
KubeCon
CloudNativeCon
North America 2024

KubeCon | CloudNativeCon

North America 2024

Running WebAssembly Workloads Side-by-Side with Container Workloads

Jiaxiao (Joe) Zhou, Microsoft

Jiaxiao (Joe) Zhou

- Software Engineer at Microsoft's Azure, building open-source software
- Maintainer of the CNCF Containerd Runwasi and SpinKube
- Recognized contributor of the Bytecode Alliance

𝕏: x.com/jiaxiao_zhou

in: linkedin.com/in/mossaka

: github.com/mossaka

Where we are going today

- Sidecars and Use Cases
- What is WebAssembly
- WebAssembly vs. Containers
- Running WebAssembly with Sidecars
- Running WebAssembly as Sidecars
- The Future
- Questions

Kubernetes Sidecar Pattern

- Deploy an auxiliary container alongside the main application
- Share the **same namespaces** and **cgroups**
- Provide logging, monitoring, networking, configuration management, state management etc.
- Kubernetes **Sidecar Containers** feature in v1.29 [beta]

Sidecar Use Cases

State and Communication

Service Mesh

Logging

Kubernetes Sidecar Pattern

- Sidecars could be **heavy-weight**
- Sidecar containers consume additional CPU, memory and network resources
- **Operational complexity** in deployment and management
- Impact on Pod scaling and cluster efficiency

WebAssembly

webassembly.org

- A **portable** compilation target
- **Near-native** speed
- **Sandboxed**
- **Lightweight**
- Supported by many major programming languages
- No web-specific assumptions

WASI: WebAssembly System Interface

wasi.dev

- WASI is a standard set of APIs to interact with any host
- Brings WebAssembly use cases outside the browser
- Designed with similar principles of WebAssembly in mind: safe, portable, efficient
- WASIp1 has wide adoptions among languages and toolings
- WASIp2 is based on WebAssembly Component Model
  - Released in Jan, 2024
  - Two standardized “Worlds” to target to: `wasi-cli` + `wasi-http`

Containers

- **Benefits**
  - Production tested over decades
  - Native speeds
  - Broad ecosystem support with standard toolings like OCI and Kubernetes
- **Downsides**
  - Containers can often be hundreds of MBs in size
  - Code startup is often not fast enough for many use cases
  - Containers built per architecture
  - Inter-container communication has a lot of overhead

WebAssembly

- **Benefits**
  - Sub-milliseconds cold starts
  - Fast inter-wasm communication
  - High density of guest applications
- **Downsides**
  - Linux libraries may not compile
  - Language toolings are not great
  - Relatively new technology

WebAssembly in Kubernetes?

Distributing WebAssembly

tag-runtime.cncf.io/wgs/wasm/charter

- Like containers, WebAssembly can be stored and retrieved from OCI registries as **OCI Artifacts**.
- WebAssembly OCI Artifact Format

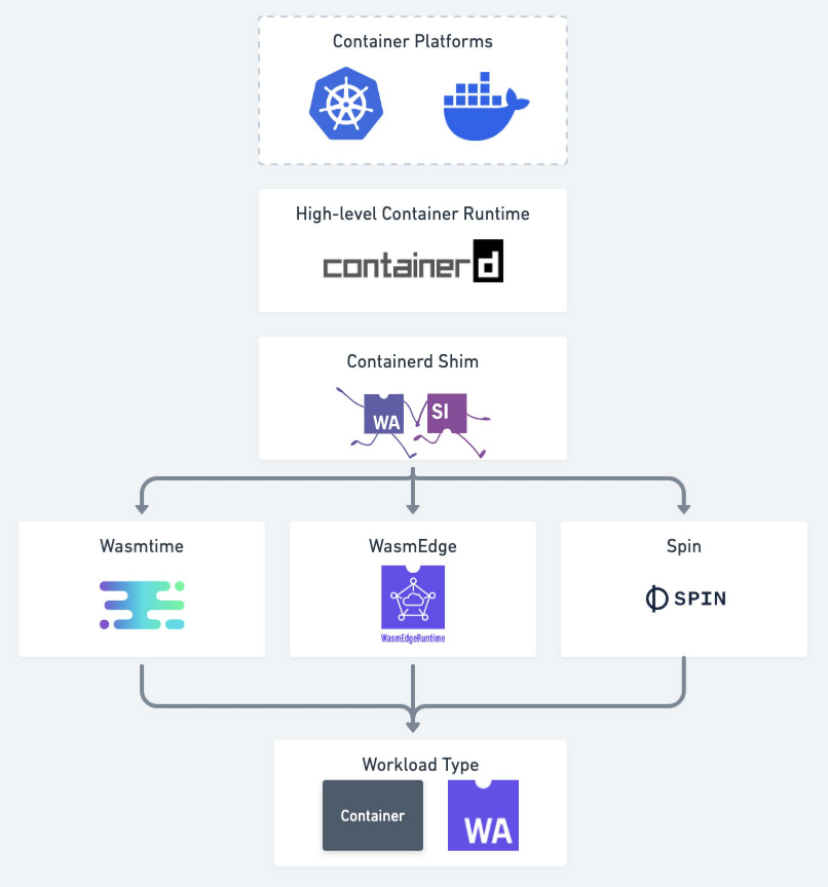
```
Config:
  Digest:  sha256:66305959b88c33eb660c78bed6e9e06ec809a38f06f89a9ddf5b0cb8b22f0c0c
  MediaType: application/vnd.wasm.config.v0+json
  Size:    413B

Layers:
  Digest:  sha256:a31c2628694eb560dd0e8f82de12e657268c761727c3ad98638c9c55dd46c5df
  MediaType: application/wasm
  Size:    87818B
```

