Author: Vijay Gandhavale

Category: DevOps

Sub-category: DevOps with SAFe®

Nurture SAFe® Delivery in DevOps Culture

Problem Statement

General problem faced while using SAFe® is interdependencies between the teams consumes more time than calculated, specifically between development team and system team or deployment operations team.

Proposed Solution

How DevOps culture can solve this problem. Below are some DevOps principles that will solve this problem.

- **Consideration of Perspectives:** The cultural change must be considered from the perspective of both the driver and the participants. DevOps as an input into requirements for future investments in skills, technologies and processes related to its initiative.
- **Flexibility**: In case of exceptional situations, the organization must be flexible enough even to temporarily let go of DevOps values. DevOps helps to develop its road map, including, but not limited to, potential starting points and desired end states.
- **Continuous Integration Process**: A process must be in place to integrate the changes made.
- **Agile Decision Making**: The teams must have the agility to be able to decide the tools they need to use based on their own skills and expertise.
- **Collaboration and Transparency**: There should be no hidden agenda between the Dev and Ops teams.
- **Automation**: Always think of automating the process and task to reduce manual efforts and use appropriate DevOps tools. I.e. For build automation & integration, tools like Maven or Gradle with Jenkins, for packaging and imaging, Packer or Docker etc and for building the environments chef, puppet, ansible etc. can be used.

Consider a scenario wherein the development team triggers regular build to test environment, 2 days before end of sprint for QA acceptance. And team found that Test environment is responding very slowly and not able to perform deployment.

With SAFe® practice, Dev team is going to send mail or communicate the problem to Ops team, or by raising a Jira or incident to Ops team. But Ops team accept that ticket at low priority as there is high priority production deployment for previous sprint work going on.

Now problem remains as it is, how to resolve Test Env issue?

When DevOps culture is adopted by teams, dev team can work collaboratively with Ops or system team to understand the problem and getting into problem resolution as associate to Ops team.

Certain steps to clean up the test environment can be understood by a developer or dev team and they can try that steps to get that env up, instead of waiting for Ops team to work on that ticket, Dev team can work on that ticket and deploy their work on test env with some efforts.

Now questions arises is that environment issue is so frequent, and it takes much time of Dev team.

To tackle this issue team need to bring DevOps fully in SAFe® delivery.

Below are possible steps that team could consider.

1) To understand this problem properly, team need to spend some time in learning on how to analyze these type of problem, so need to have collaboration with system or Ops team. Plan some sessions with them and understand processes that they follow and their operational culture. Need to plan these activities as Technical debt in Program Increment (PI).

2) They could shape the product backlog with technical stories under Technical debt epic and distribute them across sprints in the PI in coordination with PO. Prioritizing them during the inception of the first PI would help in setting up the dev ops function as an integral part of development.

3) Once they start learning Ops team's skill to analyze these kind of problems. Team would gradually gain confidence over the process and eventually improve the productivity.

4) Getting that confidence team would be in position to categorize this problem into some of the frequently occurring incidents like server space, server memory, application space, different supporting processes etc. and classify into likely, most likely etc. Once the problem is identified, team can apply appropriate solution.

Case study and solutions

1) Consider an example where Test Env is running on Linux OS.
   If developer knows, there is 'TOP' command which can list down all the processes that are running on the server, then they can use it and see which processes are consuming more memory. By Collaborating with the Ops team, the development team would gather knowledge of many of such command like 'PS', 'DU' etc. and developers will be able to sort out these Ops issues on any such environment, thus making the total development activity agile. Going forward they can write shell script or python scripts to monitor the server as well as services running on the server.

2) Once the categorization of problem is done, below are some of the solutions for each category. (As per Prieto's principal top 20% issues cause 80% of the problems)

    a) Server space/Shortage of disk space: The solution to this issue is to basically free up disk space by doing either of the following

a. Archive Log files
b. Cluster DB tables
c. Free up temp space (cleaning temp directories and other housekeeping activities)

The above activities could be automated by writing shell/python scripts which would save man hours and improve sprint velocity. And ensuring higher the availability of environment.

b) Server Memory : The solution to this issue is to basically free up RAM by doing either of the following
   a. Check the applications and processes that are consuming higher memory
   b. Restart them if necessary.
   c. Kill the unnecessary processes.
   d. If the server is having JVM processes running, then they could increase the HEAP size, PermGen space and applying appropriate GC algorithm

   The above issues could be resolved by upgrading the Hardware (more RAM).

