Moving into a serverless world with Tekton and Knative

Presenter: Veer Muchandi
Title: Chief Architect - Container Solutions, NACS
Social Handle: @VeerMuchandi
Blogs: https://blog.openshift.com/author/veermuchandi/

Serverless Model

Serverless computing is a cloud-computing execution model in which the cloud provider acts as the server, dynamically managing the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity.[1] It is a form of utility computing.

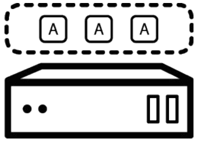
So, how can enterprises running kubernetes platform use this serverless computing model?

Knative Overview - Components

.. a Kubernetes-based platform to deploy and manage modern serverless container based workloads. Extends K8S with CRDs

Knative Serving

Deploy serverless containers, can “scale to 0”, snapshots of code and configuration



Knative Eventing

Common infrastructure for consuming and producing events that stimulate applications

* Knative Eventing is Tech Preview with OpenShift Container Platform 4.4

Tekton Pipelines (formerly Knative Build, branched out)

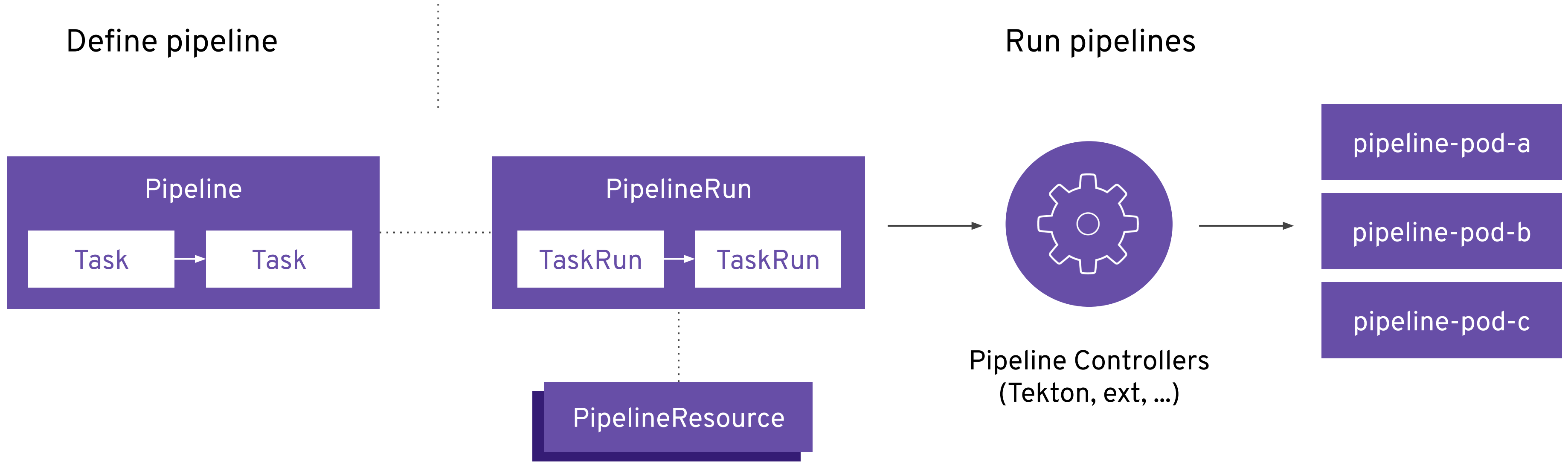
Tekton

Provides Kubernetes native modern resources for declaring CI/CD pipelines.