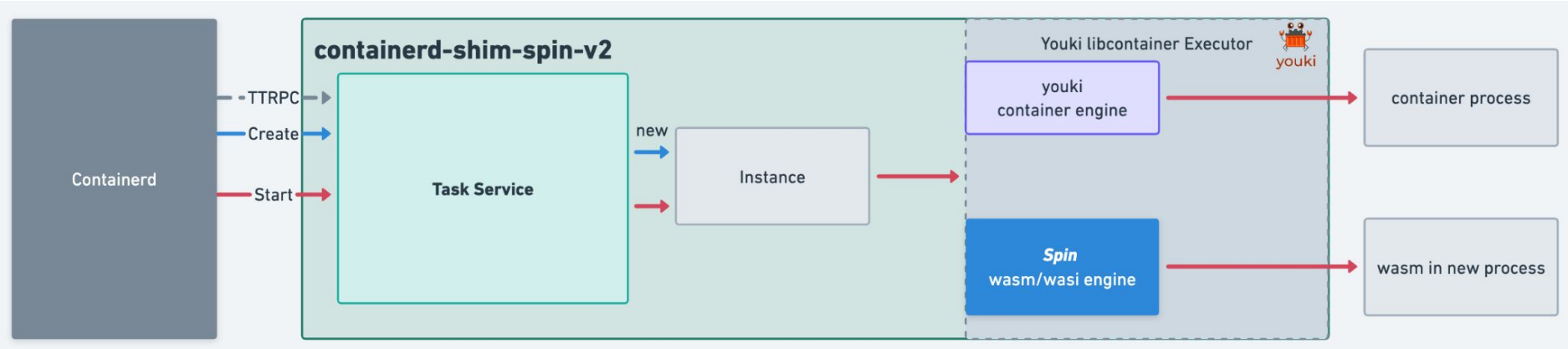
Running WebAssembly

github.com/containerd/runwasi

- Facilitates WebAssembly workloads in Containerd
- Support multiple WebAssembly / WebAssembly System Interface (WASI) runtimes
- Can run WebAssembly side-by-side with Containers



Runwasi Architecture



Workloads

github.com/spinkube/containerd-shim-spin

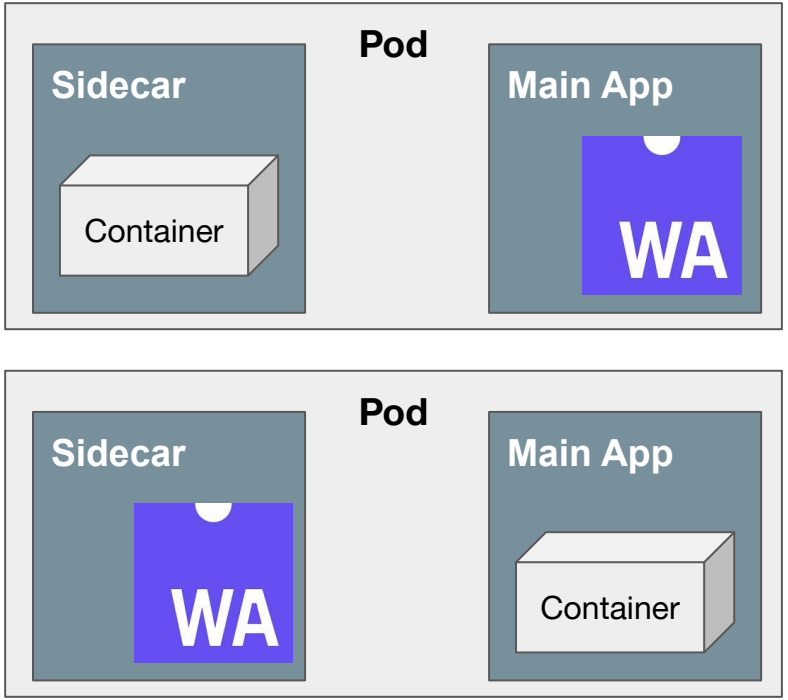
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: wasm-spin
labels:
  app: wasm-spin
spec:
  runtimeClassName: wasmtime-spin-v2
  containers:
    - name: spin-hello
```

