Juju - Google Go in a scalable Environment

Frank Müller / Oldenburg / Germany

Frank Müller

Born 1965 in Oldenburg

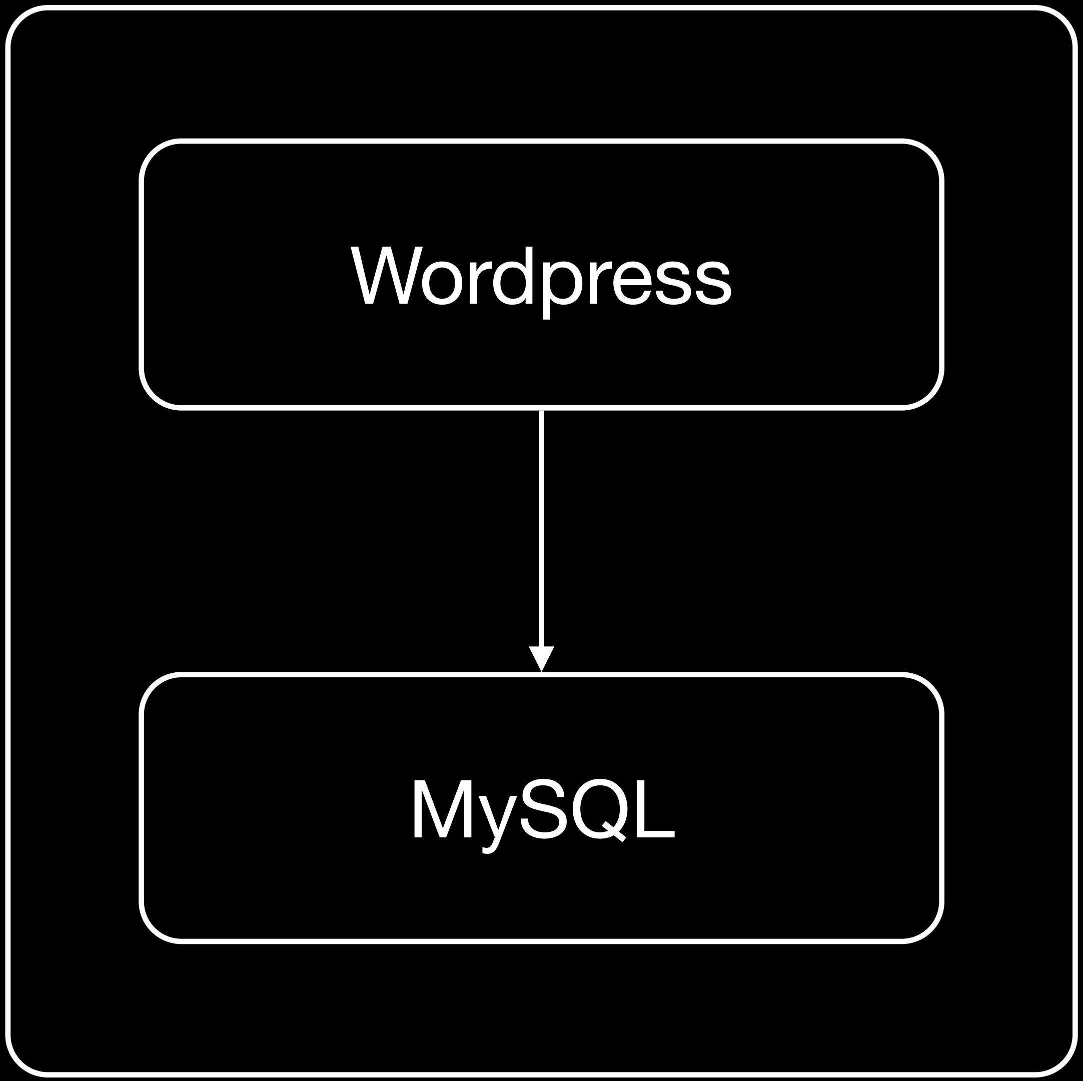
Software Development since 1984

Author since 1999

Book about Go in 2011

