N-ABLE (PVT) LTD
36, Bristol Street, Colombo 01, Sri Lanka

Mobitel – CICD Pipeline

As-Build Document

Confidential

Version: 1.0
Issue Date: 07/07/2022
Document Type: SIGNOFF

Contents

Acronym/ Terminology	Definition
ACE	IBM App Connect Enterprise

Version		Author	Notes	Reviewer
0.1	07/07/2022	Iresh Madhusanka	Initial Release	Samitha Udalagama

1. Introduction

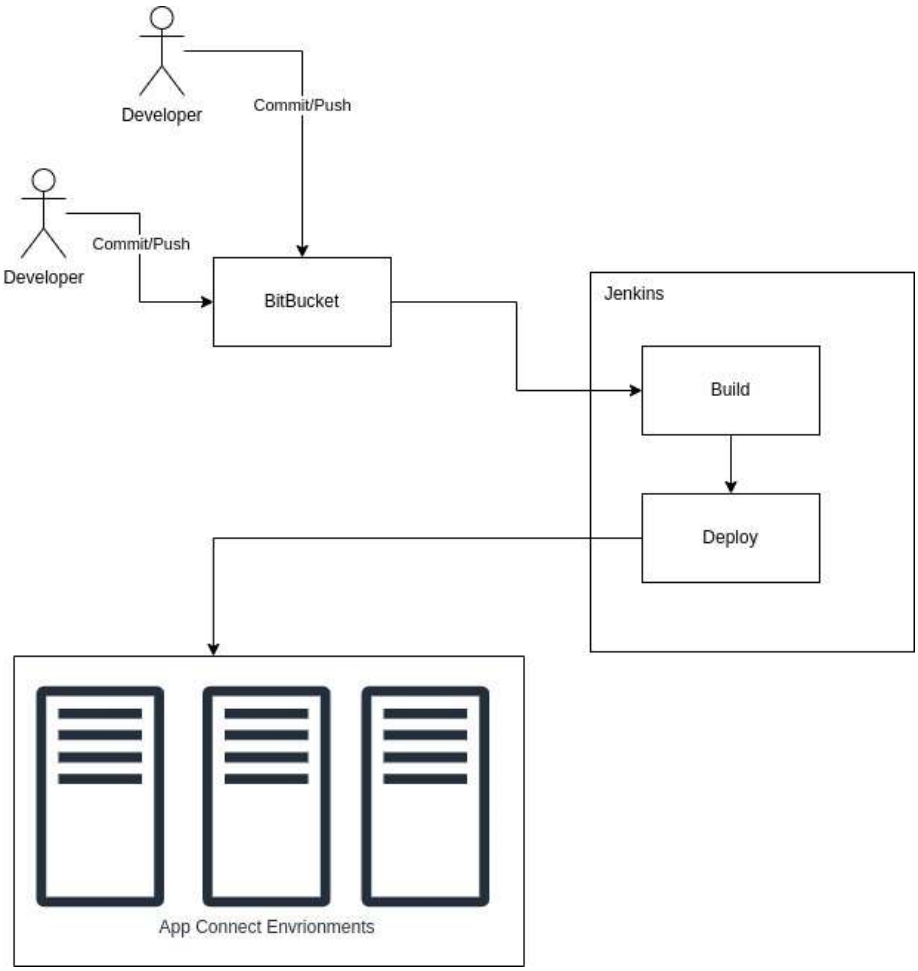
The adoption of DevOps methodologies at Mobitel, the continuous integration and continuous delivery pipeline streamlines and reduces the friction in the process of taking an IBM App Connect application to production from the stage of development. The pipeline automates the build and deployment of applications reducing the time taken for each development to deployment cycle.

2. Scope of document

The scope of this document is limited to providing a high-level solutions architecture of implemented CICD pipeline.

