

CASE

STUDY



Migration from Jenkins to GitHub Actions





Business Needs/Background

Introduction The migration from Jenkins to GitHub Actions marked as a pivotal moment for a SDK-Core Components within the DCES Department. This case study outlines the challenges encountered, strategies implemented, and the benefits gained through this transition.

Challenges

- 1. New to GitHub Actions:** The primary challenge was the unfamiliarity with GitHub Actions. As a new tool, it required a learning curve to understand its features, syntax, and best practices.
- 2. Matrix Strategy:** Implementing the matrix strategy in GitHub Actions was another challenge. This strategy allows running jobs in parallel with different configurations, which was essential for our diverse deployment environments.
- 3. Caller Workflow:** Designing a reusable caller workflow for multiple projects proved complex, requiring meticulous planning to ensure ease of adoption by other projects.





Migration Strategy

- 1. Assessment:** Evaluated the current infrastructure and identified components for migration.
- 2. Planning:** Developed a detailed migration plan, including timelines and resource allocation.
- 3. Azure Landing Zone Selection:** Chose an appropriate Azure landing zone to meet our scalability and security needs.
- 4. Infrastructure as Code (IaC):** Utilized scripts to automate the creation of infrastructure components.
- 5. Ingress Implementation:** Integrated Azure Application Gateway and Azure Front Door for managing ingress traffic.
- 6. Testing:** Conducted thorough testing to ensure the migrated application functioned as expected.
- 7. Deployment:** Deployed the application to the new Azure landing zone.





Strategies Employed

1. Reusable Workflows

- o We created reusable workflows that could be called by any project. This approach ensured consistency and reduced duplication of effort across multiple projects.
- o Example: The [checkout.yml](#) workflow is used to check out the repository code.
- o Example: The [deploy.yml](#) workflow handles the deployment process for Java-based applications deploying to a server.





2. Matrix Strategy

o We employed the matrix strategy to manage different deployment environments (e.g., SQE and DEV), enabling parallel execution of the same deployment job with varying configurations.

```
-----Step 5 : Conditional Deployment (Push Only)-----#
Deploy-To-ENV:
  if: github.event_name == 'push' && startsWith(github.ref, 'refs/heads/develop') # Combined condition
  needs: [ Build-and-Upload-Artifacts ]
  strategy:
    matrix:
      include:
        - targetENV: 'sqa'
          deploy-SSH-server: ${vars.SQA_DEPLOYMENT_SSH_SERVER}
        - targetENV: 'dev'
          deploy-SSH-server: ${vars.DEV_DEPLOYMENT_SSH_SERVER}
  uses: se-bb-admin/sdk-shared-gitactions/.github/workflows/deploy.yml@main
```