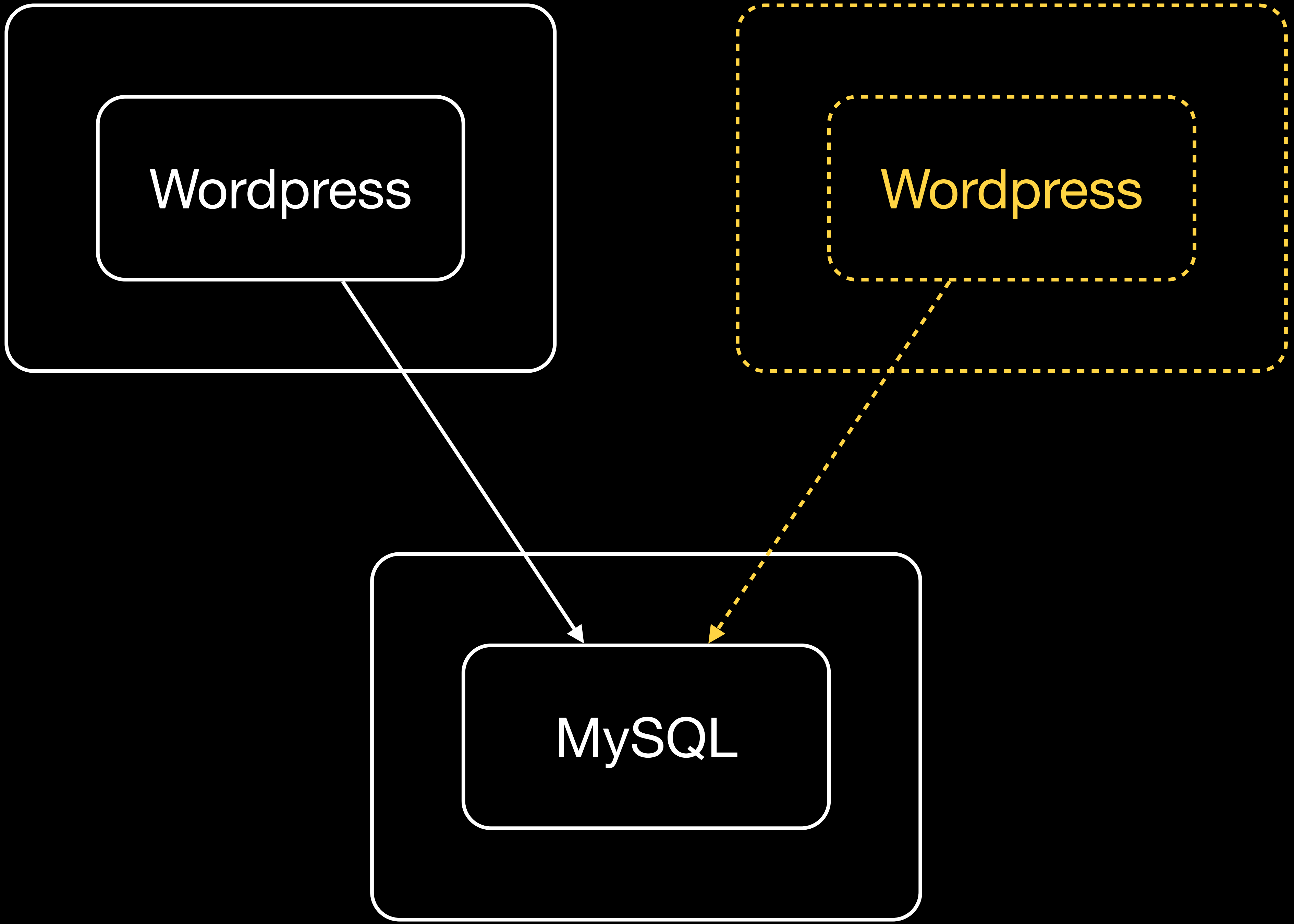
Introduction

Tideland



Well known scenario

Tideland



Different in clouds

Scaling means effort

Amazon Web Services

MAAS

Local

OpenStack

Microsoft Azure

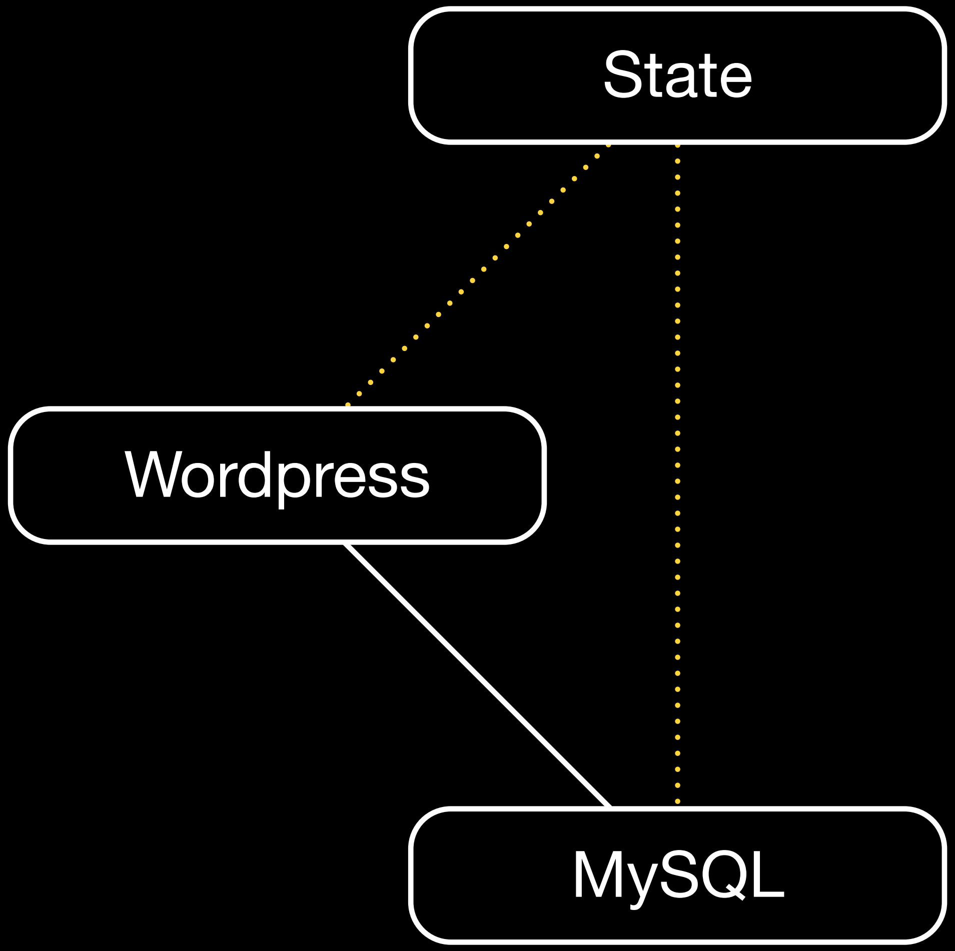
