

How Netflix OSS Helps Companies Build Microservices Architectures

Netflix was among the first companies to publicize its move to microservices from a monolithic architecture. The Netflix backend application has been powered by microservices since 2009. The entire customer-facing Netflix website has run entirely on AWS and microservices since the end of 2010, and the company has been sharing information about its technology use since 2010. In 2012, Netflix began to open-source its code, allowing other companies to use its libraries to build their own AWS cloud-based microservices architectures.

This article, part of *ProgrammableWeb*'s [microservices series](#), highlights several Netflix open source software projects for building microservices architectures and highlights other microservices-related open source

software projects.

Netflix Open Source Software Center

Netflix is committed to open source. Netflix both leverages and provides open source technology focused on providing the leading Internet television network. Our technology focuses on providing immersive experiences across all internet-connected screens. Netflix's deployment technology allows for continuous build and integration into our worldwide deployments serving members in over 50 countries. Our focus on reliability defined the bar for cloud based elastic deployments with several layers of failover. Netflix also provides the technology to operate services responsibly with operational insight, peak performance, and security. We provide technologies for data (persistent & semi-persistent) that serve the real-time load to our 62 million members, as well as power the big data analytics that allow us to make informed decisions on how to improve our service. If you want to learn more, jump into any of the functional areas below to learn more.



Big Data

Tools and services to get the most out of your (big) data

Data is invaluable in making Netflix such an exceptional service for our customers. Behind the scenes, we have a rich ecosystem of (big) data technologies facilitating our algorithms and analytics. We use and contribute to broadly-adopted open source technologies including Hadoop, Hive, Pig, Parquet, Presto, and Spark. In addition, we've developed and contributed some additional tools and services, which have further elevated our data platform. [Genie](#) is a powerful, REST-based abstraction to our various data processing frameworks, notably Hadoop. [Inviso](#) provides detailed insights into the performance of our Hadoop jobs and clusters. [Lipstick](#) shows the workflow of Pig jobs in a clear, visual fashion. And [Aegisthus](#) enables the bulk abstraction of data out of Cassandra for downstream analytic processing.



Build and Delivery Tools

Taking code from desktop to the cloud

Netflix has open sourced many of our Gradle plugins under the name [Nebula](#). Nebula started off as a set of strong opinions to make Gradle simple to use for our developers. But we quickly learned that we could use the same assumptions on our open source projects and on other Gradle plugins to make them easy to build, test and deploy. By standardizing plugin development, we've lowered the barrier to generating them, allowing us to keep our build modular and composable.

We require additional tools to take these builds from the developers' desks to AWS. There are tens of thousands of instances running Netflix. Every one of these runs on top of an image created by our open source tool [Aminator](#). Once packaged, these AMIs are deployed to AWS using our Continuous Delivery Platform, [Spinnaker](#). Spinnaker facilitates releasing software changes with high velocity and confidence.



Common Runtime Services & Libraries

Runtime containers, libraries and services that power microservices

The cloud platform is the foundation and technology stack for the majority of the services within Netflix. The cloud platform consists of cloud services, application libraries and application containers. Specifically, the platform provides service discovery through [Eureka](#), distributed configuration through [Archaius](#), resilient and intelligent inter-process and service communication through [Ribbon](#). To provide reliability beyond single service calls, [Hystrix](#) is provided to isolate latency and fault tolerance at runtime. The previous libraries and services can be used with any JVM based container.

The platform provides JVM container services through [Karyon](#) and [Governator](#) and support for non-JVM runtimes via the [Prana](#) sidecar. While Prana provides proxy capabilities within an instance, [Zuul](#) (which integrates Hystrix, Eureka, and Ribbon as part of its IPC capabilities) provides dynamically scriptable proxying at the edge of the cloud deployment.

Netflix OSS for Building Microservices Architectures

The official [Netflix OSS site](#), currently lists more than 30 open source projects, some of them extending beyond microservices. The code helps developers handle several aspects of cloud platform development, such as big data analytics, storing and serving data in the cloud, improving performance, and ensuring security at scale.

Here we focus on some of the most popular Netflix OSS projects for building microservices architectures.

Archaius

[Archaius](#) is a Java library for distributed configuration. It includes a set of APIs that Netflix uses for configuration management. Netflix built Archaius to enable changes to the behavior of deployed services dynamically at runtime. This ability helps ensure the availability of deployed applications and also allows properties to be changed dynamically. For example, specific application features could be enabled or disabled dynamically based on the request context, such as the country of origin, specific server region, specific server instance, or the user's device.

Eureka

[Eureka](#) is a REST-based registry service that features the Eureka Client, a Java-based client component that improves interactions with the service. Eureka includes a built-in load balancer using basic round-robin load balancing for mid-tier load balancing.

Microservices architectures often consist of dozens, sometimes hundreds, of fine-grained services. Amazon Web Services (AWS) adds additional wrinkles. As the Eureka [documentation](#) explains, "In AWS cloud, because of its inherent nature, servers come and go. Unlike the traditional load balancers which work with servers with well-known IP addresses and host names, in AWS, load balancing requires much more sophistication in registering and de-registering servers with load balancer on the fly. Since AWS does not yet provide a middle tier load balancer, Eureka fills a big gap in the area of mid-tier load balancing." Eureka makes it possible for clients

to discover the addresses of available endpoints; it also keeps track of application-specific Web service metadata and Web service availability.

Hystrix

Availability is one of the most important benefits of building applications based on microservices architecture. Netflix built [Hystrix](#), a latency and fault tolerance library, as a means to control the interactions between microservices. The library isolates points of access between microservices, which helps prevent cascading failures and fault tolerance problems.