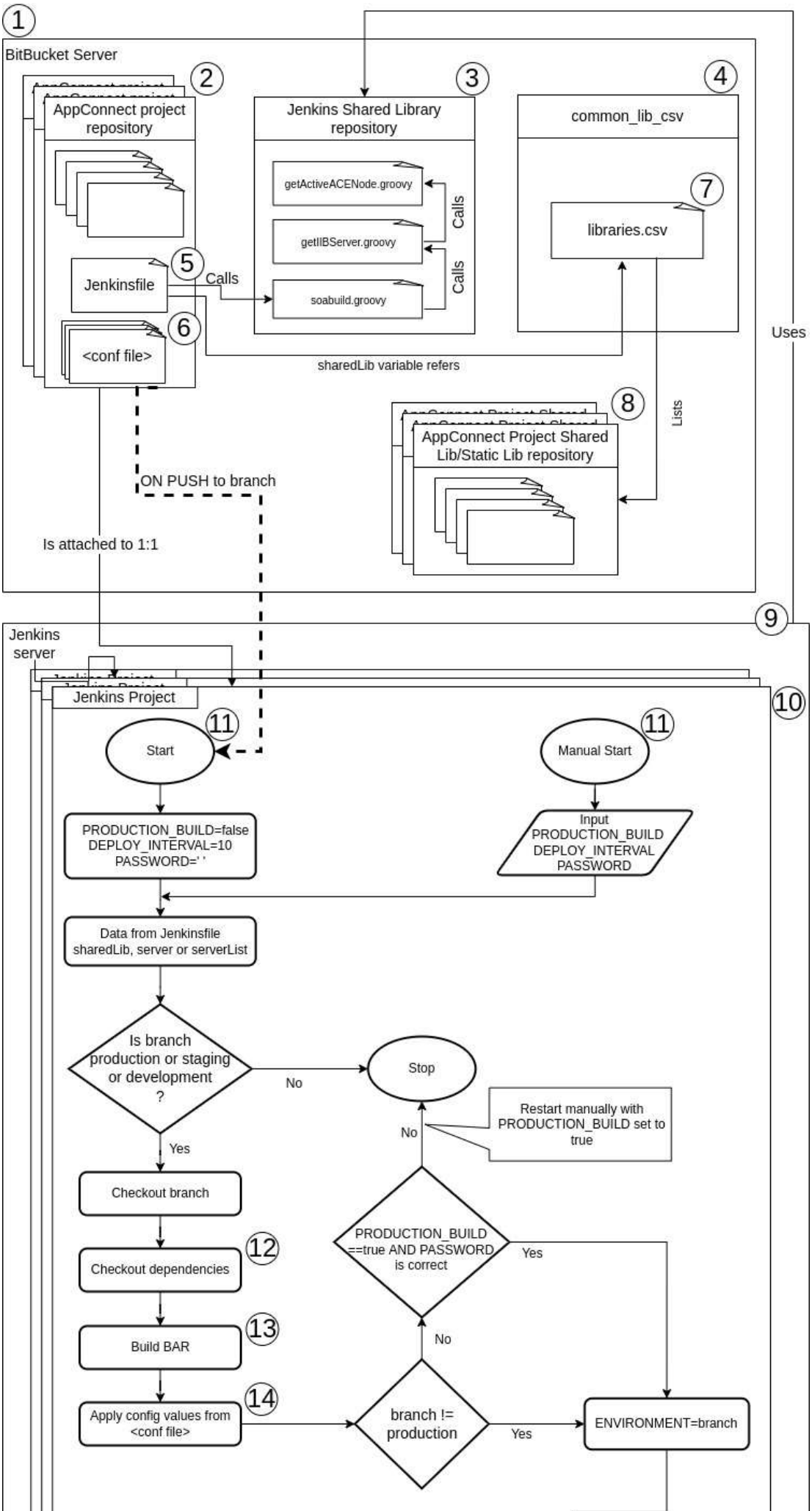
3. Solution overview

The following diagram illustrates a high-level overview of the implemented pipeline.



4. Solution architecture and implementation

The following diagram illustrates the CICD pipeline architecture and workflow.


1
BitBucket Server
2
AppConnect project repository
3
Jenkins Shared Library repository
4
common_lib_csv
7
libraries.csv
getActiveACENode.groovy
getIIBServer.groovy
soabuild.groovy
Calls
Calls
5
Jenkinsfile
Calls
6
<conf file>
sharedLib variable refers
Uses
Lists
8
AppConnect Project Shared Lib/Static Lib repository
ON PUSH to branch
Is attached to 1:1
9
Jenkins server
Jenkins Project
10
11
Start
11
Manual Start
PRODUCTION_BUILD=false
DEPLOY_INTERVAL=10
PASSWORD=' '
Input
PRODUCTION_BUILD
DEPLOY_INTERVAL
PASSWORD
Data from Jenkinsfile
sharedLib, server or serverList
Is branch production or staging or development ?
No
Stop
Restart manually with PRODUCTION_BUILD set to true
No
Yes
Checkout branch
PRODUCTION_BUILD ==true AND PASSWORD is correct
Yes
12
Checkout dependencies
No
13
Build BAR
14
Apply config values from <conf file>
branch != production
Yes
ENVIRONMENT=branch