Juju for provisioning

Tideland

```
juju init -w
juju bootstrap
```

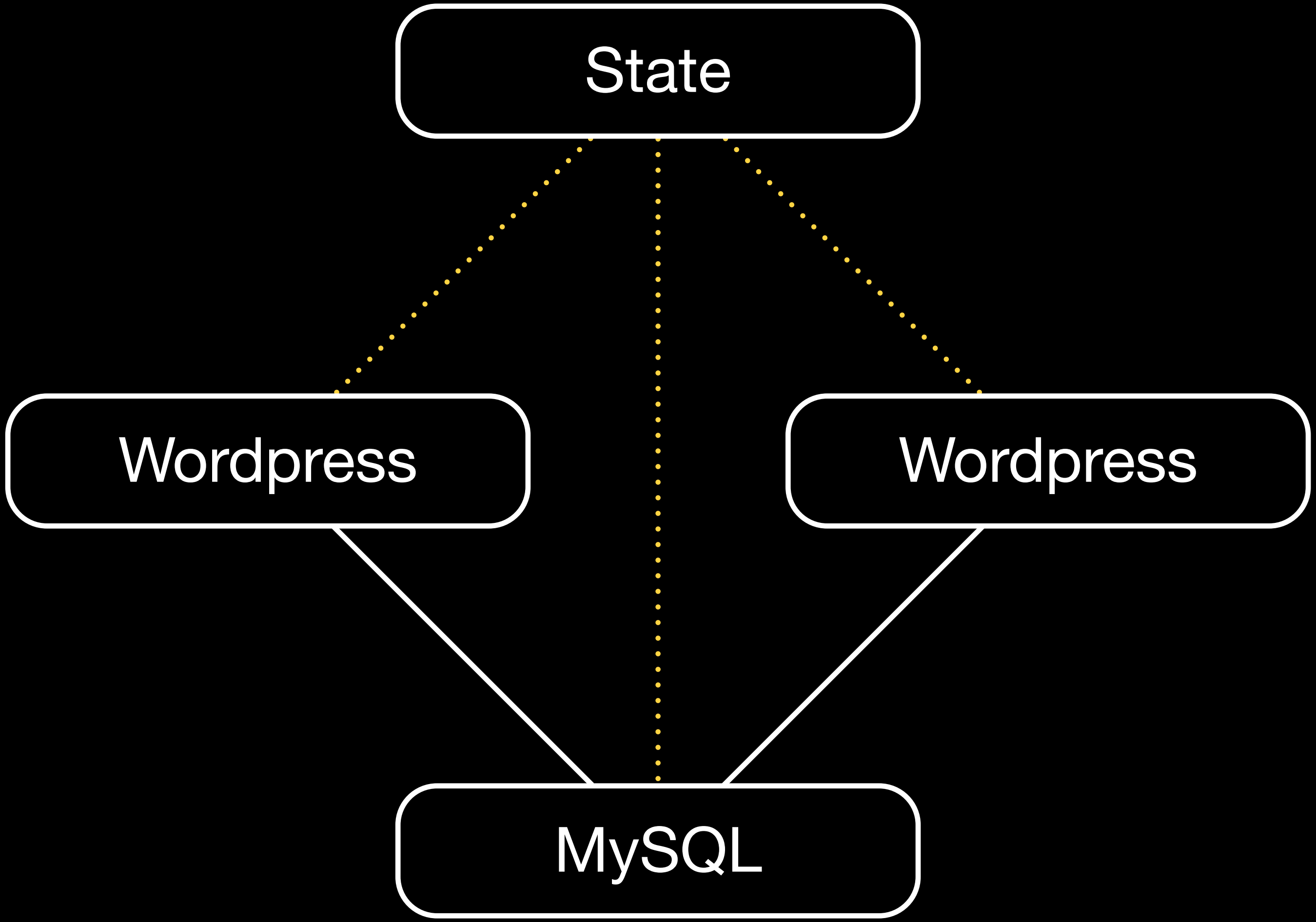
Create your environment

```
juju deploy cs:precise/wordpress
juju deploy cs:precise/mysql
juju add-relation wordpress mysql
```