```yaml
apiVersion:
node.k8s.io/v1
kind: RuntimeClass
metadata:
name: wasmtime-spin-v2
handler: spin
```

```
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.spin]
runtime_type = "io.containerd.spin.v2"
```

Running WebAssembly Side-by-Side with Containers, Why?

See it in Action!



1. A drop-in replacement for Linux containers while still integrating with familiar tools (e.g., Istio, Dapr, or OpenTelemetry collector)
2. Run WebAssembly applications as sidecars in your cluster
3. ???

Running Linux Sidecars with Wasm

- A Linux container running a Envoy proxy can act as a sidecar for your Wasm application, handling complex networking tasks such as routing and load-balancing
- Sidecars can manage stateful storage and caching (e.g. Dapr)
- Proprietary code that cannot be compiled to Wasm can run as a sidecar container

Demo!

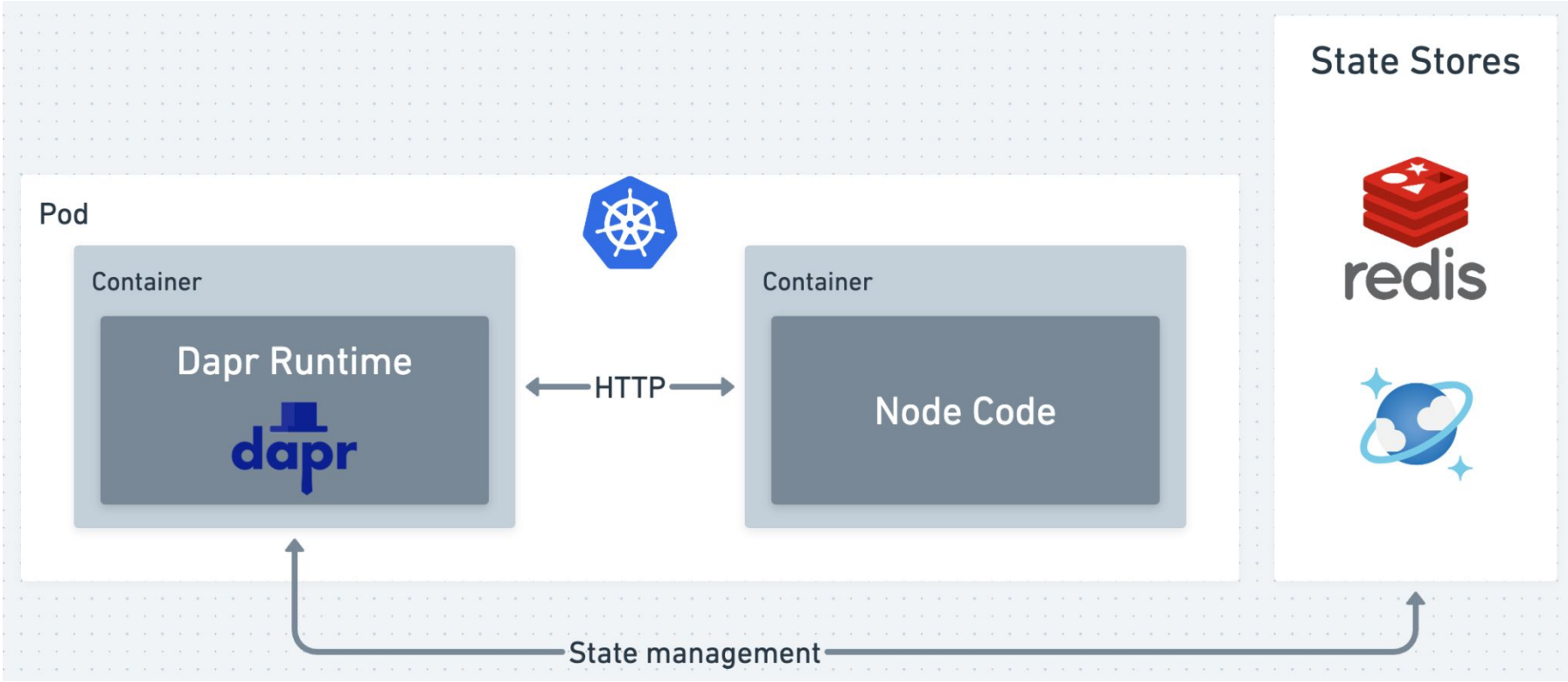
github.com/keithmattix/istio-wasm-demo

WebAssembly as Sidecars

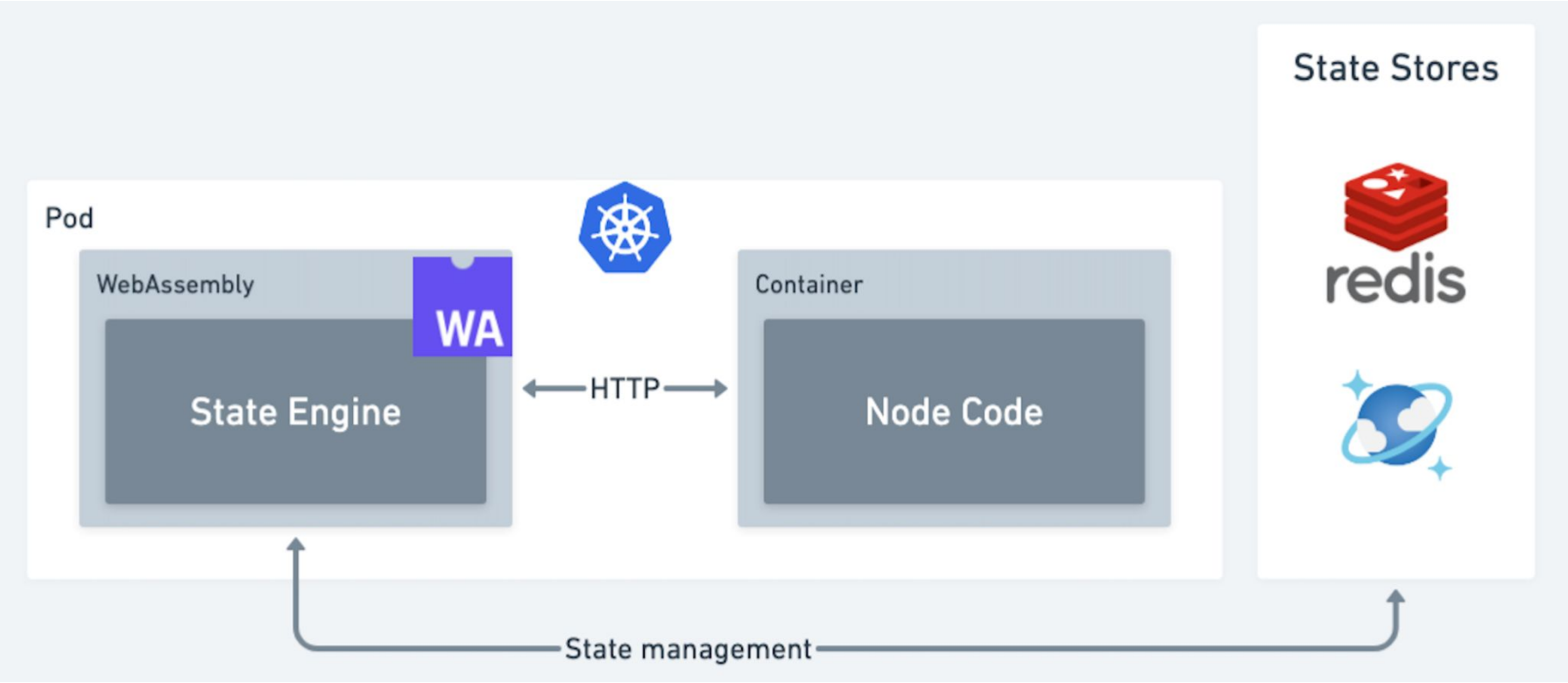
Running Wasm as Sidecar Containers

- Significantly reduce binary size
- Wasm instance is per request, and has no cost if there are no traffic
- Wasm is well suited for filtering, validating or transformation data before requests reach the main application.

Demo


State Stores
redis
Pod
Container
Dapr Runtime
dapr
HTTP
Container
Node Code
State management

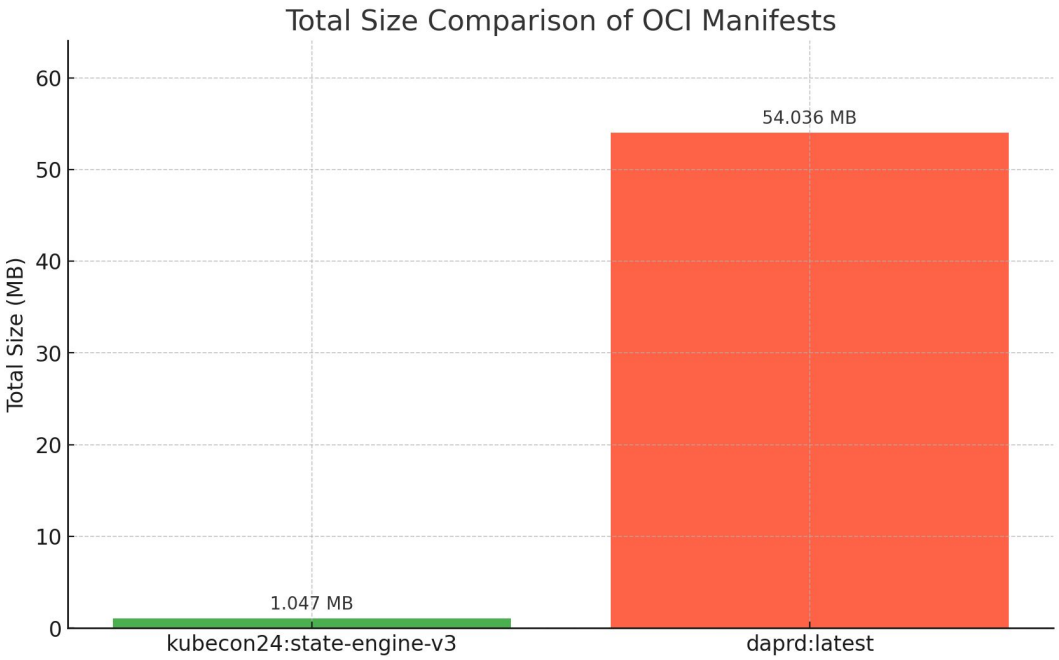
Demo


State Stores
redis
Pod
WebAssembly
WA
State Engine
HTTP
Container
Node Code
State management

DEMO!

github.com/Mossaka/state-engine

Wasm OCI Artifact Size Comparison



