Shift-left site reliability engineering for self-healing applications

Jürgen Etzlstorfer
Technology Strategist
@jetzlstorfer
Dynatrace

Florian Bacher
Technology Strategist
@bacherfl
Dynatrace
Our Challenge: Delivering better software faster
Collecting more evidence: https://dynatrace.ai/acsurvey

Commit Cycle Time: From Dev to Pro

95th Percentile

Goal: 1h to Production
Some results from our survey

95th Percentile

- 2 days
- 1 out of 10
- 0 hotfixes
- ~4 hours

Median

- 12.5 days
- 3 out of 10
- 3 hotfixes
- 4.8 days

Code to Production (Commit Cycle Time)

Business Impacting Deployments

Per Production Deployment

MTTR (Mean Time to Repair)

Evaluate for yourself: https://dynatrace.ai/acsurvey
What we want

- commit
- build
- test
- stage
- prod

Automatically push to next stage

What we (sometimes) have

- commit
- build
- test
- stage
- pre-prod1
- pre-prod2
- prod

Break pipelines early! (shift-left & quality gates)
Shift-left quality gates – performance-spec-as-code

```json
{
  "lowerBound": 1,
  "upperBound": 100,
  "comment": "global configuration environment-wide",
  "timeseries": [
    {
      "timeseriesId": "com.dynatrace.builtin:service.responsetime",
      "aggregation": "avg",
      "entityIds": "SERVICE-3211ABE8813B9239",
      "lowerBound": 1000000,
      "upperBound": 2000000
    }
  ]
}
```

Performance Signature

Build Performance Analysis

Automatic Feedback
How to increase quality

- Automate operations (self-healing) – auto-mitigate bad deployments in production
- Automate deployment (shift-right) – push “monitoring-as-code” for auto-validation and auto-alerting
- Automate quality (shift-left) – automate the pipeline and stop bad code changes before they reach prod
- Automated monitoring – monitoring as feature of the end-to-end pipeline
How to increase resilience

- commit
- build
- test
- stage
- prod

Auto-remediation-as-code
Auto-remediation building blocks

- **Monitoring**: know what’s going on in your applications
  - End-to-end
  - Full-stack – fully integrated in all stages

- **Automation/Execution**: perform mitigation/remediation actions
  - Access to all systems

Automation via APIs
Auto-remediation with Ansible (Tower)

- APIs are key to enable automation
- Ansible Tower provides rich API for managing Ansible jobs
- Playbooks can be orchestrated in workflows and job templates
Options to build (auto)remediation

https://medium.com/@sashman90/ops-mitigation-triangle-300c81d97df6
Introducing keptn
What we have seen organizations actually do!

• Building Custom Integrations with OpenSource & Commercial tools for
  • Deployment Pipelines
  • Testing Pipelines
  • Auto-Remediation
  • Notifications
  • Auditing
What we have seen organizations struggle with

Quote: „Pipelines seem to become our new future unmanageable legacy code!“

- Teams want to **stick with existing tools** to **protect** investment
- **Containing** lots of **custom code** for tool integration, error handling, logging, ...
- **Getting harder** to **maintain** the more tools get integrated
- **Pipelines** becoming more **complex** requiring **dedicated teams**
- **Uncoordinated** deployments between pipelines resulting in **unstable** Environments
Core capabilities

- Automated multistage unbreakable delivery pipelines
- Self-healing blue/green deployments
- Event-driven runbook automation

Design Principles

- GitOps-based collaboration
- Operator patterns for all logic components
- Monitoring and operations as code

- Built on and for Kubernetes
- Event-driven and serverless
- Pluggable tooling
keptn – conceptual architecture

Autonomous Cloud Control Plane

Core

keptn

Services

GitOps
Container Registry
Continuous Delivery
AIOps
Operations Automation
Test Automation
ChatOps
Data Provider

Platform

Dev Namespace
Staging Namespace
Production Namespace

Environment Definition (shipyard file)
keptn – Shipping through Unbreakable Continuous Delivery Pipelines

1: push
2: deploy dev
3: test
4: evaluate
5: promote
6: deploy staging
7: test
8: evaluate
9: promote
10: deploy production
11: evaluate
12: operate

Autonomous Cloud Control Plane

dev

stage

prod
Demo Time

keptn
Key takeaways

• Everything-as-code
  • Performance signature as code
  • Auto-remediation as code

• Increase resilience by automated quality checks in pipelines

• kep tn as open source tool for implementing unbreakable pipelines
  • https://keptn.sh
  • https://github.com/keptn
Even more talks!

- **Building autonomous operations for Kubernetes with keptn**
  Tuesday May 7th 2.30-3.15pm
  Speaker: Alois Reitbauer

- **Dynatrace Operator**
  Wednesday May 8th 4.20-4.40pm + Thursday May 9th 11.30-11.50am
  Speaker: Markus Heimbach

- **Unifying OpenShift cluster, container and application monitoring**
  Wednesday May 8th 3.30-3.50pm
  Speaker: Asad Ali

- **Unbreakable DevOps on Red Hat OpenShift**
  Thursday May 9th 1-1:45pm
  Speakers: Peter Hack & Florian Bacher
Thank you!