Fortune 500 CICD/GitOps

Pelotech

Executive summary

Fortune 500 chemical manufacturing company had a functional but outdated ecommerce site that needed to be evaluated and modernized. As experts in delivering high availability stable services, Pelotech was brought in to evaluate the key areas of opportunity.

We examined existing systems and found a number of key problems hindering not only the client's ability to deliver a modernized highly scalable solution but also preventing effective fundamental working practices.

Solutions included:

- Transitioned from Request to Proposal coordination model
- Adopted a modern Continuous Integration process
- Adopted a GitOps based Continuous Delivery model
- Optimized and refactored Kubernetes
- Deferred: Continuous Deployment

By deploying these solutions the customer has been able to improve communication, dramatically improve code quality, code velocity and ability to add more value in shorter cycles.

Background

Client is a 300 plus year old chemical manufacturing company with global distribution both of clients and operations. They have a thriving existing ecommerce site generating multibillion dollar revenues.

That said, their existing website is built on a platform released in 1998 and is showing serious signs of age. In particular it struggled under heavy load, had regular outages, and was difficult to update.

The client had been in the process of rebuilding this site for several years and had abandoned attempts on at least two previous occasions. They hired a partner of Pelotech's to develop and modernize the front end application using NodeJs, React, and GraphQL.

Our partner reached out to us to help with CICD, infrastructure and Kubernetes based cloud deployments. The client in particular needed a solution for horizontal scale and global high availability.

Evaluation

We examined existing systems and found a number of key problems hindering not only the client's ability to deliver a modernized highly scalable solution but also preventing effective fundamental working practices.

- No automated verification ran anywhere in the process. If tests or static analysis existed they were run manually and never blocked code integrations or deployments.
- All build, test, deploy cycles were adhoc and no information was available on what was deployed where and when.
- All build scripts, deployment configurations, and even container image definitions were stored on random hosts machines with no source control or organized management.
- Requests for changes to deployments were difficult to make, track, and manage with frequent transcription errors.
- As a result of significant manual and adhoc practices, firefighting, failed attempts to centralize control, and constant infighting were rampant.

While applications were deployed to Kubernetes clusters, very little was optimized and anti-patterns such as live editing of container file systems were standard operating procedure.

Solutions

Transitioned from Request to Proposal coordination model

As a general improvement in process we worked with the client to transition their cross team coordination model from team members filing Jira or Service Now tickets against one another to teams providing merge requests with proposed changes. In particular in coordinating developer and operations changes this yielded massive improvements in communication, clarity and speed of issue resolution. Furthermore this combined with a sensible review process has all but eliminated transcription errors.

Adopted a modern Continuous Integration process

To ensure code is high quality and fully functional at all times Pelotech replaced an old version of Atlassian's BitBucket with a modern Kubernetes based GitLab installation enabling rich code reviews, linting, testing and static analysis.

Pelotech also decommissioned an existing Jenkins server filled with custom scripts, jobs, and manual configurations and replaced it with source controlled build processes for the 23 nodejs, java, and openresty based applications deployed as components of the ecommerce application.

Adopted a GitOps based Continuous Delivery model

Pelotech deployed the CNCF's Flux tool to synchronize Kubernetes configurations between the 13 globally distributed Kubernetes clusters expected to support the application and the GitLab installation. Pelotech was able to unify configurations across these environments and help teams refactor their applications to better support the right level of configuration. This drastically reduced environmental issues and one off cluster problems.

Optimized and refactored Kubernetes

Once Kubernetes configurations were source controlled, reviewing and optimizing deployment configurations was trivial. In addition to unifying configurations and simplifying application structure Pelotech helped the client adopt autoscaling and autohealing architectures, as well as straightforward disaster recovery and high availability plans. Pelotech helped the client enable sensible metrics strategies, dashboards and alerting with Prometheus, Grafana, and AlertManager as well as log management with ElasticSearch and Fluentd. Pelotech also led efforts in automating DNS and TLS certificate management with ACME.

Deferred: Continuous Deployment

Pelotech investigated a client's desire to reach a continuously delivered application deployment architecture. Together however we decided to take a more incremental approach in modernizing their workflow. The client was not fully ready to remove human approval steps from the deployment process. We also felt they needed time to build up the automated testing infrastructure and organizational comfort with the components. Moving from an approval based workflow to a proposal based one was already a significant change.

Conclusion

By moving from a request and approval based workflow to a proposal based workflow teams improved coordination, reduced frustrations and accelerated feature delivery. Core teams previously inundated with requests can rapidly approve proposals while leaf teams previously blocked and waiting are now able to provide assistance in servicing their own requests.

By adopting a standardized review process for code and deployment related files the entire organization now has a shared process for approving and reviewing changes. Release management approves new image deployments just as development leads approve code changes. This has led to a much clearer shared understanding of application state.

Development throughput has increased while code quality has also increased. By removing thirty minute manual test runs per commit for developers and shortening deploy cycles from hours to minutes we have nearly doubled average ticket throughput. At the same time by running linting, package and container auditing and unit tests on every change we are able to meaningfully improve quality and drastically reduce regressions.

By migrating all assets into git we have been able to hunt down ship stopper performance issues and cross architectural coordination concerns that placed the entire multi-year project at risk of failure.

Our work with this client continues at present with future efforts aimed at improving the organization’s ability to rapidly respond to emerging market opportunities With a focus in particular on shorter ship cycles, progress deployments, and Continuous Deployment.