This job is too hard.

Brendan Burns – @brendandburns

Distinguished Engineer – Microsoft Azure

Co-Founder Kubernetes

Microsoft

Azure

The job.

- Build new features?
- Weekly releases?
- Perfect uptime?
- Remain buzzword compliant?

Microsoft

Azure

The tools.

- Patterns and practices.
- Infrastructure support.
- Code re-use.
- Hacker news.

Microsoft

Azure

The approach.

- Learn everything.
- Build everything.
- Explain everything.
- Monitor everything.
- Rinse and repeat.

Microsoft

Azure

We can’t keep doing this.

Microsoft

Azure

Fortunately, history repeats

Microsoft

Azure

What happens when the job gets to hard?

Microsoft

Azure

We build tools.

Microsoft

Azure

We find patterns

Microsoft

Azure

Historical reference points

- Machine code - 1940s
  - Fortran - 1956
  - The art of computer programming - 1968

Microsoft

Azure

Historical reference points

- Software Scaling - 1980s
  - Java/C# - 1990s
  - Design Patterns - 1995

Microsoft

Azure

Historical reference points

- Reliable Distributed Systems - 2010s
  - Docker & Kubernetes – 2013-14
  - ????

Microsoft

Azure

Solutions: Tools

- Examples:
  - Compilers
  - Debuggers
  - Unit testing
  - Integrated Development Environments
  - …

Microsoft

Azure

Solutions: Tools

- Helm (https://helm.sh)
- Kubernetes Extension for Visual Studio Code
- Draft (https://draft.sh)
- Telepresence (https://telepresence.io)
- Jenkins X (https://github.com/jenkins-x)
- Spinnaker (https://spinnaker.io)
- …

Microsoft

Azure

Demo: Visual Studio Code for Kubernetes

Microsoft

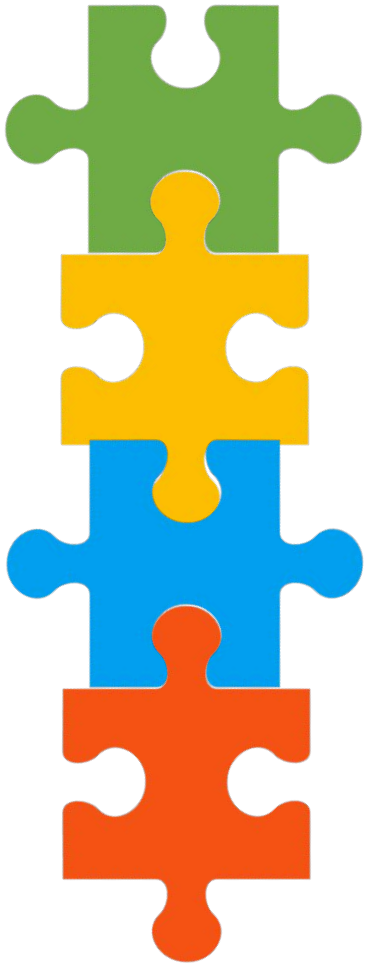
Azure

Demo: Draft (https://github.com/Azure/draft)

