

Vignette 1 – Implementing Continuous Delivery: The JIDO Approach

Forrest Shull

One theme that emerges from the work in this study is that DoD certainly does have successes in terms of modern, continuous delivery of software capability; however, in too many cases, these successes are driven by heroic personalities and not supported by the surrounding acquisition ecosystem. In fact, in several cases the demands of the rest of the ecosystem cause friction that, at best, adds unnecessary overhead to the process and slows the delivery of capability. The Joint Improvised-Threat Defeat Organization (JIDO), within the Defense Threat Reduction Agency, is a compelling example.

JIDO describes itself as “the DoD’s agile response mechanism, a Quick Reaction Capability (QRC) as a Service providing timely near-term solutions to the improvised threats endangering U.S. military personnel around the world.”¹¹ As such, the speed of delivery is a key success criterion, and JIDO has made important improvements in this domain. Central to accomplishing these successes has been the adoption of a DevSecOps solution along with a continuous ATO process, which exploits the automation provided by DevSecOps to quickly assess security issues.

At least as important as the tooling are the tight connections that JIDO has enabled among the stakeholder groups that have to work together with speed to deliver capability. JIDO has personnel embedded in the user communities associated with different COCOMs, referred to as Capability Data Integrators (CDIs). These personnel are required to be familiar with the domain, familiar with the technology, and forward-leaning in terms of envisioning technical solutions to help warfighter operations. Almost all CDIs have prior military experience and are deployed in the field, moving from one group of users to another, helping to train them on the tools that are available, and at the same time understanding what they still need. CDIs have tight reachback to JIDO and are able to identify important available data that can be leveraged by software functionality and can be developed with speed through the DevSecOps pipeline.

JIDO has also focused on knocking down barriers among contractors and government personnel. JIDO finds value in relying on contractor labor that can flex and adapt as needed to the technical work, with effort spent on making sure that the mix of government personnel and multiple contractor organizations can work together as a truly integrated team. To accomplish this, JIDO has created an environment with a great deal of trust between government and contractors. There are responsibilities that are inherently governmental and tasks that can be delegated to the contractor. Finding the right mix requires experimentation, especially since finding the personnel with the right skillset on the government side is difficult.

Despite these successes at bringing together stakeholders within the JIDO team, stakeholders in the program management office (PMO) sometimes describe substantial difficulties in working with the rest of the acquisition ecosystem, since on many dimensions the Agile/DevSecOps approach does not work well with business as usual. For example, they describe instances where the Services or the Joint Chiefs push back on solutions that were created to address requirements from the field. Thanks to the CDIs, JIDO can create a technical solution that answers identified

¹¹ JIDO SecDevOps Concept of Operations, v1.

requirements from warfighters in the field, but that does not mean it will get approval for deployment. There is a mismatch and potential for miscommunication when the organizations that control deployment don't own the requirements themselves.

Also, because JIDO operates in an agile paradigm in which requirements can emerge and get re-prioritized, it is difficult for the organization to justify budget requests upfront in the way that their command chain requires. JIDO addresses this today by creating notional, detailed mappings of functionality to release milestones. Since a basic principle of the approach is that capabilities being developed can be modified or re-prioritized with input from the warfighter, this predictive approach provides little or no value to the JIDO teams themselves. Even though JIDO refuses to map functionality in this way more than 2 years out, given that user needs can change significantly in that time, the program has had to add headcount just to pull these reports together.

JIDO has no problem showing value for the money spent. It is able to show numbers of users and, because it has personnel embedded with user communities, can discuss operational impact. As mentioned above, JIDO's primary performance metric is "response from the theater." Currently, JIDO faces a backlog of tasks representing additional demand for more of its services, as well as a demand for more CDIs. Despite these impactful successes, the surrounding ecosystem unfortunately provides little in the way of support and much that hinders the core mission. It is difficult to see how these practices can be replicated in other environments where they can provide positive impact, until these organizational mismatches can be resolved.

JIDO/J6 SECDEVOPS & NEXTGEN GOVERNANCE

Purpose: Illustrate CI/CD Pipeline, the related DevOps Workflow and Quality Gates to push to production with governance and transparency.

