




Pros and cons of using Kubernetes as a development platform

Who we are



Software Engineer  **Red Hat**

Eclipse Che committer 

workspaces.openshift.com



Software Engineer



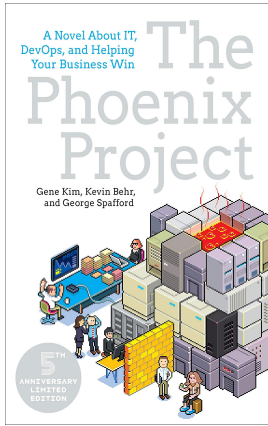
Eclipse Che Lead



CNCF Ambassador



Developer Efficiency



“For Phoenix, it takes us **three or four weeks** for new developers to get builds running on their machine...”

Anyone, anytime, can contribute to a project **without installing or configuring** software.



Eclipse Che

Virtual Machines?



Containers?



Eclipse Che

Environment managers?



Configuration Managers?



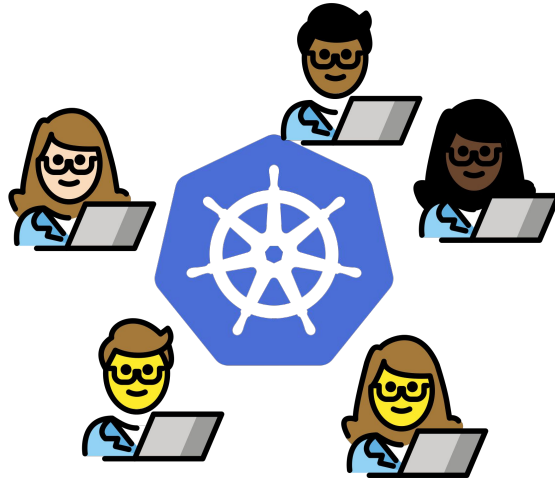
Pros

Using Kubernetes as a CDE Platform

Vendor Neutral and Standard

Community

Scalability



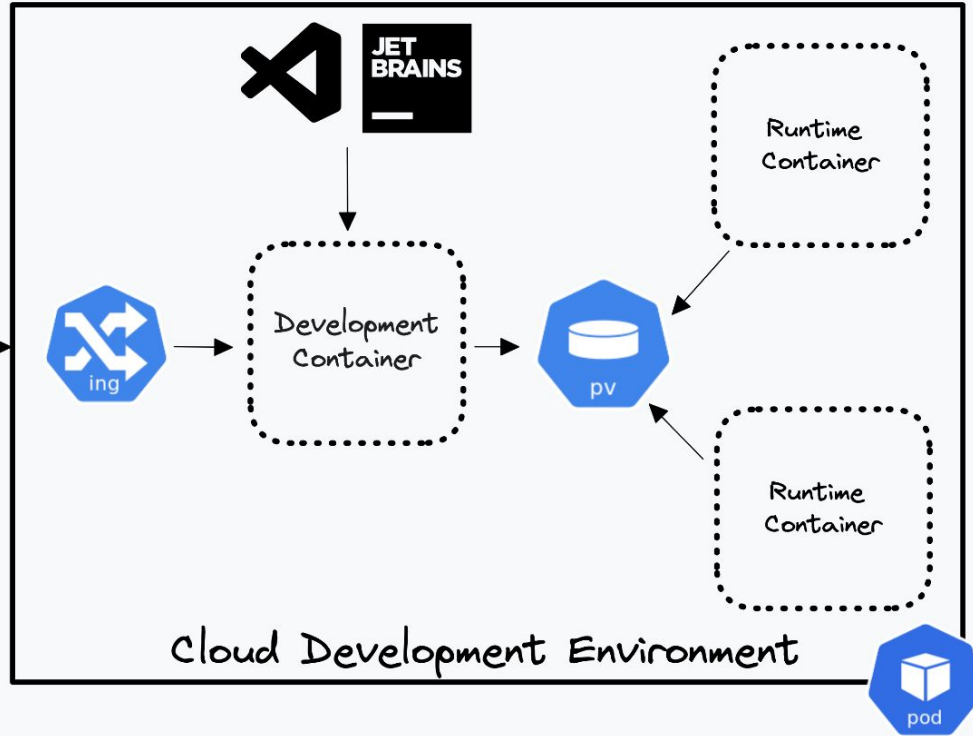
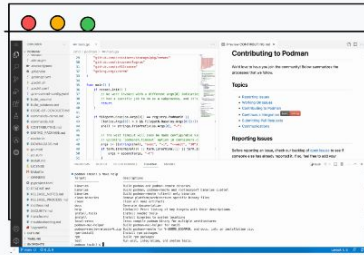
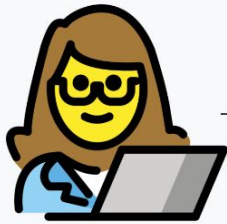
Enterprise

