### What are DORA Metrics?

**DevOps Research and Assessment (DORA)** team has identified four key metrics that indicate the performance of a software development team.

### Core Components (KPIs):

| Metric | What It Measures | High Performance Indicator |
| --- | --- | --- |
| **Deployment Frequency** | How often successful releases are made to production | Multiple deployments per day |
| **Lead Time for Changes** | Time taken from code commit to deployment in production | Short lead times indicating fast delivery |
| **Change Failure Rate** | Percentage of deployments causing production failures | Low failure rate indicating stability |
| **Mean Time to Recovery (MTTR)** | Time to restore service after a failure | Fast recovery times showing good incident response |

## TeamUp DORA Dashboard at AT&T

* **Access the dashboard:** <https://teamup.web.att.com/dora-metrics>
* Filter by timeframe and metrics, e.g., deployment frequency:  
  <https://teamup.web.att.com/dora-metrics?d=90_d&p=40&t=deployment-frequency>
* **Search by MOTS\_ID** to view specific app metrics. Example: IDP - Platform - 30655.
* Dashboard reports production deployment details such as deployment date, build ID, artifacts, repositories — ideal for tracking microservices deployment.

### Integration Steps in Jenkins

#### **1. Prerequisites**

* **Access to the shared library**: Request access to the cicd\_functions library via a ticket.
* **Configure Jenkins**:
  + Add the shared library under **Pipeline Libraries** in Jenkins.
  + Use credentials with read access to the library.

#### **2. Import the Library**

At the top of your [Jenkinsfile](command:_github.copilot.openRelativePath?%5B%22Jenkinsfile%22%5D), include:

@Library('cicd\_functions@main') \_

#### **3. Define Artifact Details**

Artifacts represent metadata about your build or deployment. Example:

artifact = [  
 uniqueId: "<project>-<branch>-BUILD\_NUMBER",  
 name: "<project>-<branch>",  
 version: <artifact\_version>,  
 gitCommitHash: <commit\_hash>,  
 gitRepositoryURL: <repo\_url>  
]

#### **4. Send Metrics**

Use the cicd\_notifications.sendDoraMetrics function to send metrics for builds or deployments.

* **Build Example**:

cicd\_notifications.sendDoraMetrics(  
 buildOrDeploy: "build",  
 teamUpEnv: "production",  
 motsId: "<MOTSID>",  
 project: "<Project-Name>",  
 pipelineName: "<Pipeline-Name>",  
 buildId: BUILD\_NUMBER,  
 buildUrl: BUILD\_URL,  
 artifact1: artifact  
)

* **Deploy Example**:

cicd\_notifications.sendDoraMetrics(  
 buildOrDeploy: "deploy",  
 teamUpEnv: "production",  
 isProduction: true,  
 targetEnvironment: "<target-env>",  
 motsId: "<MOTSID>",  
 project: "<Project-Name>",  
 pipelineName: "<Pipeline-Name>",  
 buildId: BUILD\_NUMBER,  
 buildUrl: BUILD\_URL,  
 artifact1: artifact  
)

### Pipeline Example

A complete pipeline might look like this:

pipeline {  
 agent any  
 stages {  
 stage('Build') {  
 steps {  
 echo "Building the application..."  
 }  
 post {  
 always {  
 script {  
 def artifact = [  
 uniqueId: "${project}-${env.BUILD\_BRANCH}-${BUILD\_NUMBER}",  
 name: "${project}-${BUILD\_BRANCH}",  
 version: "${env.BUILD\_VERSION}",  
 gitCommitHash: "${gitCommitHash}",  
 gitRepositoryURL: "${gitRepositoryURL}"  
 ]  
 cicd\_notifications.sendDoraMetrics(  
 buildOrDeploy: "build",  
 teamUpEnv: "production",  
 motsId: "<MOTSID>",  
 project: "${project}",  
 pipelineName: "${env.JOB\_NAME}",  
 buildId: "${env.BUILD\_NUMBER}",  
 buildUrl: "${env.BUILD\_URL}",  
 artifact1: artifact  
 )  
 }  
 }  
 }  
 }  
 stage('Deploy') {  
 steps {  
 echo "Deploying the application..."  
 }  
 post {  
 always {  
 script {  
 def artifact = [  
 uniqueId: "${project}-${env.BUILD\_BRANCH}-${BUILD\_NUMBER}",  
 name: "${project}-${BUILD\_BRANCH}",  
 version: "${env.BUILD\_VERSION}",  
 gitCommitHash: "${gitCommitHash}",  
 gitRepositoryURL: "${gitRepositoryURL}"  
 ]  
 cicd\_notifications.sendDoraMetrics(  
 buildOrDeploy: "deploy",  
 teamUpEnv: "production",  
 isProduction: true,  
 targetEnvironment: "${env.TARGET\_ENV}",  
 motsId: "<MOTSID>",  
 project: "${project}",  
 pipelineName: "${env.JOB\_NAME}",  
 buildId: "${env.BUILD\_NUMBER}",  
 buildUrl: "${env.BUILD\_URL}",  
 artifact1: artifact  
 )  
 }  
 }  
 }  
 }  
 }  
}

# DORA Metrics Shared Library

The dora-metrics shared library is designed to integrate DORA metrics into Jenkins pipelines. It provides reusable methods to send build and deployment metrics to the DORA TeamUp system, enabling better tracking of software delivery performance.