Microsoft

Azure

Cluster Services

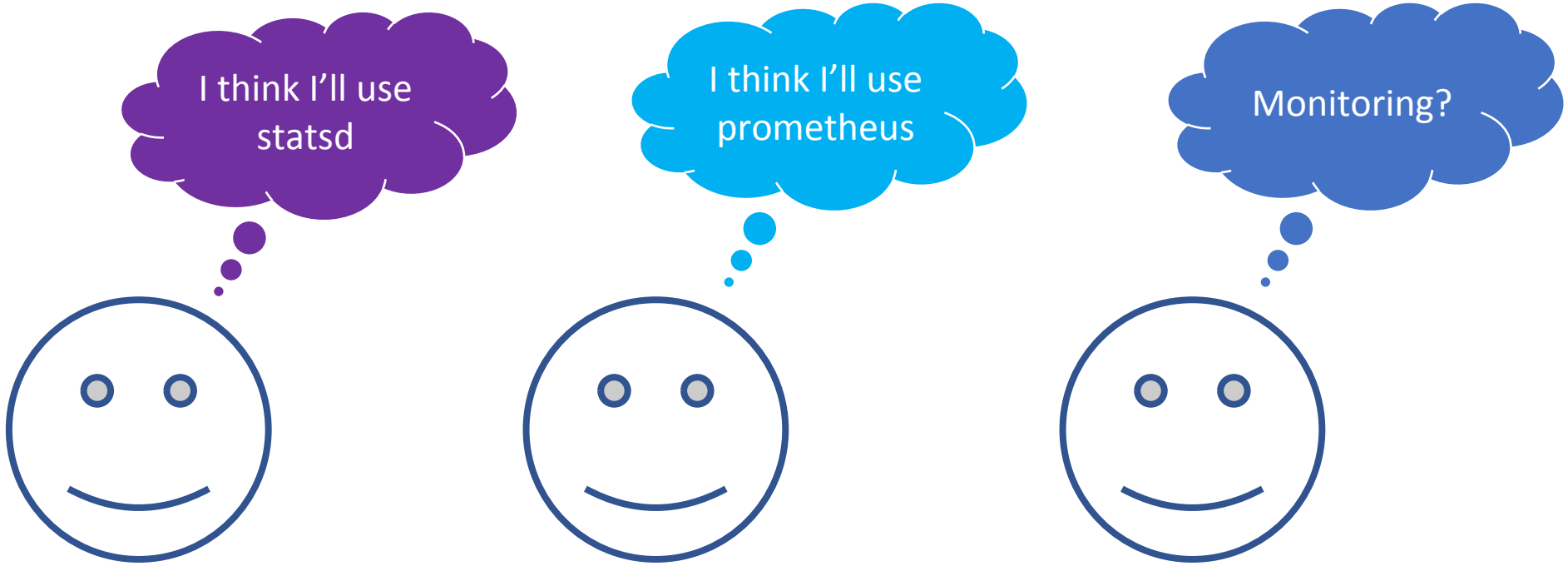


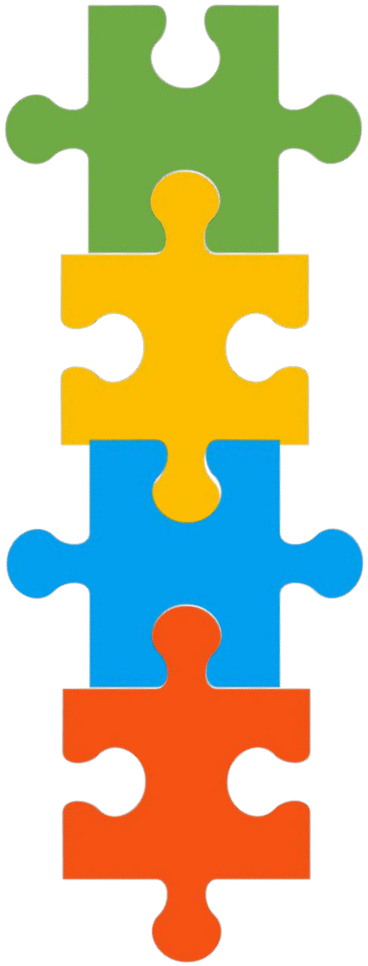
- Cluster wide standardization of common services
  - e.g Logging, Monitoring, Security
- Automatically enabled by ***deploying into the cluster***
- Ensure consistency, concentration of skills & best practices