Resource efficiency

Extensibility

Repeatable Dev environments

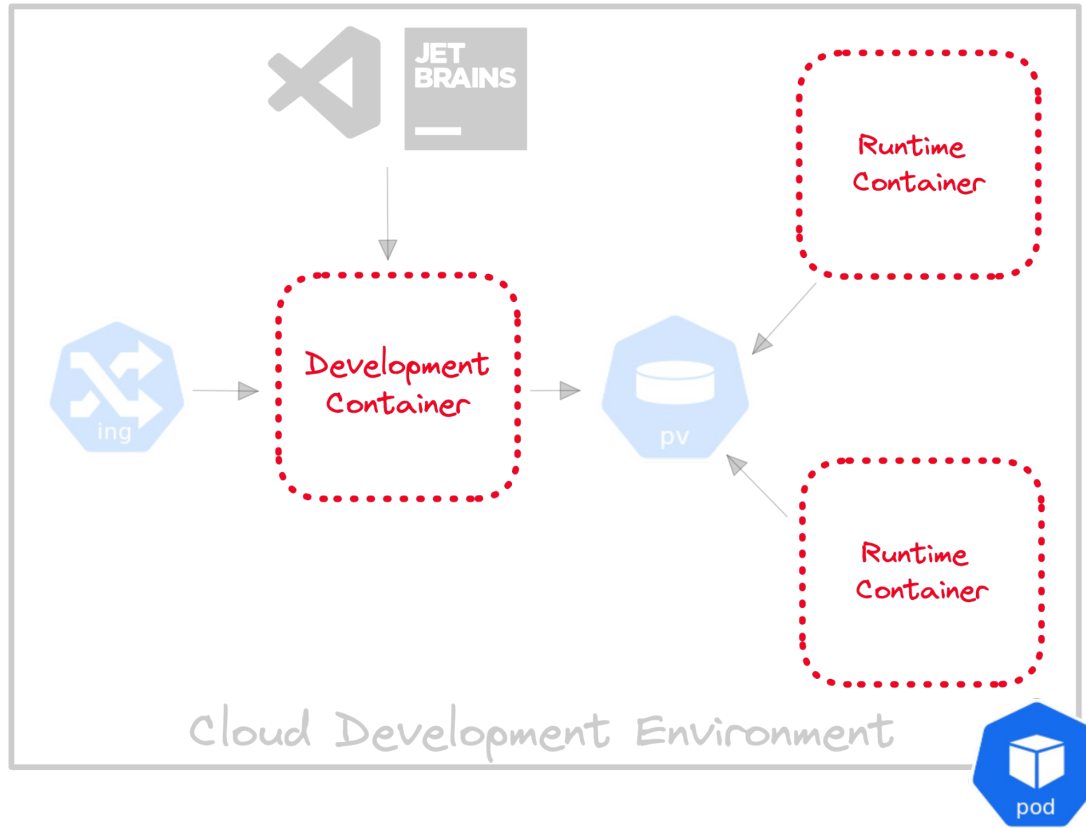
Eclipse Che Architecture Workspace



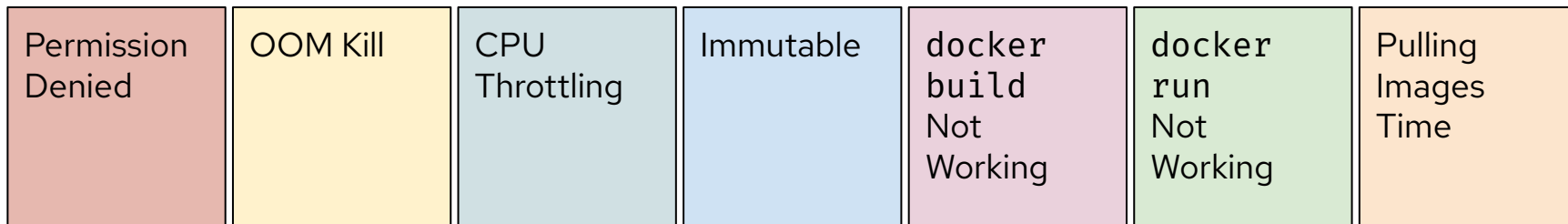
Cons

The issues we faced building Eclipse Che on top of Kubernetes

Running a CDE in a Kubernetes Pod



Running a CDE in a Kubernetes Pod

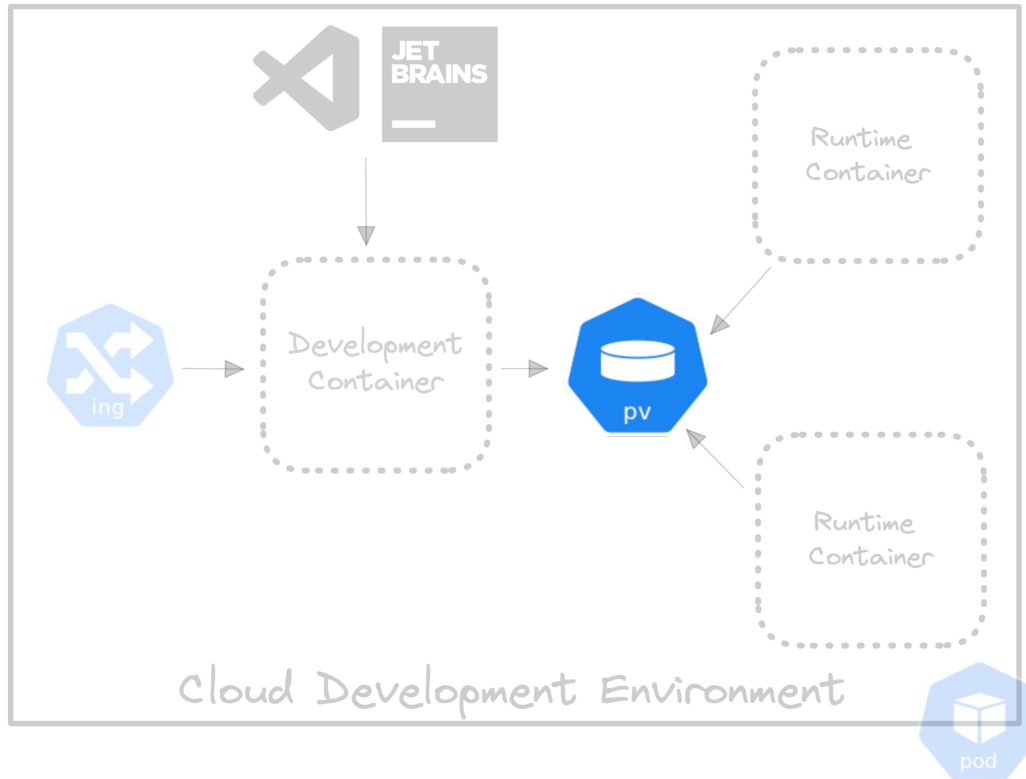


Running a CDE in a Kubernetes Pod

Permission Denied	OOM Kill	CPU Throttling	Immutable	docker build Not Working	docker run Not Working	Pulling Images Time
Build images with \$HOME R/W for arbitrary unprivileged user	Specify correct Memory Limit	Specify correct CPU Limit (or don't at all!)	Install packages at build time. Use of a universal developer image.	Use Rootless build and the right Pods Security Context	kubedock	Image Pre-Pulling