Github: [ATT-DP2/apm0009754-bbnmslstls-jenkins-libraries](https://github.com/ATT-DP2/apm0009754-bbnmslstls-jenkins-libraries)

## Features

* **Build Metrics**: Automatically sends build-related metrics, including project name, pipeline name, build ID, and artifact details.
* **Deployment Metrics**: Sends deployment-related metrics, including target environment and production status.
* **Reusable Helper Methods**: Provides utility methods for retrieving GitHub team information, repository details, and creating artifacts.

## Methods

### 1. sendDoraBuildMetrics

**File**: vars/commonMethods.groovy

This method sends build-related metrics to the DORA TeamUp system.

#### Parameters:

* **env**: Jenkins environment object (e.g., JOB\_NAME, BUILD\_NUMBER, BUILD\_URL).
* **cicd\_notifications**: Utility object for sending notifications and metrics.

#### Example Usage:

post {  
 always {  
 script {  
 def cm = new commonMethods()  
 cm.sendDoraBuildMetrics(env, cicd\_notifications)  
 }  
 }  
}

### 2. sendDoraDeployMetrics

**File**: vars/commonMethods.groovy

This method sends deploy-related metrics to the DORA TeamUp system.

#### Parameters:

* **env**: Jenkins environment object (e.g., JOB\_NAME, BUILD\_NUMBER, BUILD\_URL).
* **cicd\_notifications**: Utility object for sending notifications and metrics.
* **k8s\_namespace**: Kubernetes namespace for the deployment (used to determine if it’s a production environment com-att-bbnmsls-ol-prod).

#### Example Usage:

post {  
 always {  
 script {  
 def cm = new commonMethods()  
 cm.sendDoraDeployMetrics(env, cicd\_notifications, params.k8s\_namespace)  
 }  
 }  
}

### 3. Helper Methods

**File**: src/org/dora/metrics/Helper.groovy

#### getRelease(env)

* **Description**: Extracts the release name from the Jenkins job name or Git branch.
* **Returns**: A string representing the release name.

#### getServiceName(env, team, repoUrl)

* **Description**: Extracts the service name from the repository name based on the team.
* **Returns**: A string representing the service name.

#### getProjectName(env, team, repoUrl)

* **Description**: Constructs the project name based on the team and service name.
* **Returns**: A string representing the project name.

#### createArtifact(env, repoUrl, revision)

* **Description**: Creates an artifact object containing metadata such as unique ID, name, version, Git commit hash, and repository URL.
* **Returns**: A map representing the artifact.

#### getGitHubTeam(repoUrl)

* **Description**: Extracts the GitHub team name from the repository URL.
* **Returns**: A string representing the team name.

## Integration with Jenkins Pipelines

### Adding the Library

To use the dora-metrics library in your Jenkins pipeline, include it using the @Library annotation:

@Library('cicd\_functions','dora-metrics') \_

### Example Pipeline

Here’s an example of how to use the dora-metrics library in a Jenkins pipeline:

pipeline {  
 agent any  
 stages {  
 stage('Build') {  
 steps {  
 echo "Building the application..."  
 }  
 post {  
 always {  
 script {  
 try{  
 def cm = new commonMethods()  
 cm.sendDoraBuildMetrics(env, cicd\_notifications)  
 }catch(warn){  
 echo "[WARN] : Failed to send DORA build metrics : ${warn}"  
 warn.printStackTrace()  
 }   
 }  
 }  
 }  
 }  
 stage('Deploy') {  
 steps {  
 echo "Deploying the application..."  
 }  
 post {  
 always {  
 script {  
 try{  
 def cm = new commonMethods()  
 cm.sendDoraDeployMetrics(env, cicd\_notifications, params.k8s\_namespace)  
 }catch(warn){  
 echo "[WARN] : Failed to send DORA build metrics : ${warn}"  
 warn.printStackTrace()  
 }  
   
   
 }  
 }  
 }  
 }  
 }  
}

### Notes

* Ensure the cicd\_notifications object is properly initialized in your pipeline.
* The k8s\_namespace parameter should be passed as part of the pipeline parameters.
* Review the logs for any errors related to sending metrics.