4.1. Components and concepts

4.1.1. IBM App Connect Enterprise applications

IBM App Connect Enterprise applications run on integration servers. Mobitel's application deployment architecture dictates that a set of applications is associated with a given integration server. Therefore 6 sets of applications run on 6 different integration servers listed in the section IBM App Connect Environments. As a result, each application is designated to run on a specific integration server.

4.1.2. IBM App Connect Enterprise libraries

IBM App Connect Enterprise libraries are shared across multiple IBM App Connect Enterprise applications as dependencies. They contain common modules used by the applications. The source code of a shared library is kept in a bitbucket repository ⑧ (One repository for each library).

4.1.3. Common_lib_csv ④

This is a repository that contains the file libraries.csv ⑦ which is an index of all available IBM App Connect libraries in the organization. This file is referred to by the pipeline workflow's 'Checkout dependencies' ⑫ step to build the application with the dependent libraries.

libraries.csv is of the following format,
<library_name>,<git_clone_url>,<path_to_lib_from_repo_root>

Example:

```
RESTRequest_SharedLibrary,https://github.com/IreshMMOut/RESTRequest_SharedLib.git,/
RESTRequest_SharedLibrary_v1.1.0,https://github.com/IreshMMOut/RESTRequest_SharedLib_v1.1.0.git,/
```

4.1.4. App Connect Project repository ②

The project repository contains the source code of an IBM App Connect Enterprise application. Each IBM App Connect Enterprise application has one to one relationship with an App Connect Project repository.

The project repository contains a Jenkinsfile ⑤ and 3 conf files ⑥ (1 for each development, staging, production branch/environment) except for the application source files.

4.1.4.1. Jenkinsfile ⑤

```
soabuild {
    sharedLib = 'https://bitbucket.mobitel.lk/projects/IIB/repos
                 /common_lib_csv/raw/libraries.csv?at=refs%2F
                 heads%2Fmaster'
    server = 'IS_SYS'
}
```

'sharedLib' parameter refers to libraries.csv file described in section Common_lib_csv.

'server' parameter refers to the integration server designated for the application whose source code is hosted in the project repository.

4.1.4.2. 3 Conf files ⑥

The names of the conf files are based on the application name and branch.

Example:

myapplication.conf.development
myapplication.conf.staging
myapplication.conf.production

A conf file contains the properties specific to the application through which the application could be configured.

```
sampleFlow#MQ Input.queueName=NEW_INPUT_QUEUE
sampleFlow#sampleSubflow1.queueName
sampleSubflow1#queueName
SUBOUT=NEW_SUBOUT
```

4.1.5. Jenkins server ⑨

Jenkins server is the automation server where the CiCD pipeline is executed.

4.1.6. Jenkins project ⑩

A Jenkins project represents the pipeline for a specific App Connect project repository. An App Connect project repository has one to one relationship with a Jenkins project.

4.1.7. Jenkins shared library repository ③

The Jenkins shared library repository contains the source files of the definition of the Jenkins pipeline. The Jenkins project uses this repository to pull down the pipeline definition.

4.1.8. IBM App Connect Environments

Each integration node in every environment contains the following set of integration servers,