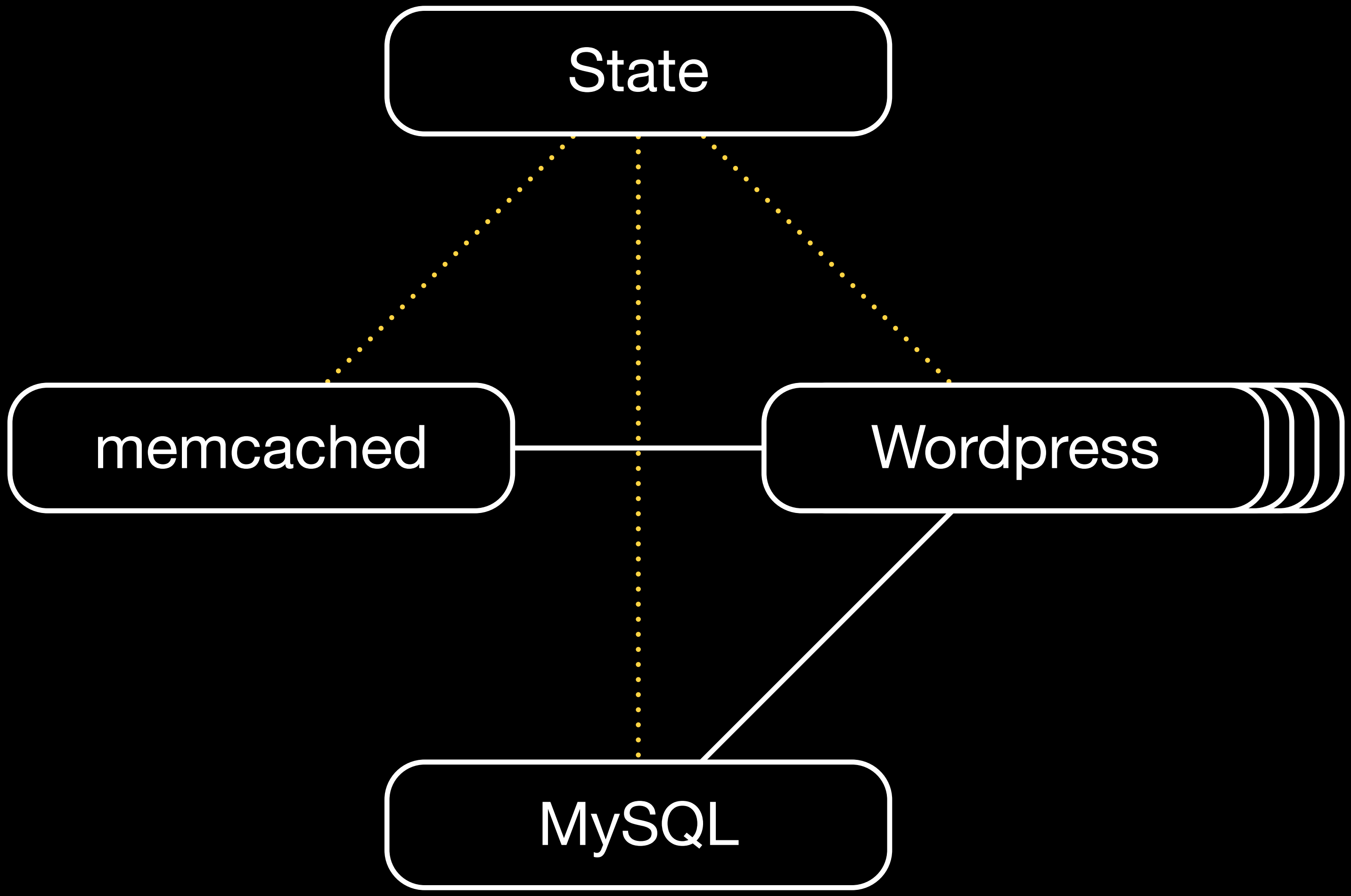
Add and relate services

```
juju add-unit wordpress
```