* Tech Preview with OpenShift Container Platform 4.4

Tekton

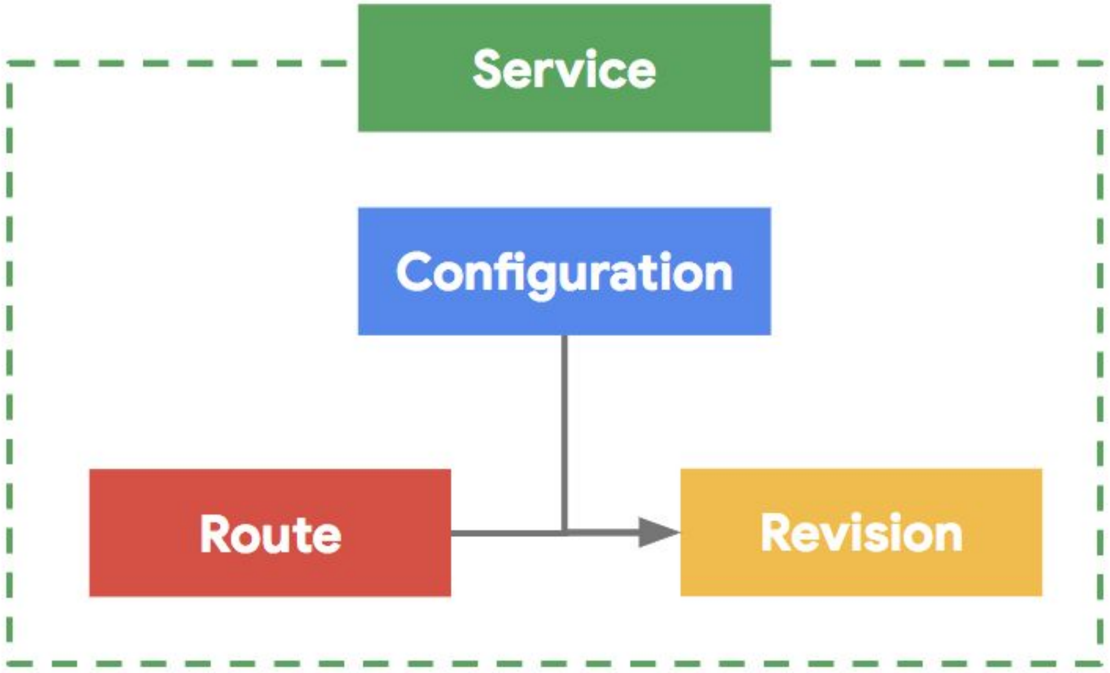
OpenShift Pipelines Architecture



Knative Serving

Knative Serving

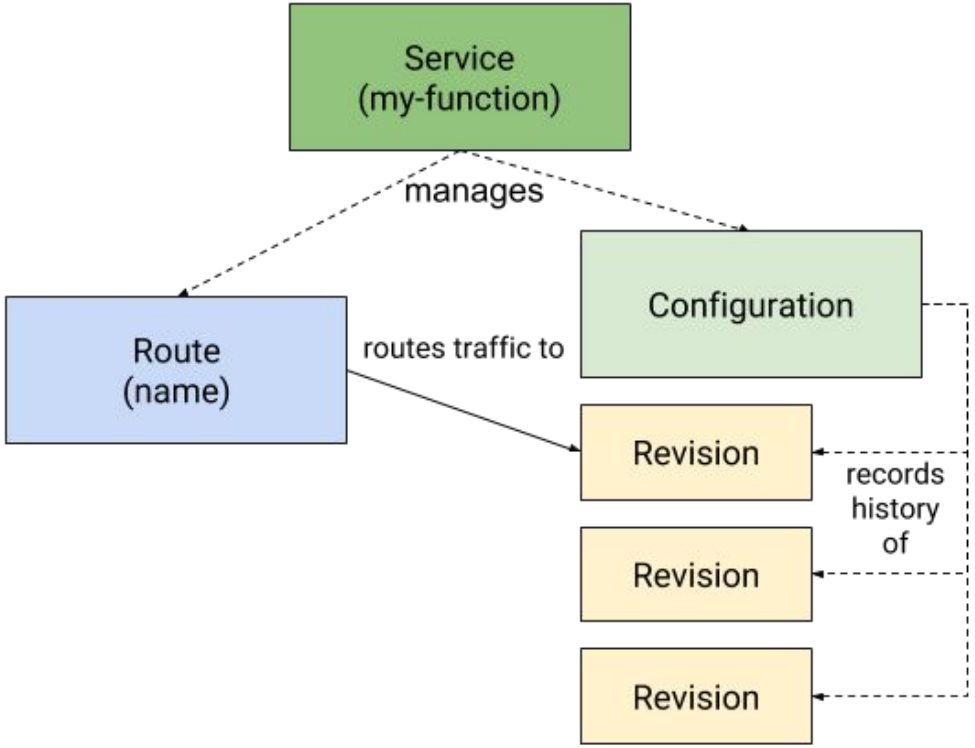
.. serverless primitives with containers



Service—is the combined lite version of all the objects below to enable simple use cases

- **Configuration**—is the desired state for your service, both code and configuration
- **Revision**—represents an immutable point-in-time snapshot of your code and configuration
- **Route**—assigns traffic to a revision or revisions of your service