   These activities can be automated by using monitoring tools like Datadog, Appdynamics, Splunk etc.

c) Application Errors: This issue actually triggers one or more of the above issues. The solution to this issue is to basically analyzing application and server logs.
   a. Log analysis.
   b. Analyze memory map of the application (identifying memory leaks, Concurrency issues, Dead lock scenarios etc)
   c. Analyzing coding issues (null pointers or resource allocation).
   d. To improve code quality by performing static code analysis using open source plugins and tools like seamle, sonar.
   e. Automate testing by writing Unit, Integration and Behavioral test cases.

   To automate of above activities they could use some of the tools.
   Memory and Security: - Passlane, HP Sec,
   Log analysis: - Greylog, Logstash,
   Memory management: - Jprofile, J visual, Jmeter etc.

d) Other processes: Use appropriate build tools and APIs like Maven, Gradle, Ant etc. to build artifacts, CI tools and APIs like Jenkins and Bamboos to configure and deploy artifacts, repository systems to manage artifacts like GIT, SVN etc. This will improve the processes and velocity of SAFe® deliverables.

For more details on DevOps culture.

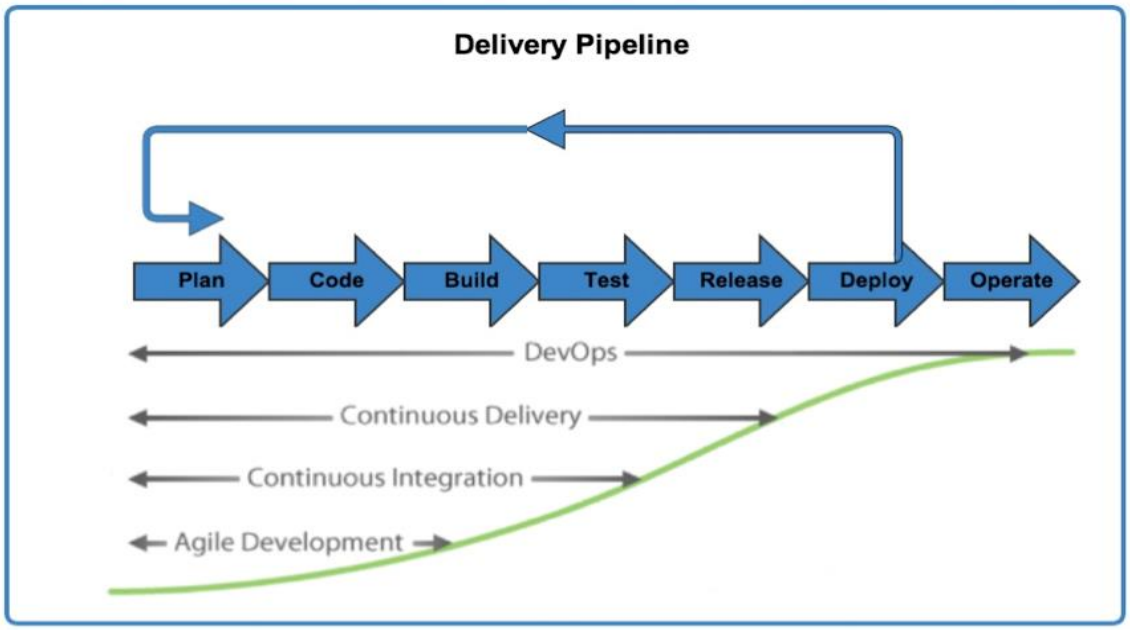

Figure 1 DevOps Culture

Above diagram shows the start and end point of DevOps in SAFe® delivery and how it overwhelms the SAFe® delivery.

Summary

This technical paper with mentioned case studies will help the organization to reach to an amalgamated DevOps state. This is where software is not just delivered continuously, but all of the operational needs are constantly met. Monitoring, metrics, the ability to automatically roll back software releases, scalability and much more are the key building blocks to a powerful DevOps organization.

With this thoughtful step-by-step process of DevOps adoption, organizations can meet the needs of business in a mindful and cohesive way without compromise.

References

http://devops.com

http://www.scaledagileframework.com/

https://www.gartner.com

http://www.gallop.net