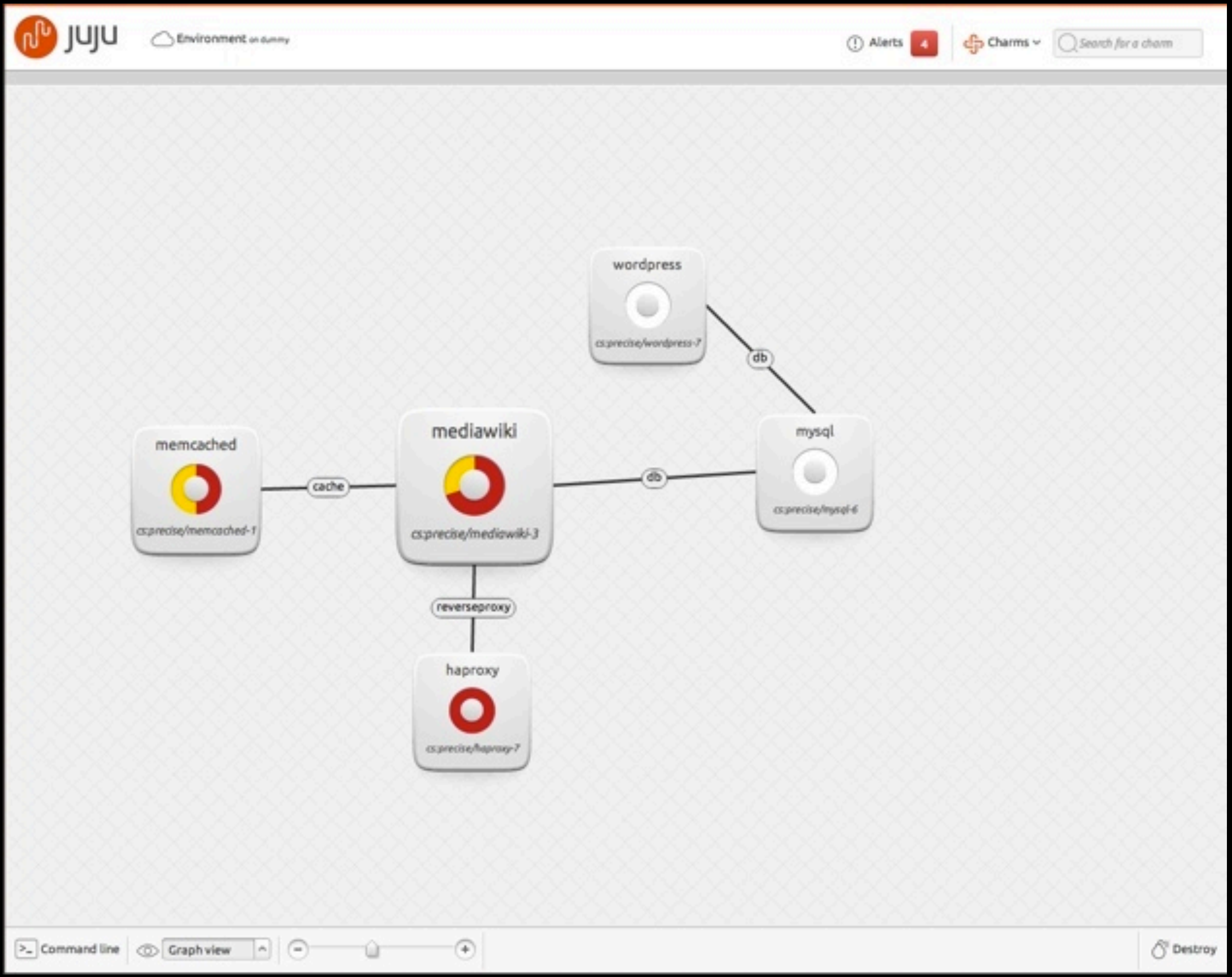
Add another unit

```
juju add-unit -n2 wordpress
juju deploy memcached
juju add-relation wordpress memcached
```



Scale!