Microsoft

Azure

Cluster Services

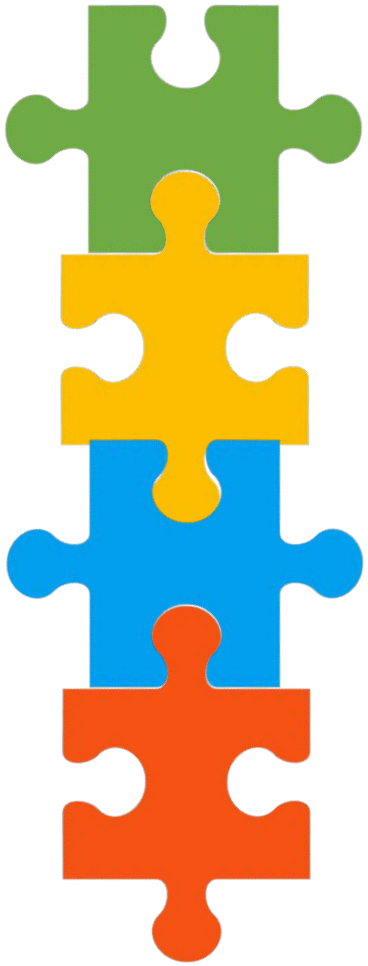




Microsoft

Azure

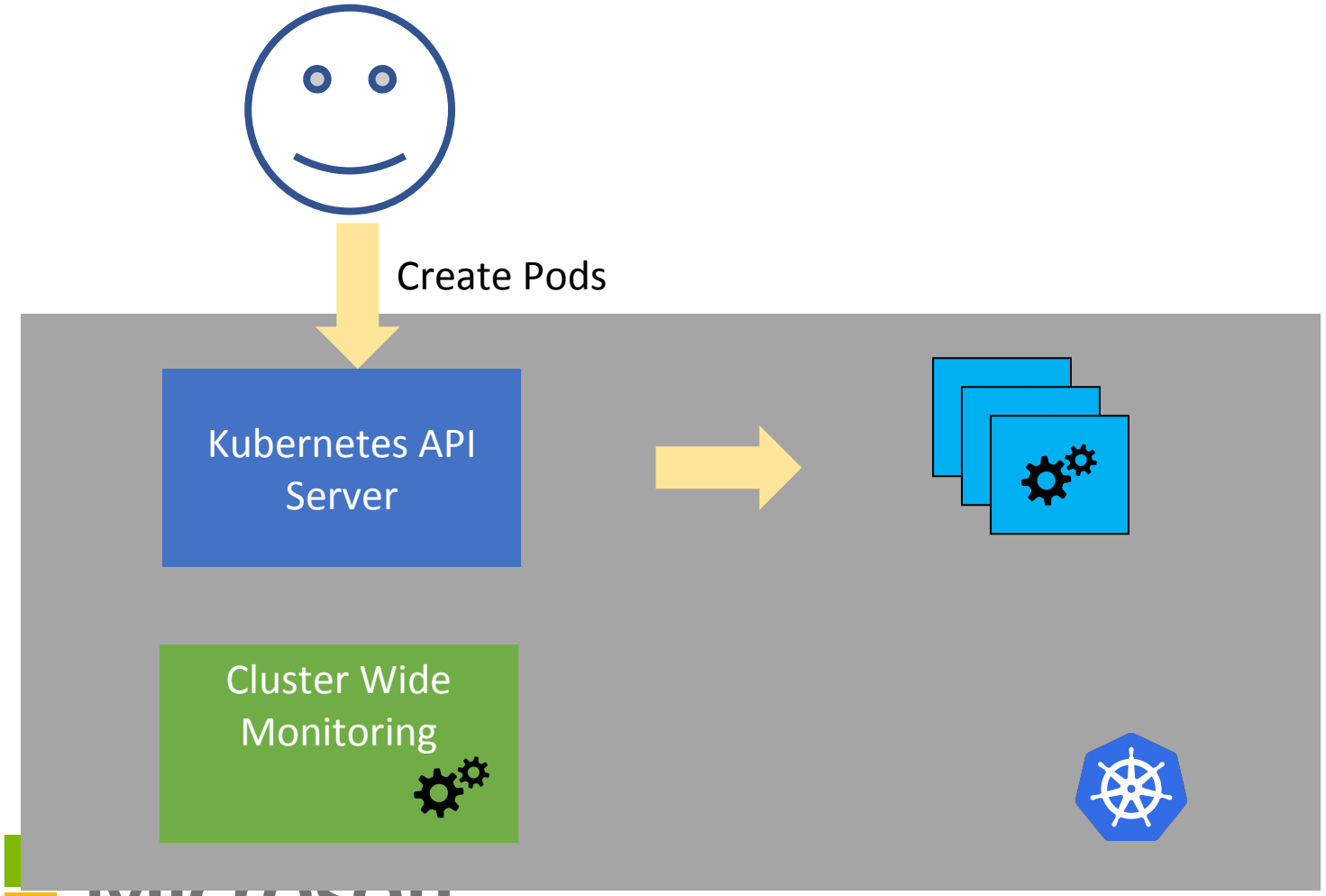
Cluster Services

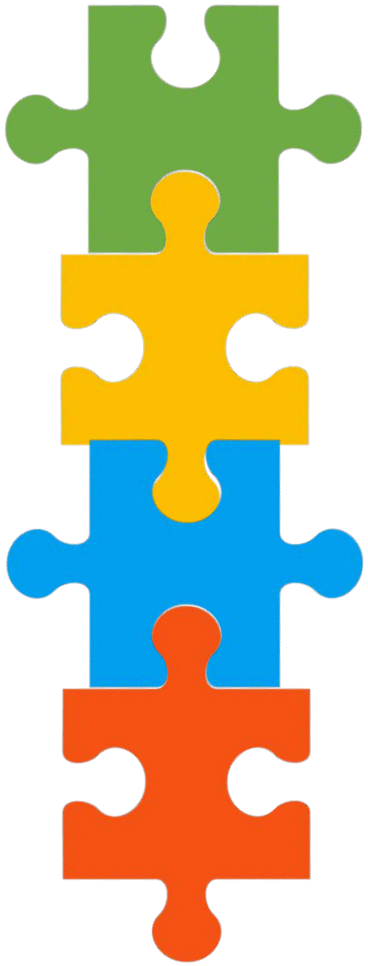


Microsoft

Azure

Cluster Services

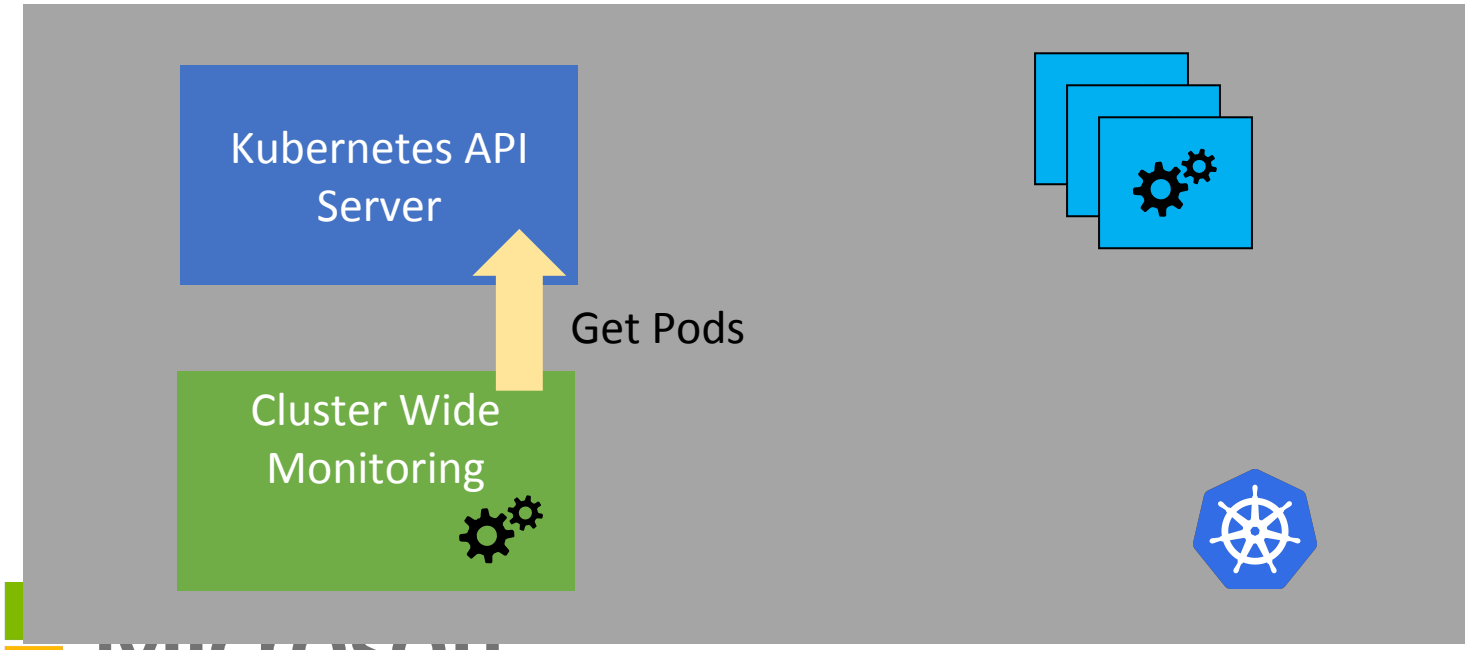


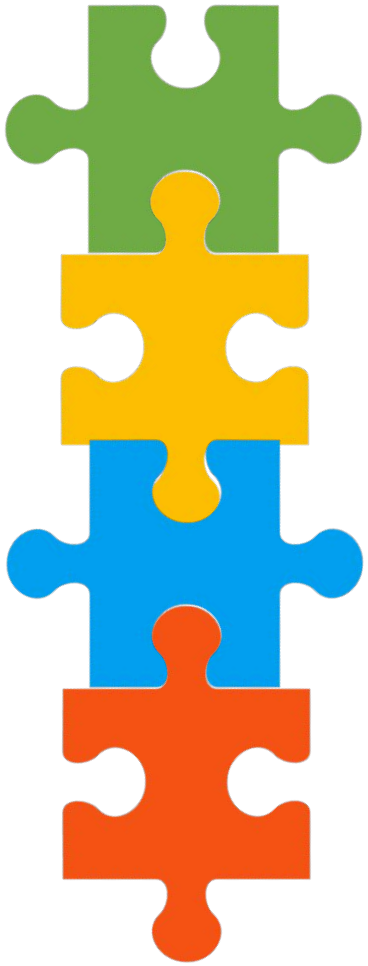


