Sidecar containers can be used to increase the functionality of the main container.

Types of sidecar containers:

1. Init Containers

2. Adapter Containers

3. Ambassador Containers

**Init Containers:**

A Pod can have multiple containers running apps within it, but it can also have one or more init containers, which run before the app containers are started.

<https://medium.com/bb-tutorials-and-thoughts/kubernetes-learn-init-container-pattern-7a757742de6b>

Init containers are exactly like regular containers, except:

* Init containers always run to completion.
* Each init container must complete successfully before the next one starts.
* Init containers are executed sequentially.

If a Pod's init container fails, the kubelet repeatedly restarts that init container until it succeeds. However, if the Pod has a restartPolicy of Never, and an init container fails during startup of that Pod, Kubernetes treats the overall Pod as failed.

To debug init containers: **kubectl logs <pod-name> -c <init-container-name>**

**Adapter Containers:**

Adapter containers runs along with main container and perform the

activities which are asked to do. Example, copy the files or logs from emptyDir and send them to AWS S3 or Elastisearch.

<https://medium.com/bb-tutorials-and-thoughts/kubernetes-learn-adaptor-container-pattern-97674285983c>

**Ambassador Containers:**

The Ambassador container is a special type of sidecar container which simplifies accessing services outside the Pod. When you are running applications on kubernetes it’s a high chance that you should access the data from the external services. The Ambassador container hides the complexity and provides the uniform interface to access these external services.

<https://www.magalix.com/blog/kubernetes-patterns-the-ambassador-pattern>

<https://medium.com/bb-tutorials-and-thoughts/kubernetes-learn-ambassador-container-pattern-bc2e1331bd3a>