Tideland



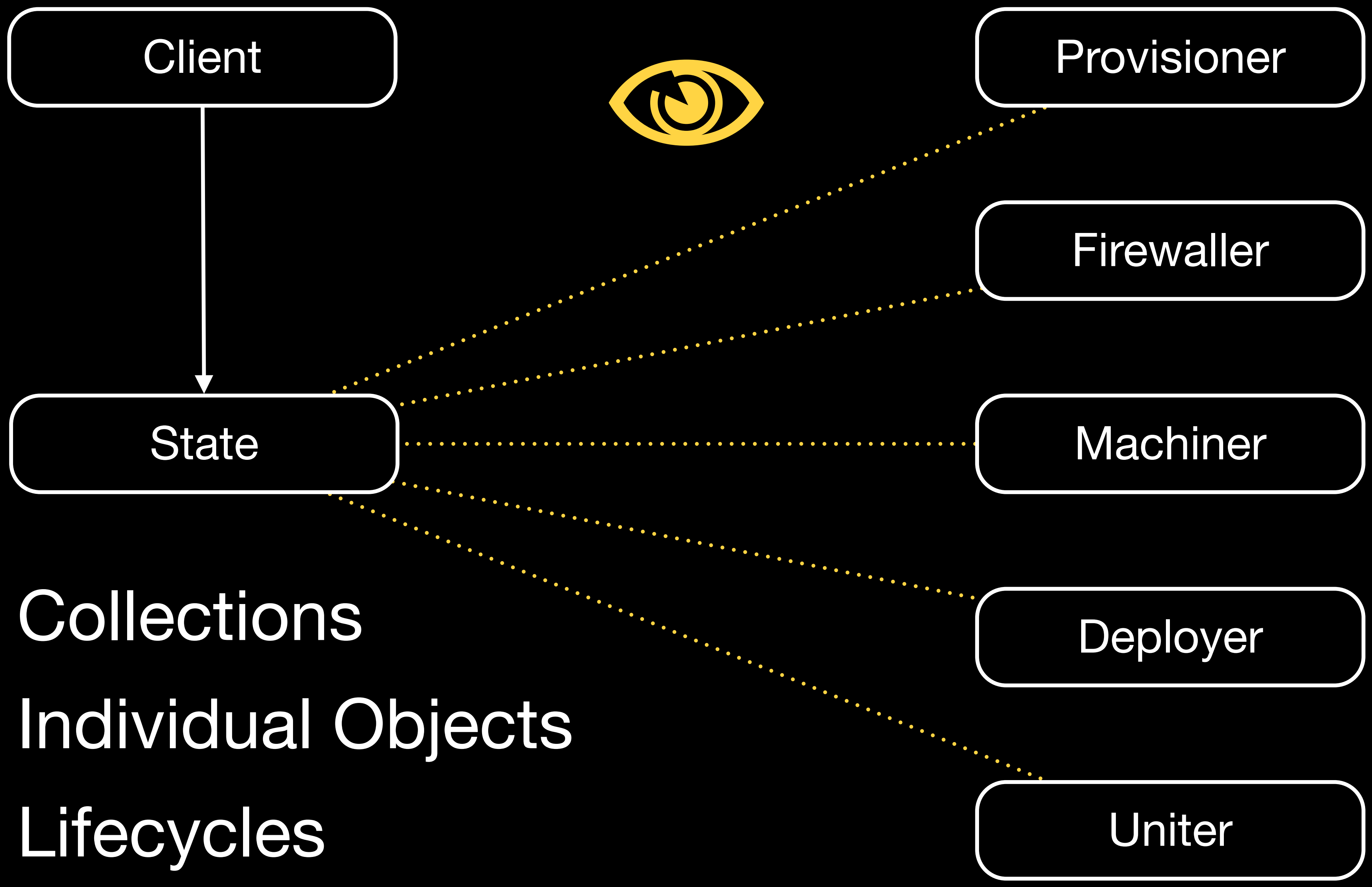
Also web UI available

Why Google Go?