Microsoft

Azure

Cluster Services

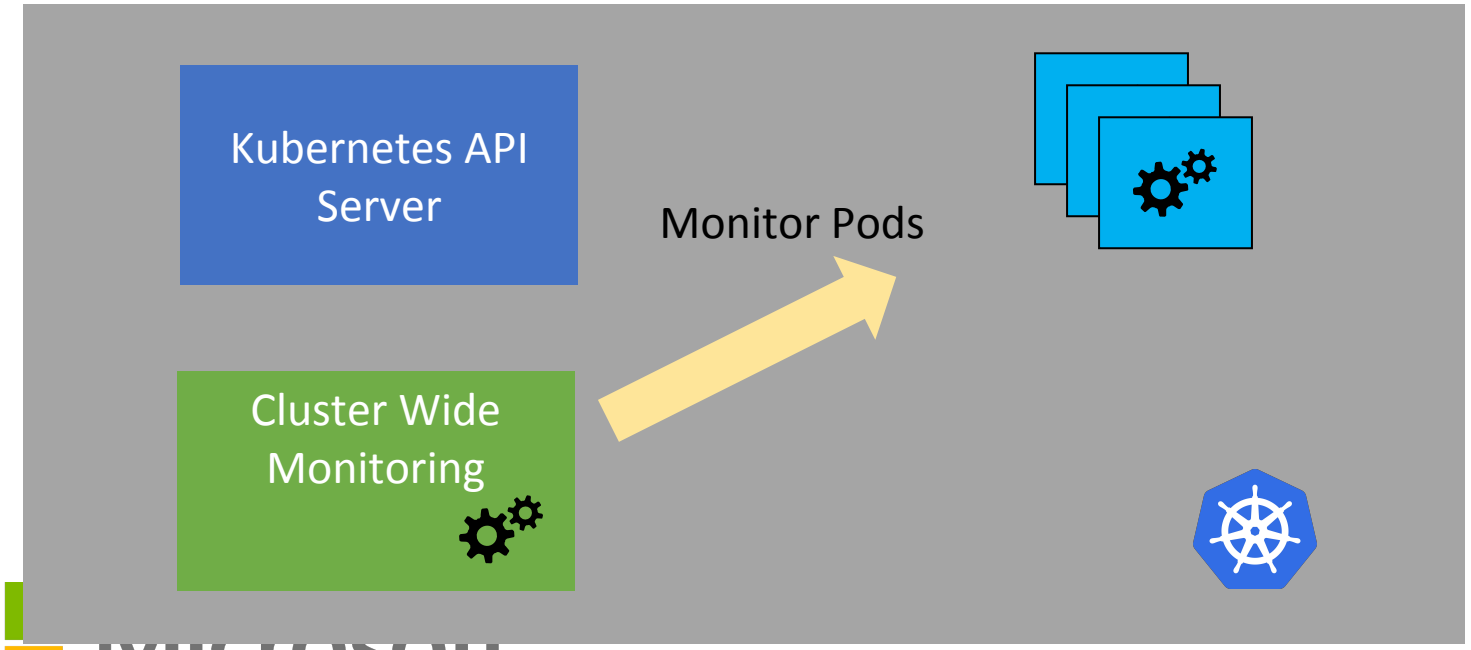


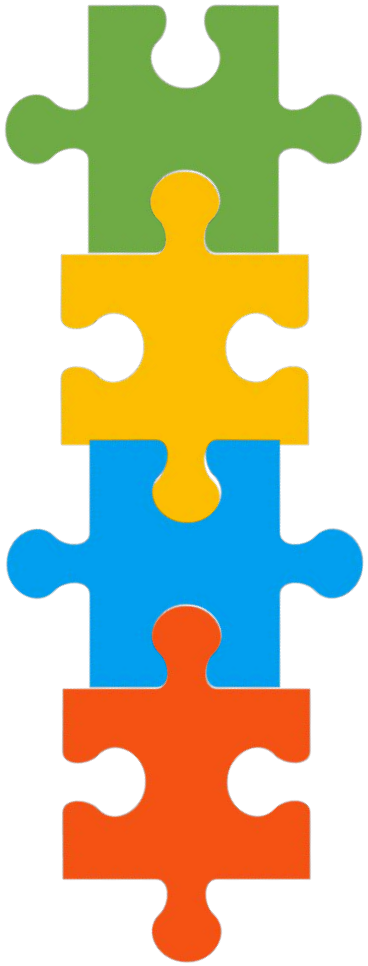


Microsoft

Azure

Cluster Services

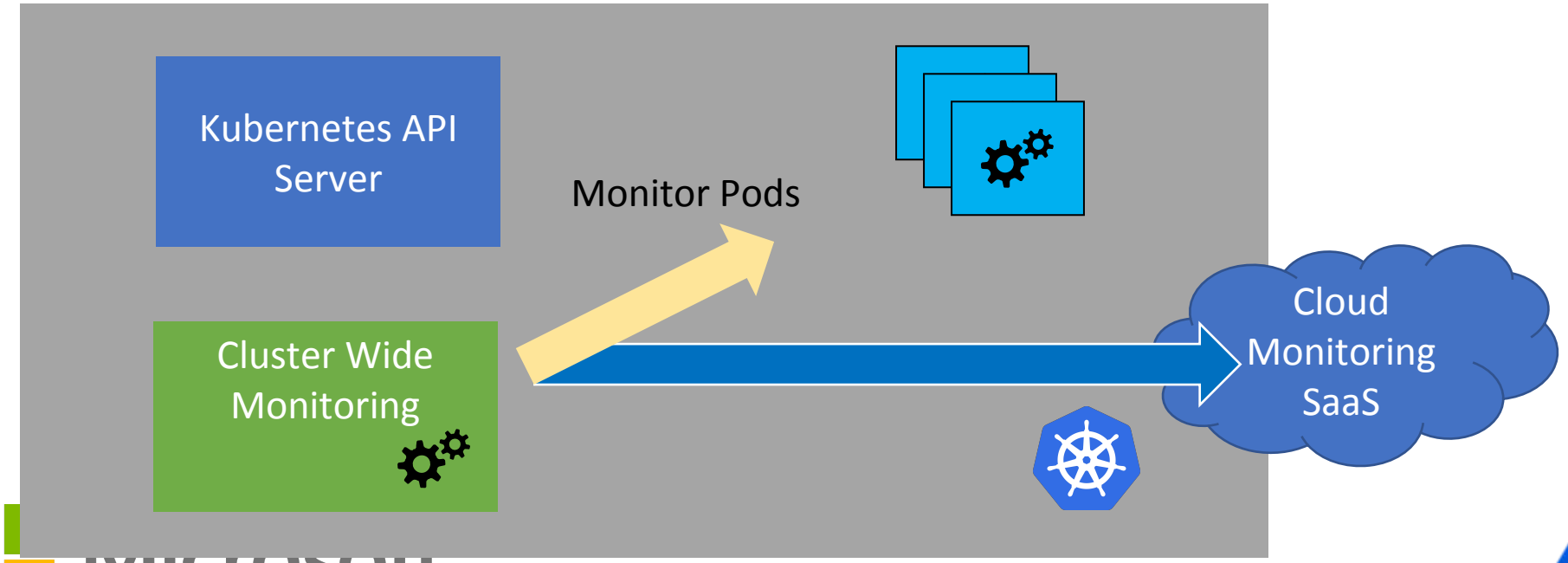


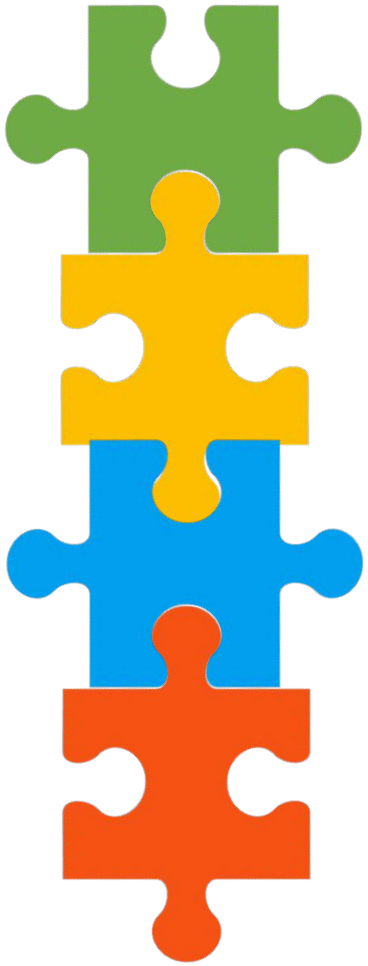


Microsoft

Azure

Cluster Services





Microsoft

Azure