Persistence Volumes: Network Attached Storage



Persistence Volumes: Network Attached Storage

Write Errors

Exclusive Access
Mode

Mount Timeout

Quotas

Persistence Volumes: Network Attached Storage

Write Errors

Use Block Storage Only

Exclusive Access Mode

One Persistent Volume per workspace

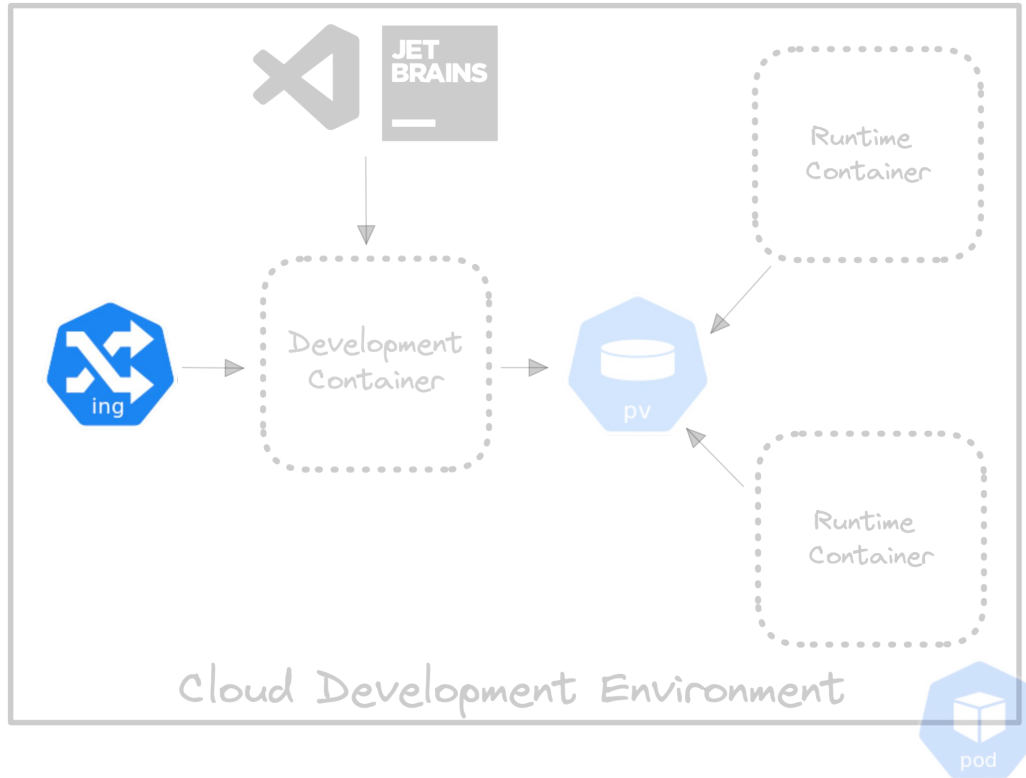
Mount Timeout

Ephemeral / Asynchronous Storage (experimental feature)

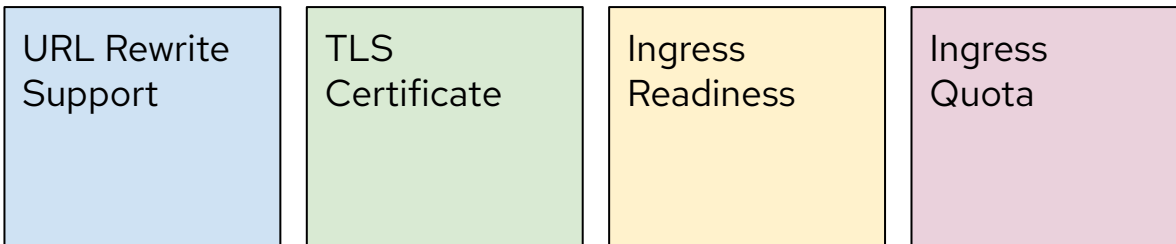
Quotas

Fail Fast / Explicit Error Message

Networking: running behind a reverse proxy



Networking: running behind a reverse proxy



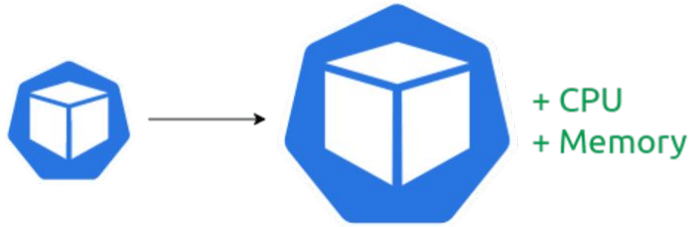
Networking: running behind a reverse proxy

URL Rewrite Support	TLS Certificate	Ingress Readiness	Ingress Quota
Backend services have an "external" and an "internal" URL	Use one unique external domain	Use a Threshold of number of success to consider a service available	Delete Ingresses at workspace stop