Tideland

Architectural insights

Tideland



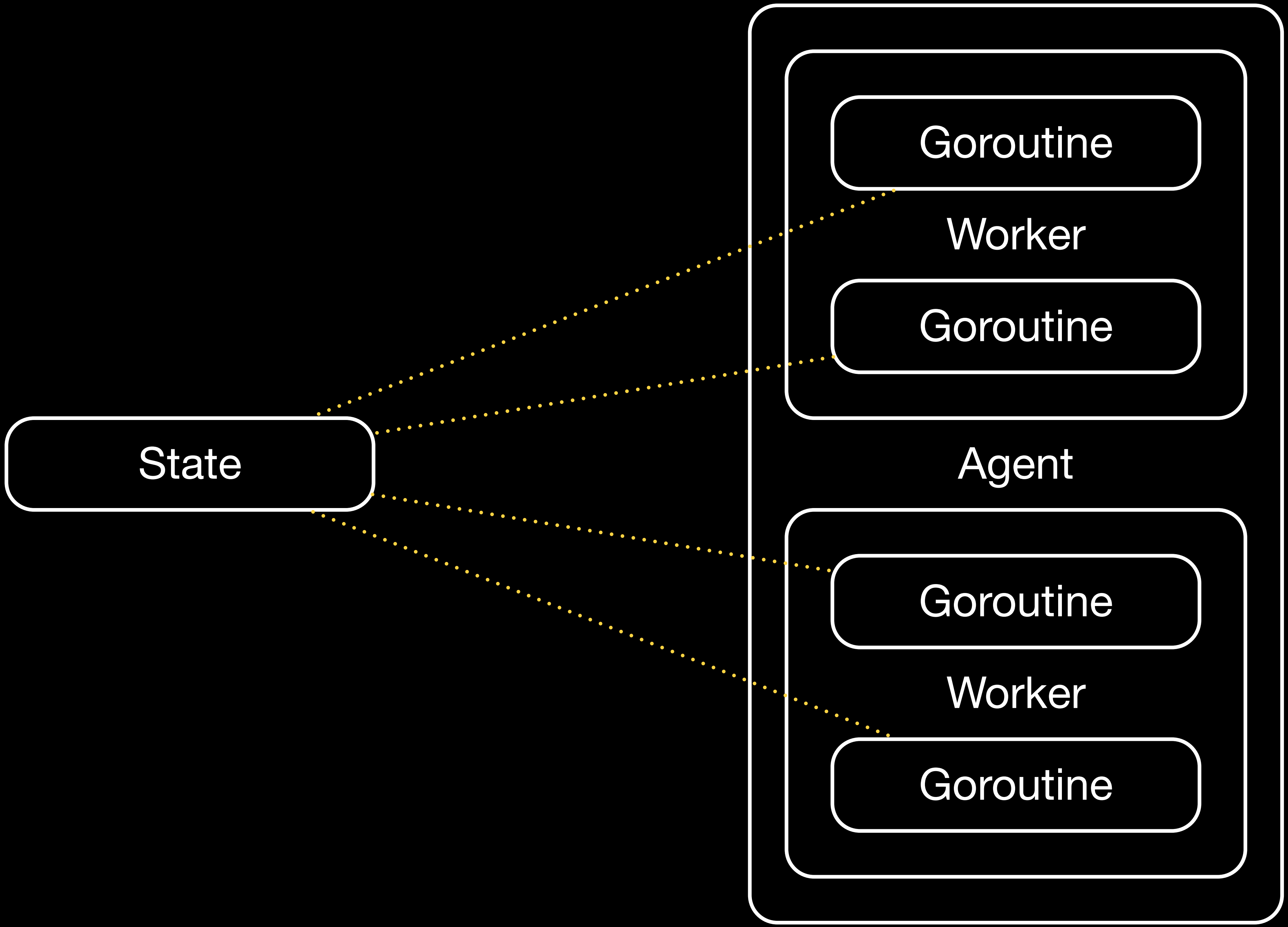
Collections

Individual Objects

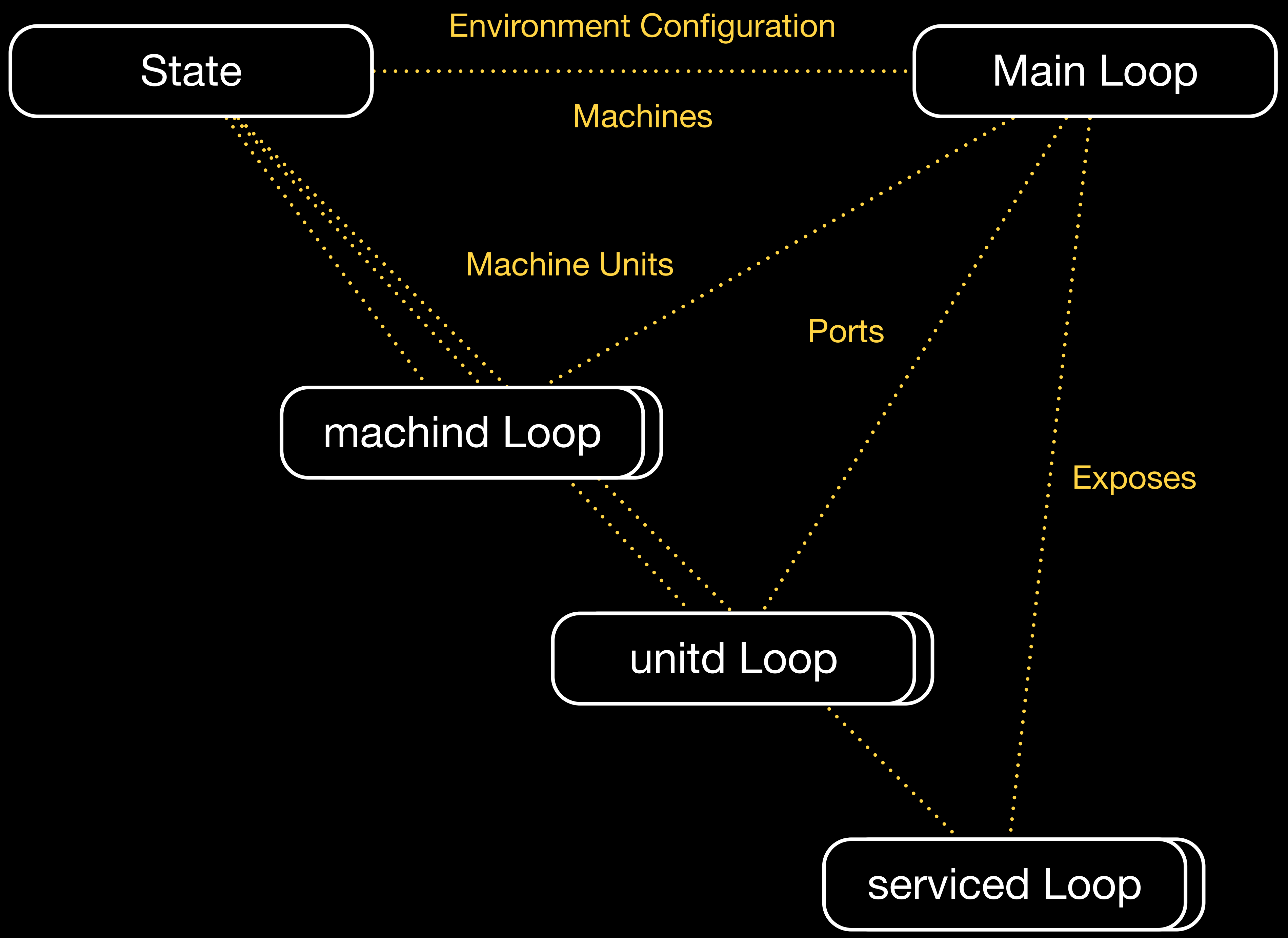
Lifecycles

State - Watcher - Worker

Tideland



Lots of concurrent work


State
Environment Configuration
Main Loop
Machines
Machine Units
Ports
machind Loop
Exposes
unitd Loop
serviced Loop

Firewaller

Goroutine control

- not part of the language spec
- launchpad.net/tomb
- signals to leave loops
- wait until goroutine stopped
- leave in case of an error
- remember and retrieve that error

Monitoring and stopping

```
// loop processes ...
func (t *T) loop() {
	defer t.tomb.Done()
	for {
		select {
		case <-t.tomb.Dying:
			// Cleanup ...
			return
		case f := <-t.fooChan:
			if err := t.foo(f); err != nil {
				t.tomb.Kill(err)
			}
		case b := <-t.barChan:
			// ...
		}
	}
}
```

