Engineering Service Catalogs for DevOps

This paper is derived from selections from my book.



The book “Engineering DevOps” can be obtained here: [**mybook.to/engineeringdevops**](http://mybook.to/engineeringdevops)

This white paper and other useful papers, glossary checklists, slides, blueprints, templates and engineering practices can be downloaded from [**https://engineeringdevops.com/documents**](https://engineeringdevops.com/documents)

About the Author:

**Marc Hornbeek** a.k.a. DevOps\_the\_Gray esq.

[linkedin.com/in/marchornbeek](https://www.linkedin.com/in/marchornbeek)

June 6, 2020

© Engineering DevOps Consulting 2019, 2020

All rights reserved

**How do you Engineer Service Catalogs for DevOps?**

A DevOps service catalog is an organized and curated collection of business- and IT-related DevOps services that can be performed by, for, or within an enterprise.

DevOps Service catalogs act as knowledge management tools for the employees and consultants of an enterprise, allowing them to route their requests for and about DevOps services and related topics to the subject matter experts who own, are accountable for, and operate them.

Each service within DevOps service catalog is very repeatable and has controlled inputs, processes, and outputs.

When engineered properly, this approach ensures that the application, pipeline, and infrastructure choices that are available from the service catalog includes tools that are needed to measure and calibrate the performance of those choices.

**Figure 1—DevOps Service Catalog Examples** shows two example commercial service catalogs for DevOps.

**Why Is the Service Catalog Important to DevOps Engineering?**

A DevOps service catalog allows organizations to create, centrally manage, and execute applications, toolchains, and infrastructures. For example, a toolchain instance (or collection of toolchain instances) may map to a type of application within the enterprise.

A service catalog represents information about all applications, pipelines, and infrastructures that are available and operated within an enterprise.

Implementing DevOps in large organizations is difficult. Each application can potentially have multiple environments each with multiple and varied CI/CD pipelines, tool stacks, metrics, and workflows. Calcified silos result from the inherent legal, geographical, and financial constraints. This makes them culturally resistant to change and less agile.

DevOps service catalogs can play a key role in resolving increasing expectations from the business for capabilities like self-service and on-demand provisioning of resources and growing impatience with trouble-ticket culture.

DevOps service catalogs can also support the desire to experiment with rapidly evolving productive platforms (e.g., cloud-based) that can accelerate the time to market while controlling spending and technology choices and enforcing corporate policies.



***Figure 1—DevOps Service Catalog Examples***

**How Are DevOps Service Catalogs Engineered for DevOps?**

Service catalogs allow application development teams to onboard their applications and microservices easily with pre-defined templates and pre-approved technologies for typical projects. Metrics-based goal tracking is the foundation for IT performance. As indicated by ***Figure 2—Continuous Feedback Service Catalog Metrics***, tracking performance of applications, databases, continuous testing, releases, infrastructure, and continuous security for a portfolio of rapidly changing products is not feasible without a consistent feed- back strategy.

A DevOps service catalog is used to ensure Continuous Feedback monitoring tools are part of DevOps pipelines and that Continuous Feedback monitoring tool choices are consistent, so performance data is made visible across the portfolio of applications.



***Figure 2 —Continuous Feedback Service Catalog Metrics***

As indicated by ***Figure 3—Continuous Feedback Monitoring Framework***, monitoring tools at each layer feed logs and event data to VSM tool to provide an aggregated view of feedback data.



***Figure 3—Continuous Feedback Monitoring Framework***

The ***Third Way of DevOps*** prescribes continual experimentation, which requires taking risks, learning from success and failure, and understanding that repetition and practice are the prerequisites of mastery.

As indicated in ***Figure 4—Flexible Standardization***, with The Third Way, IT must have controlled flexibility to at once ensure standards for toolchains, application stacks, and environments that are used for DevOps and SRE while also facilitating experimentation and changes requested by users. The DevOps service catalog approach needs to be sophisticated able enough to do both.

***Figure 5 —Flexible Service Catalog Management*** Workflow illustrates a DevOps service catalog protocol that offers controlled flexibility.



***Figure 4—Flexible Standardization***

The following are recommended engineering practices for a DevOps service catalog are broken into two categories: DevOps Capabilities Creation and DevOps Capabilities Management.

**Engineering practices for DevOps Capabilities Creation** are detailed as follows:

• Users can author a toolchain based on a Domain-Specific Language (DSL) to ensure that only “approved” tools are included in the toolchain.

• Deployment resources (cloud provider, resources) that need to be provisioned are included.