3. Caller Workflow:

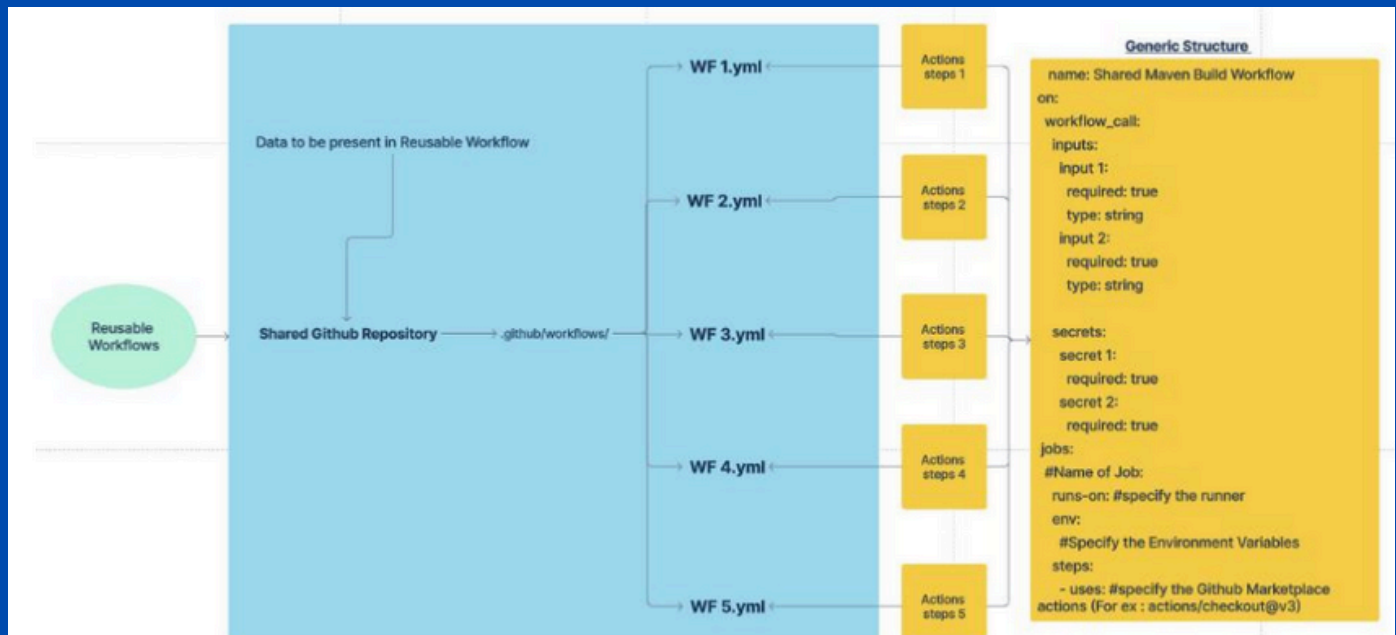
- o A caller workflow was created to trigger reusable workflows easily, facilitating the adoption of a standardized CI/CD pipeline across projects with minimal adjustments.
- o The reusable or caller workflow is particularly beneficial for projects using Java-based applications and deploying applications onto servers.

```
uses: se-bb-admin/sdk-shared-gitactions/.github/workflows/deploy.yml@main
with:
  runner: sdkrunner
  PROJECT_NAME: ums-ui
  INSTALLER_NAME: install-ums-ui
  ENV: ${{ inputs.environment }}
  VERSION: ${{ inputs.version }}
  MAVEN_REPOSITORY: artifactory
  DEPLOYMENT_SSH_SERVER: ${{ inputs.SSH_Server }}
  DEPLOYMENT_SCRIPT_FILE: deploy.sh
```

