

# Potential Experiments/areas to tackle

## Scenario

- Existing Egeria cohort including metadata repository, UI components
- Kafka already deployed
- I want to add a connector to a new technology
  - build my **integration** connector / metadata access store
  - create my image
  - deploy it in k8s
  - must work as part of existing cohort

## Proof points

#	Area	Objective	Current Status	Who
1	Containers	<p>Build customized containers with only the content required for a particular usage:</p> <ul style="list-style-type: none"> <li>• Start with integration connector</li> <li>• figure out approach to gathering correct runtime dependencies easily</li> <li>• figure out parameterization needed to make this a useful starting point for orgs</li> <li>• Exclude:</li> </ul> <p>Value: Can help in short term with existing model - already see need for this</p>	Already being looked at for IBM/HMS	Nigel
2a	Application	<p>Create a streamlined application which can startup a single server (and shutdown cleanly - we have concern here with egeria) when application is launched, based off an existing configuration.</p> <p>Should be runnable from the command line, and just launch as a process. It can consume files/environment variables (in k8s these can be mapped from config maps, secrets etc). will not update configuration</p> <p>Figure out how much, if any, of existing platform/admin is needed/can be reused.</p> <p>Specifically needs to be able to run a server hosting an integration connector</p> <p>Value: Could be used even in short-medium term as a convenience to start single platform</p>		Ljupch o ++ David
3	Operator	<p>Create custom resource definition for a 'server' (just 1 per 'platform' aka java process) &amp; create an operator that can deploy/undeploy a pod containing running this server, as well as a service based on a pre-existing configuration:</p> <ul style="list-style-type: none"> <li>• probably Java (rather than go) - general team skills?</li> <li>• use autostart as an initial 'hack'</li> </ul> <p>Value: Allows for active management in k8s environment</p>	see egeria-kubernetes-operator for a) platform operator in go b) initial port of above to java	Nigel
4	Document	<p>Document our design principles ie only use of ephemeral storage (create another wiki page?)</p> <p>Value: Information sharing, reviews, consensus</p>	<a href="#">Design Principles for Cloud native Egeria</a>	all
5	Configuration /storage	<p>Determine how egeria server has no dependency on persistent storage – ie which connectors/mechanisms are needed for storage ie potentially including</p> <ul style="list-style-type: none"> <li>• Configuration</li> <li>• Cohort registry</li> <li>• audit log</li> <li>• (metadata repositories)</li> </ul> <p>Consider use of existing kubernetes resources - directly or via mapping - config maps, secrets, custom resources</p> <p>Look at read/write, concurrent access from multiple servers or replicas (for example we know config documents get written to during startup)</p> <p>All must be accessible from a k8s operator in addition to other ways</p> <p>Develop appropriate connectors and/or techniques to support</p> <p>Excluded: security connector</p> <p>Value: Simplification and ultimately critically important in k8s environment</p>	Existing operator uses configmaps	nigel taylan