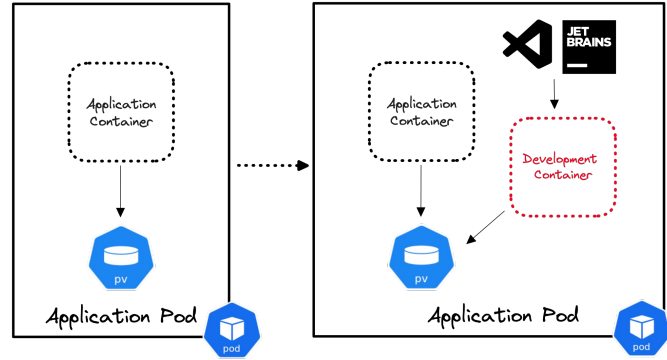
New Opportunities

Leveraging the latest Kubernetes features for CDE purposes

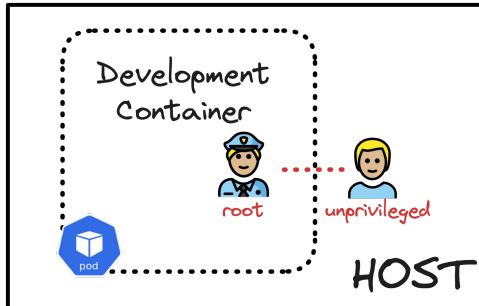
Autoscalers



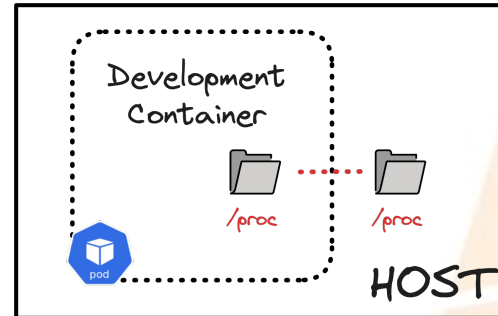
Ephemeral Containers



User Namespace

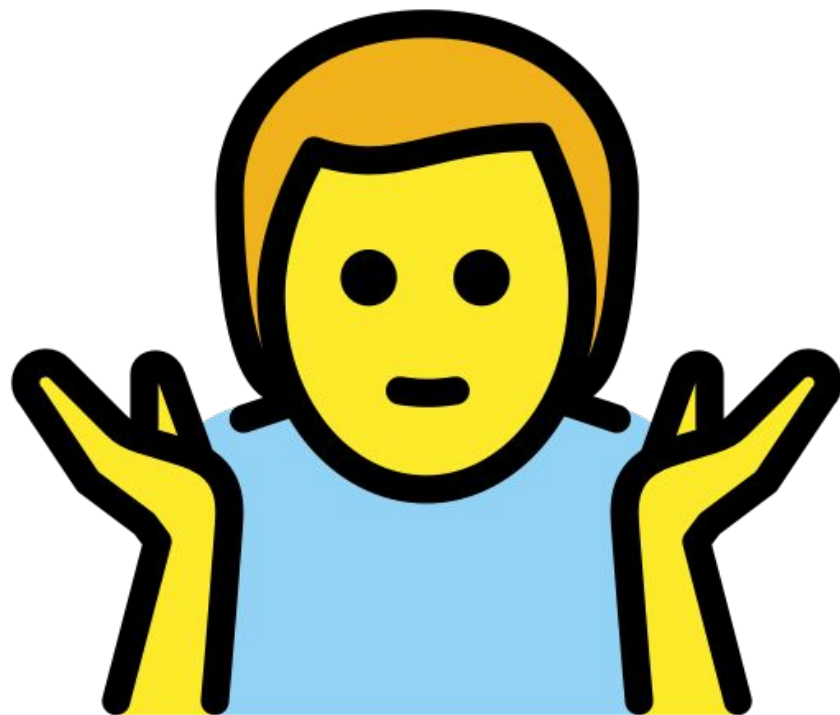


Proc Mount



Conclusion

Is it worth it?



Cons	Pros
Starting a container-based CDEs is not faster than VM-based CDEs	CDEs are immutable and software defined
Developers are not used to code and build in security hardened development environments. Some things won't work.	Administrators can effectively enforce development environments enterprise policies
CDE customization is not straightforward	When Kubernetes is the application target platform, the CDE makes it easy to test it and debug it there
Current IDE technologies are not designed to run in the cloud	Kubernetes is evolving rapidly with plenty of new features and opportunities at every release

Thank you!

Join the conversation:

 [@EclipseCon](https://twitter.com/EclipseCon) | [#EclipseCon](https://twitter.com/EclipseCon)