Loops with tomb

```
// Stop ends the main loop.
func (t *T) Stop() error {
	t.tomb.Kill(nil)
	return t.tomb.Wait()
}

// Err retrieves the error in case the backend loop died.
func (t *T) Err() error {
	return t.tomb.Err()
}
```

Stop and error handling

Tideland

Interfaces

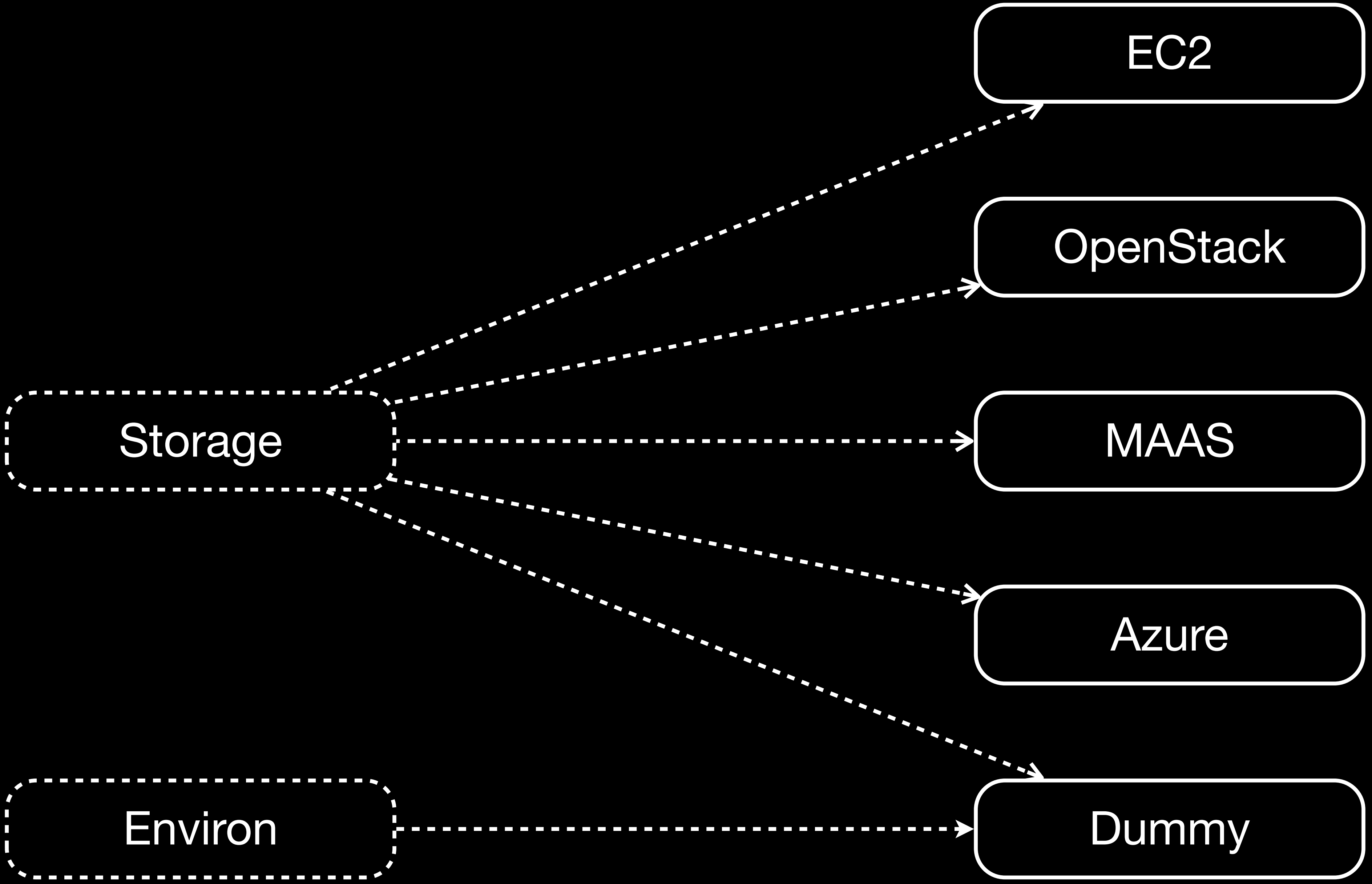
Tideland

```
type StorageReader interface {
	Get(name string) (io.ReadCloser, error)
	List(prefix string) ([]string, error)
	URL(name) (string, error)
}

type StorageWriter interface {
	Put(name string, r io.Reader, length int64) error
	Remove(name string) error
}

type Storage interface {
	StorageReader
	StorageWriter
}
```

Define behaviors



Like a toolbox

```
type Foo interface {
	DoThis(with That) error
}

type MyFoo struct { … }

func (m *MyFoo) DoThis(with That) error { … }

type MockFoo struct { … }

func (m *MockFoo) DoThis(with That) error { … }

func Bar(f Foo, t That) error {
	return f.DoThis(t)
}
```

Also help in tests

Tideland

- extreme fast builds
- cross-compilation
- binaries are simple to deploy
- table-driven tests
- benchmarks and race detection
- go get always takes tip!

What else?

Questions?

Tideland