Hystrix helps prevent distributed systems from going down due to failures of individual services or dependencies. Hystrix does this by isolating points of access between third-party libraries and services, stopping cascading failures across services, providing fall-back options, and gracefully degrading when possible.

In its [announcement post](#), Netflix explained, "Tens of billions of thread-isolated and hundreds of billions of semaphore-isolated calls are executed via Hystrix every day at Netflix and a dramatic improvement in uptime and resilience has been achieved through its use."

Ribbon

[Ribbon](#) is a client-side inter-process communication (IPC) library which takes advantage of client-side software load balancing algorithms, including load balancing strategies such as simple round robin, weighted response time, zone-aware round robin, and random load balancing. Among the modules are several REST APIs, such as APIs to integrate load balancing, fault tolerance, client configuration. Netflix described its use in a [blog post](#),

wherein engineers use Hystrix to wrap Ribbon API calls for a greater tolerance of failure and latency.

Distributed applications often consist of dozens, sometimes hundreds of services. In a large microservices architecture, communicating between services could take many network hops. Enabling a client-side load balancer like Ribbon in a microservices architecture helps reduce latency, because it allows services to intercommunicate using one network hop. Ribbon reduces the number of hops needed to communicate between services reducing latency, which in turn improves performance.

Zuul

In many ways, [Zuul](#) is the glue holding together the Netflix microservices architecture. It provides insights into other components of the Netflix architecture and makes it possible for Netflix engineers to find, isolate, and fix problems among the massive number of requests.

Netflix uses Zuul as a gateway service for its streaming application and the Netflix API. Among its functions are dynamic routing, canary and stress testing, load shedding, static response handling, authentication and security, and insights and monitoring.

Zuul provides a unified interface for distributed applications. This means that Web browsers, game consoles, and mobile applications can consume services from multiple hosts without having to manage authentication or cross-origin resource sharing (CORS) for each and every Web service in a distributed application.

Companies Using and Contributing to Netflix OSS Projects

Once Netflix open-sourced its code, plenty of developers took advantage and contributed to make the libraries even more powerful. Today, quite a few companies have built microservices architectures with the help of Netflix OSS and are active contributors to Netflix OSS projects. Here are a just a few examples.

Dynatrace

Dynatrace, which provides full-stack performance management tools, implemented Eureka and Hystrix to support its monitoring of the orchestration layer in microservices environments, according to a recent Dynatrace Ruxit [blog post](#). Organizations that use Dynatrace can add Netflix OSS monitoring by downloading a plugin file and uploading it to their Dynatrace Ruxit environments.

Nirmata

A [detailed Nirmata post](#) explains how the company moved to a microservices architecture using a combination of Netflix OSS and Docker. Recognizing that they needed more infrastructure components than regular three-tier applications (such as a service registry, traffic gateway to control the traffic to the services, and a framework to develop uniform REST API), Nirmata turned to the Netflix code, because “Netflix infrastructure has been battle tested at scale ... real scale.” Four Netflix OSS components --- Zuul, Eureka, Archaius, and Ribbon – allowed the company to deliver on their promises of continuous delivery and helped the company focus on solving challenges related to distributed applications.

Ordina

Earlier this year, Ordina was [officially added](#) to the list of active users and contributors to Netflix OSS. The announcement blog post mentions that the company has successfully helped clients in Belgium move to microservices by using the Netflix and Spring Cloud stack. Spring Cloud Netflix (by Pivotal) provides Netflix OSS integrations for Spring Boot apps, making it easier for companies to adopt and use microservices.

Other Microservices Open Source Software Projects

Netflix was among the first companies to release open source software for microservices architectures, but others have joined in the fun. Here are just a few:

Apollo

Launched in May 2015 by Capgemini, [Apollo](#) is an open source platform for powering microservices, big data platforms, and next generation applications. Capgemini's Apollo project was initially used internally to power microservices and big data platforms for its clients.

fabric8

Part of a community of Red Hat projects, [fabric8](#) is an open source platform for creating and managing microservices and integrations. Based on Docker, Kubernetes, and Jenkins, fabric8 aims to make it easier for developers to deploy Java integration solutions and services on multiple machines as well as in containers, processes, and JVMs.

Gizmo

The New York Times' [Gizmo](#) is an open source tool kit to help developers build microservices APIs and pub/sub daemons written in the Go programming language. According to the [announcement post](#), The New York Times uses Go primarily for building JSON APIs.

Kong

[Kong](#) is an open source, scalable API gateway that was first released by Mashape in April 2015. Its API gateway provides a single, unified entry point whereby clients can access the multiple services of a distributed application. Mashape originally built Kong to secure, manage, and extend the thousands of APIs and microservices for its API marketplace.

Mantl

[Mantl](#) is an open source project built by Cisco to help developers deploy its own microservices platform without having to write any glue code. Mantl combines Docker, Mesos, Terraform, and other core components. According to the [documentation](#), “The base platform contains control nodes that manage the cluster and any number of agent nodes. Containers automatically register themselves into DNS so that other services can locate them.”

Spring Cloud

[Spring Cloud](#) is part of Spring (by Pivotal), an open source application development framework for enterprise Java. Spring Cloud includes several projects that involve microservices, including; Spring Cloud Data Flow,

Spring Cloud Stream, Spring Cloud Stream Modules, and Spring Cloud Task. Spring Cloud Netflix is a project that provides Netflix OSS integrations (Eureka, Hystrix, Zuul, and Ribbon) for Spring Boot apps.

...And that's just the beginning

Netflix continues to provide open source software for many of the technologies that power its cloud platform including technologies for powering microservices. Many companies have built AWS cloud-based distributed applications, thanks to the open source software provided by Netflix.