Serving Resources



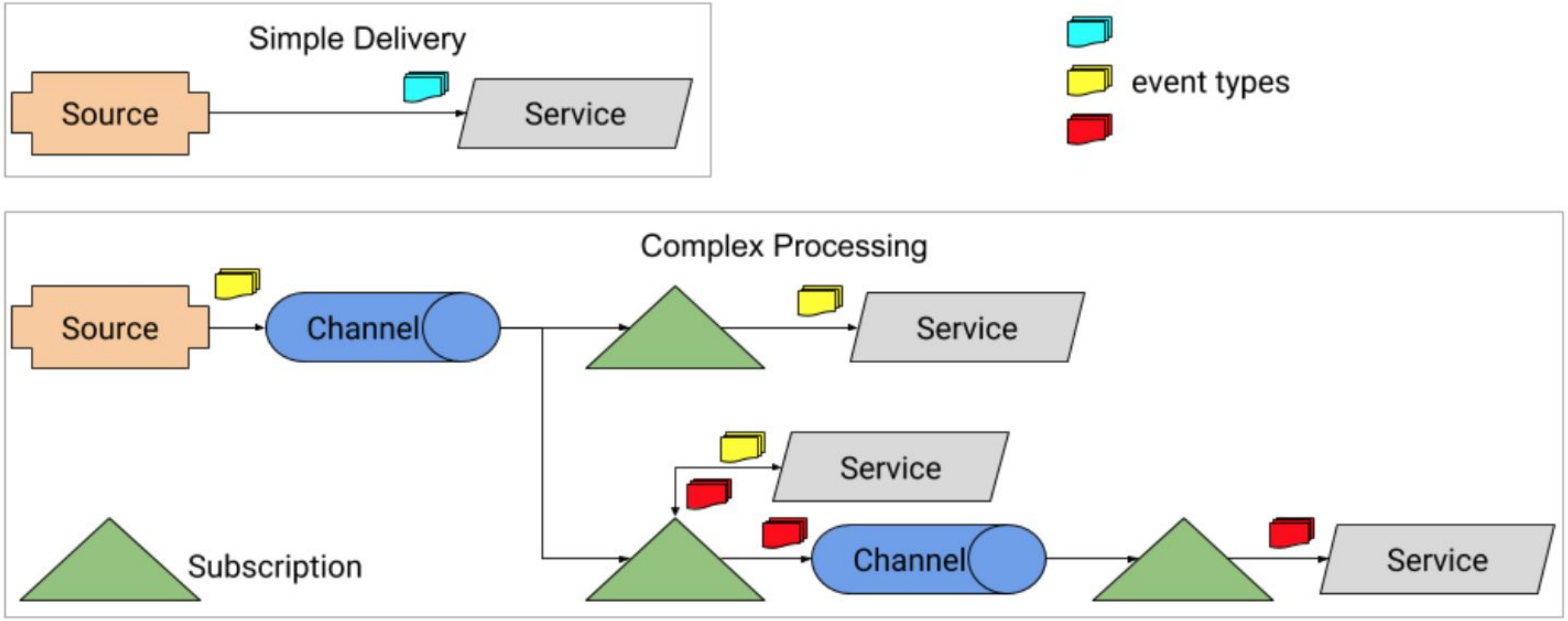
Knative Eventing

Knative Eventing

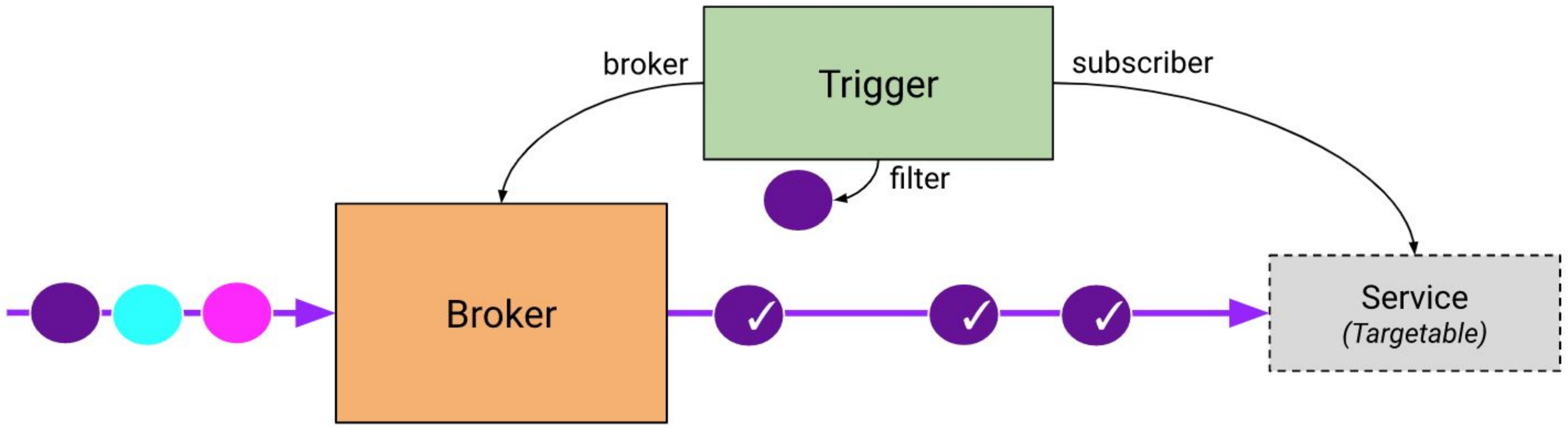
.. composable primitives for late binding event sources and event consumers

- **Event Source:** fires events. KubernetesEventSource, GithubSource, PingSource, KafkaSource etc,
- **Event Consumer**
  - **Addressable:** Receive and Acknowledge event delivery
  - **Callable:** Receive, Transform and Return 0 or 1 event
- **Event Broker**: Bucket of events. Receives events and forwards to subscribers
- **Event Trigger:** a filter on event attributes which should be delivered to an Addressable
- **Event Registry:** a collection of event types to create a trigger
- **Event Channel:** an event forwarding and persistence layer. Implementations - InMemoryChannel, KafkaChannel, NATSChannel
- **Event Subscription:** register to receive traffic from a Channel

Event Delivery Mechanisms



Knative Eventing: Brokers and Triggers



Demos

https://github.com/RedHatWorkshops/knative-on-ocp4

Thank you