Solutions: Patterns & Libraries

- Examples:
  - Posix
  - `npm install …`
  - Unit testing
  - Extensible libraries of patterns
  - …

Microsoft

Azure

Solutions: Patterns and libraries

- Brigade (https://brigade.sh)
- Service Mesh
  - Istio (https://istio.io)
  - linkerd (https://buoyant.io)
- Hadoop/Spark/…
- PaaS on top of Kubernetes

Microsoft

Azure

Brigade (https://brigade.sh)

Microsoft

Azure

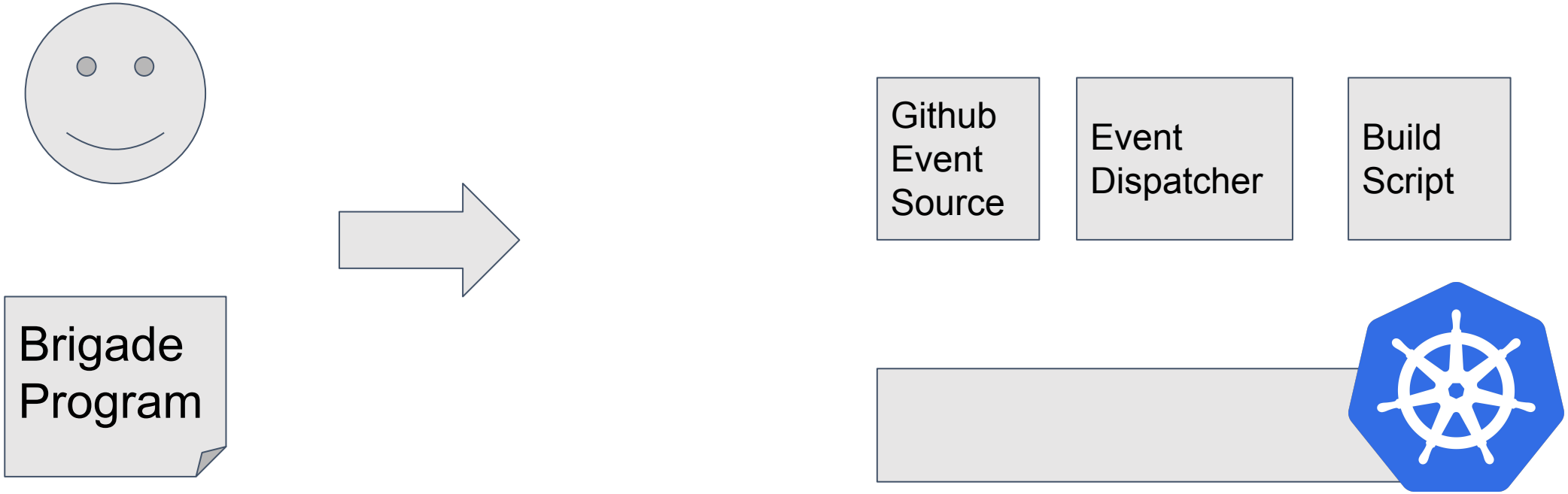
Brigade (https://brigade.sh)

```
const { events, Job , Group} = require("brigadier");
const dest = "$GOPATH/src/github.com/technosophos/ulid";

events.on("push", (e, p) => {
    var gh = JSON.parse(e.payload)
    var test = new Job("test", "golang:1.9")
    test.tasks = [
        "mkdir -p " + dest,
        "cp -a /src/* " + dest,
        "cd " + dest,
        "go get -u github.com/golang/dep/cmd/dep",
        "dep ensure",
        "make test"
    ];
    test.run() });
```