• Once the toolchain for a given application type has been authored and made available within the DevOps service catalog, users can self-service provision an application via a central UI Portal.

• Users and user groups may customize a custom portfolio for the DevOps service catalog.

• The DevOps service catalog DSL is extensible to include the new tools and services being made available at a rapid clip.

• The DevOps service catalog tool provides the ability to import pre-existing templates from a library of available templates.

• The DevOps service catalog supports a quick and easy way to update the offerings on demand with approval capability.



***Figure 5—Flexible Service Catalog Management Workflow***

***Engineering practices for DevOps Capabilities Management are*** detailed as follows:

• With Role-Based Access Control (RBAC), an end-user can access only the resources that they (or their team) provisioned via the DevOps service catalog—they do not have permission to see resources provisioned by other teams.

• Toolchains include a workflow of steps for approval, escalation if milestones are not met, and notification upon successful completion. This workflow capability is a way to bridge application process guidance and operational process guidance.

• Users can access the status of their submitted DevOps service catalog requests via the UI portal.

• Users can access charge-back information via the UI portal of the DevOps service catalog.

• Control over the resources that are included in the DevOps service catalog is strictly enforced.

• The DevOps service catalog tool supports creating and editing policies. For example, “Enforce an Enterprise-specific naming convention on all provisioned resources.”

• Pipeline tools available in the DevOps service catalog include Continuous Feedback monitoring tools needed to support data collection for Continuous Feedback metrics.

• DevOps service catalog items are access controlled so that only permitted users can change certain items.

• The evolution of the DevOps service catalog itself follows a DevOps approach in which new versions are controlled in a version management database; releases of new versions of the DevOps service catalog follow a controlled and measured pipeline and can be deployed or rolled back quickly.

• The DevOps service catalog allows users to choose to compose a pipeline from a pre-approved selection of standard toolchains, application stacks, and environments.

• Dev and SRE teams may request changes to the toolchain templates, application stacks, and environments lists in the DevOps service catalog for experimentation purposes.

• Records are kept of DevOps service catalog change requests.

• Rules are determined for approving updated or new standard toolchain templates, application stacks, and environments to the DevOps service catalog. For example, SREs must approve any changes that may impact SRE reliability or capacity management processes.

**What Is Needed to Engineer a DevOps Service Catalog?**

A successful approach to implement a DevOps service catalog includes the following:

• Develop a unified DevOps model that can deal with a large and diverse application portfolio in a consistent manner. This includes consistent monitoring, incident handling, and logging.

• Implement an approach that bridges the application-focused frameworks such as Agile with more operations-centric frameworks like ITIL.

• Break down silos and adopt DevOps without embarking on a “boil the ocean” culture change or completely redrawing the organizational boundaries.

• Extend private data center infrastructures to hybrid and multicloud environments so application teams may take advantage of the elasticity and scale offered by the public cloud.

• Start with a small team with a first mover “model” application. Set up clear metrics and automating error-prone and tedious tasks.

• Connect “islands of automation” into a toolchain and bring consistency to aspects like provisioning and deployment.

• Use DSLs (domain specific language; e.g., Azure ARM Templates, Chef Recipes, and AWS Cloud Formation) to bring consistency to the DevOps toolchain. DSL is used to define the order of the tools in the chain, as well as the expression of policies, such as security and configuration. Treating each instance of a toolchain as code has many benefits, including the ability to repeat, the ability to version, and the ability to execute.

• Select tools to implement the service catalog (e.g., AIS Service Catalog, Service Now, etc.)

• Document a DevOps service catalog publishing governance guideline.

• Provide DevOps service catalog development support so the development team can contribute to new and improved service catalog content.

**Summary**

In summary, the answer to the question: “**How do you engineer Service Catalogs for Devops?”** has been explained in this white paper. It was explained how Service Catalogs play a useful role in ensuring the choices for applications, pipelines, and infrastructures have built-in tools needed to meet good engineering practices. A well- engineered service catalog goes a long way to supporting governance. The catalog presents a limited number of choices, which helps to contain the possible combinations that need to be governed.

**Learn More**



The book “Engineering DevOps” can be obtained here: [**mybook.to/engineeringdevops**](http://mybook.to/engineeringdevops)

This white paper and other useful papers, glossary checklists, slides, blueprints, templates and engineering practices can be downloaded from <https://engineeringdevops.com/documents>

About the Author:

**Marc Hornbeek** a.k.a. DevOps\_the\_Gray esq.

[linkedin.com/in/marchornbeek](https://www.linkedin.com/in/marchornbeek)

© EngineeringDevops 2019, 2020

All rights reserved