 507685, NSE: WIPRO) is a leading global information technology, consulting and business process services company. We harness the power of cognitive computing, hyper-automation, robotics, cloud, analytics and emerging technologies to help our clients adapt to the digital world and make them successful. A company recognized globally for its comprehensive portfolio of services, strong commitment to sustainability and good corporate citizenship, we have a dedicated workforce of over 170,000, serving clients across six continents. Together, we discover ideas and connect the dots to build a better and a bold new future.

For more information,
please write to us at
info@wipro.com