CI/CD Pipeline and DevOps workflow elements

- Code Supply Chain Management
- Container Scanning for Critical Vulnerabilities (CVE)
- Automated Testing
- Test Driven Development and Code Level Unit Test Coverage
- Static Code Analysis
- Penetration Testing
- Requirements Management System
- Pipeline Thresholds

Post approval the Software Development team will:

- Begin using the paths to production to deploy Frameworks via build 1.40
- CI/CD Candidates: Prepare boom and New Search applications for CM DevOps evaluation
- Evaluate additional applications for DevOps suitability
- Continuous Improvement: Iterate on the CI/CD Pipeline to automate additional processes

PIPELINE & WORKFLOW

A. Feature Sets [SIL]

- Contains links to JIRA issues/changes implemented in this release
- Contains build notes
- Conforms to quality gate restrictions

B. Daily DevOps Meeting [SIL, SIL Project Dev Lead, IA, CM, NetOps]

- Discuss Feature Sets that will be deployed via CI/CD Pipeline

C. Pipeline [SL] (Launches with a push to git)

- Git commit message contains the JIRA ID for the Feature Set
- Pipeline always releases to NDEV
- Pipeline may create deployment artifacts depending on Feature Set status

D. Transfer [CM]

- CM will transfer Deployment Artifacts from NDEV to SLAN on a set schedule (without notification)

E. Deployment [SL, NetOps]

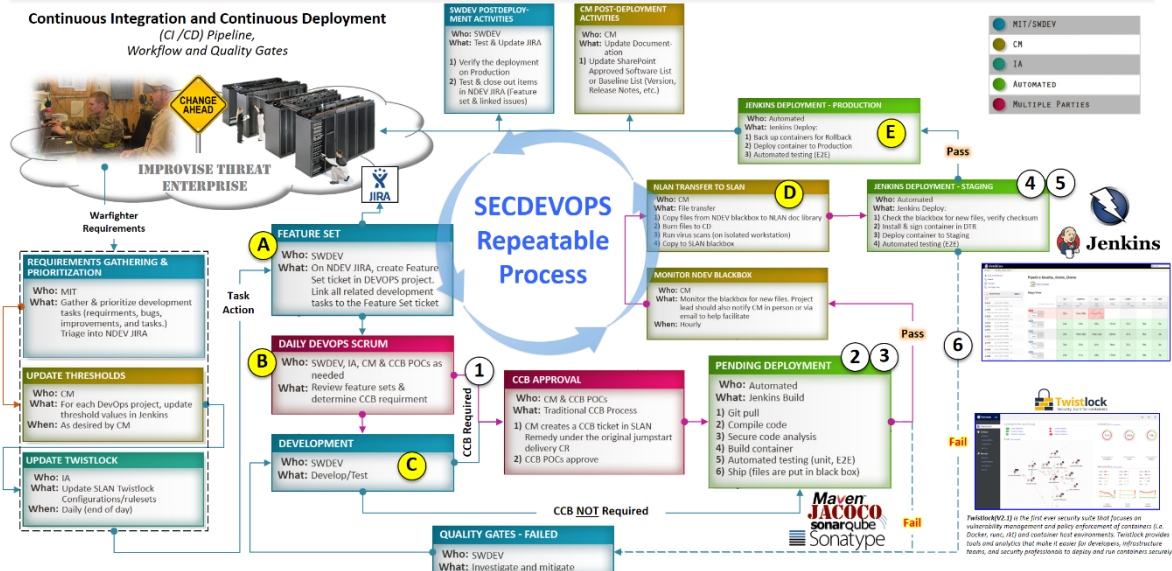
- Jenkins polls receiving directory for Deployment Artifacts
- Validate Deployment Artifacts
- Deploys project to DTR and Application Hosts
- Validate project has been deployed
- Update deployment status in Feature Set.

QUALITY GATES

- Project Status [Jira]**
 - Restrict deployment if Requires CCB
 - Restrict deployment if Development is not complete
- Unit Tests [Maven, JUnit5, Sonarqube]**
 - Exceed number of new lines
 - Code coverage percentage
- Static Code Analysis [Sonarqube, Sonatype]**
 - Quality of Code
 - Opportunity to refactor
 - Possible injection errors
 - Open source license requirements
 - Vulnerability scan
- EZE Tests [Selenium, Protractor]**
 - Automated integration tests
 - Ensure application is up and running
 - Stress Testing
- Penetration Testing [OWASP Zap]**
- Vulnerability Gate [Twistlock]**
 - Stop pipeline deployment on NDEV when vulnerability is detected
 - Restrict SLAN deployment when vulnerability is detected

Delivery Package

| | |
|----------------------|--|
| DevOps | DevOps |
| Deployment Files | Container MDS (hash) file SDO/VIDO file with links to: - Version # - Release notes/linked issues |
| Supporting Documents | JIRA Feature Set details -SCA report - Twistlock report |
| Code Scans | -SCA report - Twistlock report |
| Source Code | - Git - Container url |



Slide image received from former DTRA-JIDO chief technology officer.