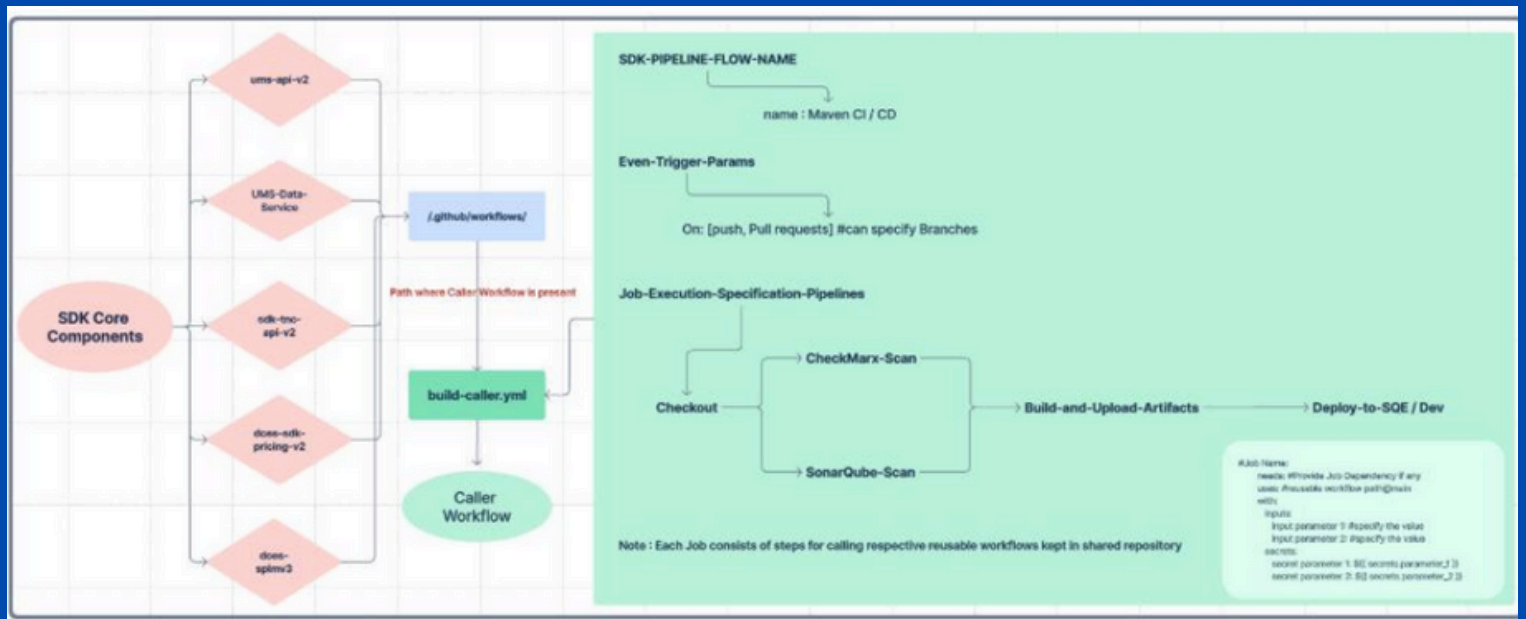




o Below flow chart is general workflow with centralized repo that we adopted.



o Below is a caller workflow implementation of SDK project.





Implementation and Demonstration

- **First Project: SDK Core Components** was the first project under DCES to migrate to GitHub Actions. This set a precedent and provided a reference for other projects.
- **Demo:** We conducted demonstrations for various Java-based projects, highlighting the advantages and ease of using GitHub Actions, which helped foster acceptance and adoption within the department.





Continuous Exploration and Improvement

- **Exploration:** We continuously explored new features and best practices of GitHub Actions. This included experimenting with different configurations, optimizing workflows, and integrating additional tools.
- **Feedback Loop:** We established a feedback loop with the development teams to gather insights and suggestions for further improvements. This iterative process helped in refining the workflows and addressing any issues promptly.
- **Documentation and Assistance:** Comprehensive documentation and timely assistance were provided to ensure that all team members could become familiar with GitHub Actions. This facilitated a smooth transition and empowered the teams to leverage the full potential of the new CI/CD pipeline.





Use of Organization-Level Runner

Organization-Level Runner: We utilized an organization-level runner for our GitHub Actions workflows. This runner was configured to handle the specific requirements of our projects, providing a consistent and reliable execution environment.

Benefits of Organization-Level Runner:

- o **Consistency:** Ensured a consistent environment across all projects, reducing the chances of environment-specific issues.
- o **Scalability:** Allowed us to scale our CI/CD processes efficiently by managing the runner at the organization level.
- o **Cost-Effectiveness:** Reduced the overhead of managing individual runners for each project, leading to cost savings.
- o **Adoption:** Other projects within the organization can easily adopt the organization-level runner by configuring their workflows to use it. This promotes standardization and simplifies the CI/CD setup across the organization.





Benefits of GitHub Actions

- **Integration:** GitHub Actions provided seamless integration with our GitHub repositories, making it easier to manage CI/CD pipelines.
- **Flexibility:** The matrix strategy and reusable workflows offered flexibility and reusability, reducing the effort required to set up pipelines for new projects.
- **Support:** GitHub's extensive documentation and community support, along with the assistance of GitHub Copilot, were instrumental in overcoming initial challenges and implementing best practices.











Conclusion

The transition from Jenkins to GitHub Actions was a resounding success, resulting in significant enhancements to our CI/CD processes. The implementation of reusable workflows, matrix strategies, organization-level runners, and the valuable support from GitHub were instrumental in this migration. This case study serves as a reference for other projects considering a similar transition.



Value Delivered to the Customer



	Reduction in overall operational infrastructure and maintenance cost by 40%.		Improved application scalability.
	Increased availability of the application by 70% with multi-region deployment.		Robust disaster recovery strategy delivered for a dynamic workload.
	Enhanced security.		Improved and efficient monitoring, logging and alerting system for the entire application.

For more information, write to us at hello@xfactor.ai

About XFactor.AI

At XFactor™.AI, we're revolutionizing industries with our bold 3D approach to digital transformation:

- 1 Digital Full-stack Development
- 2 Data Science & AI ML
- 3 DevOps & Cloud

Data is our backbone, and AI is our superpower. We blend traditional digital technologies with cutting-edge AI to create future-ready solutions that drive real impact. With a proven track record of building and exiting successful companies, our leadership brings decades of expertise in digital tech and AI to inspire trust, innovation, and results.

Whether empowering global enterprises or fueling disruptive startups, we turn bold ideas into intelligent products. Join us—we're on track to become the next unicorn by 2040! 🚀 People might call us mad, but we're focused, ambitious, and determined to make it happen.