*Note: in this experiment, I found no significant difference in terms of CPU and memory usage

Can we do better?

WebAssembly Component Model

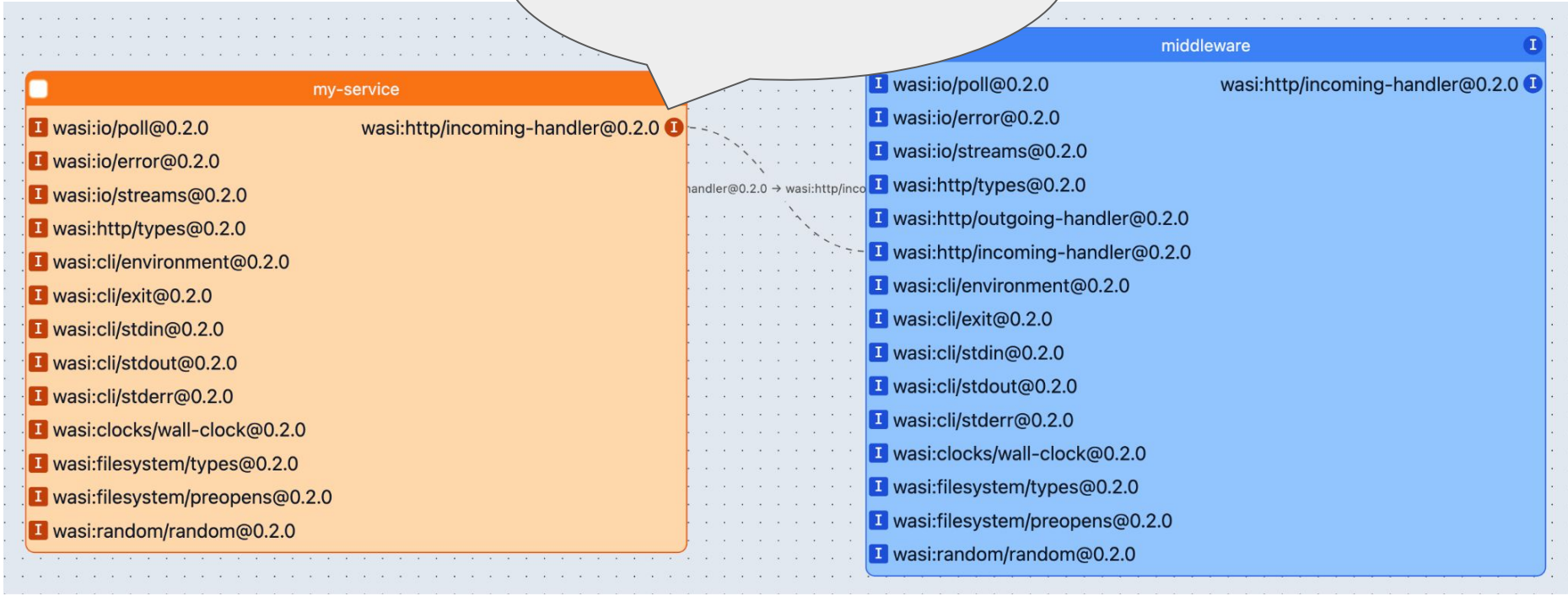
component-model.bytecodealliance.org

- ABI and IDL (WIT) for composing WebAssembly modules
- High-level types:
  - `string`, `record` (struct), `variant`, `option`, and `result`
- Worlds: a contract between the guest and the host
- **WASIp3**: you can truly compose wasi-http handlers

WebAssembly Component Composition

wasmbuilder.app

It will become
wasi:http/handler@0.3.0



DEMO!

github.com/radu-matei/spin-deps-image-manipulation

WebAssembly Components

1. Can be developed by many different programming languages
2. Inter-component communication is done by local function invocations, faster than sidecar communication
3. Synchronization