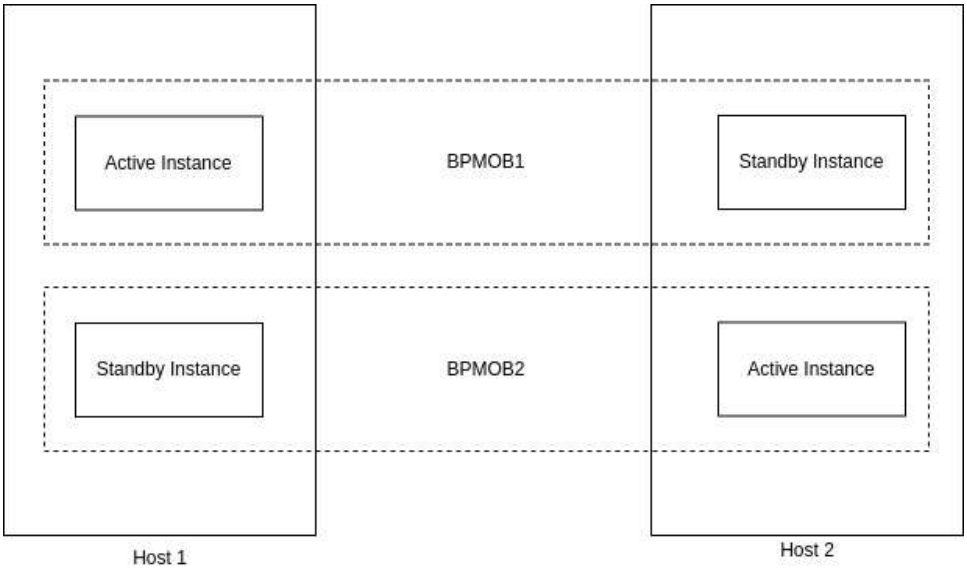
- IS_ERP
- IS_CRM
- IS_EAP
- IS_SYS
- IS_OCS
- IS_COMMON

The following environments consist of integration nodes following the described architecture for each environment.

4.1.8.1. Production environment

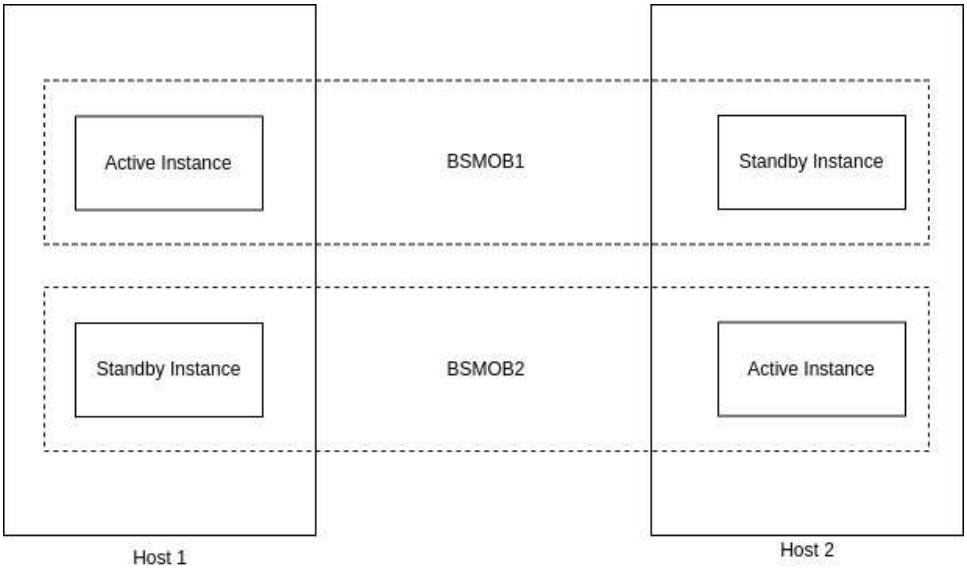
The production environment has two integration nodes each running in an

active/standby configuration running on two computers.



4.1.8.2. Staging environment

The staging environment is identical to the production environment except for the integration node names.



4.1.8.3. Development environment

The development environment consists of only a single integration node running on a single host

4.2. The pipeline

The pipeline defines workflow that automates the process of building and deploying an application to an environment. Tasks done by the pipeline are discussed below,

4.2.1. Beginning of the pipeline ⑪

The pipeline can be started either manually or is started automatically when a change is detected to either of the branches below,

- production
- development
- staging

4.2.2. Dependency resolution ⑫

The resolution of dependencies is done as discussed in section 4.1.3 and 4.1.4.1

4.2.3. Build ⑬

The application artifact is built using the mqsicreatebar utility

4.2.4. Application configuration ⑭

The configuration values defined in the relevant conf file for the branch are applied to the built application artifact file

4.2.5. Application deployment ⑮

The built application artifact is deployed to the relevant environment as follows,

If the relevant environment is not a production environment artifact is automatically deployed to the relevant environment.

If the relevant environment is the production environment, deployment is not done except when the pipeline is manually started, and production deployment is indicated and the deployment password is provided.

For any given environment, an application deployed to its designated integration server as described in section 4.1.1

4.2.6. Preserving the built artifact

The artifact built is preserved in the Jenkins server online for the production builds and can be retrieved later.