4. Good for resource consumption
5. Clear security boundaries
6. Noisy neighbor is manageable
7. Eliminate the needs for sidecar containers

WebAssembly Platforms

SpinKube

spinkube.dev

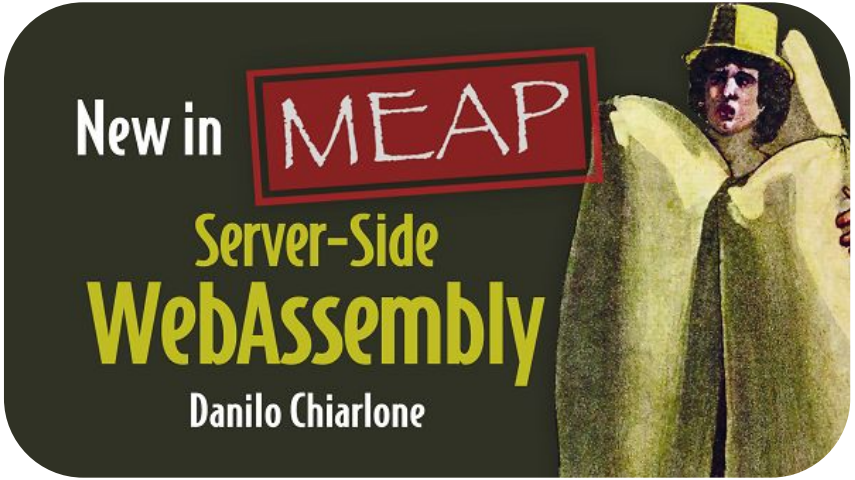
WasmCloud

wasmcloud.com

Conclusion

1. Sidecar Containers are a powerful Kubernetes Design Pattern to enhance application with additional capabilities
2. WebAssembly can take advantage of the sidecars to increase adoption and reduce container sizes
3. WebAssembly Components present an opportunity to eliminate sidecars entirely, while retaining the advantages of the sidecars, such as clear security boundary and polyglot

Check out this book!



45% off with code
chiarlone45

Questions?

Jiaxiao (Joe) Zhou

𝕏: x.com/jiaxiao_zhou

in: linkedin.com/in/mossaka

: github.com/mossaka