Microsoft

Azure

Brigade (https://brigade.sh)



Microsoft

Azure

Solutions: Language Features

- Examples:
  - `synchronized`
  - `async/await`
  - Garbage Collection
  - Single threading
  - Channels
  - …

Microsoft

Azure

How do we build a language idiomatic cloud?

Microsoft

Azure

The problem(s).

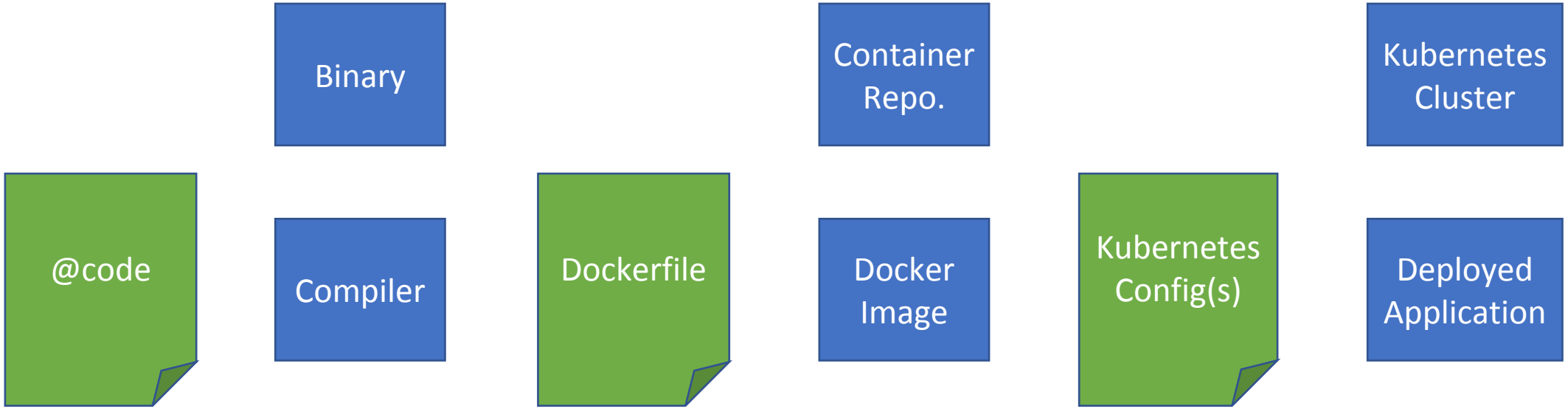
- Code to cloud*
- Cloud* to distributed system

* 'cloud' represents API driven deployment infrastructure

Microsoft

Azure

From code to cloud



Microsoft

Azure

From code to cloud

- At *least* three files
  - Code, Dockerfile, Kubernetes Config
- At *least* three languages
  - Code, Dockerfile, YAML
- At *least* three tools
  - Editor, docker, kubectl

Microsoft

Azure

And we haven't talked distributed systems...

Microsoft

Azure

Or day two operations…

Microsoft

Azure

Does this seem ok?

Microsoft

Azure

Design Principles

It has to be a formal programming language

Microsoft

Azure

Design Principles

It can’t be a PaaS/Walled Garden

Microsoft

Azure

Design Principles

It needs to be idiomatic

Microsoft

Azure

Design Principles

It has to be open source

Microsoft

Azure

More learning from history

```
public class MyMapper { … }

public class MyReducer { … }
```

Microsoft

Azure

More learning from (recent) history

```
values = [...]
values.map(lambda x … )
```

Microsoft

Azure

More learning from (recent) history

```
values = [...]
values.map(lambda x … )

pywren.map(values, lambda x …)
```

Microsoft

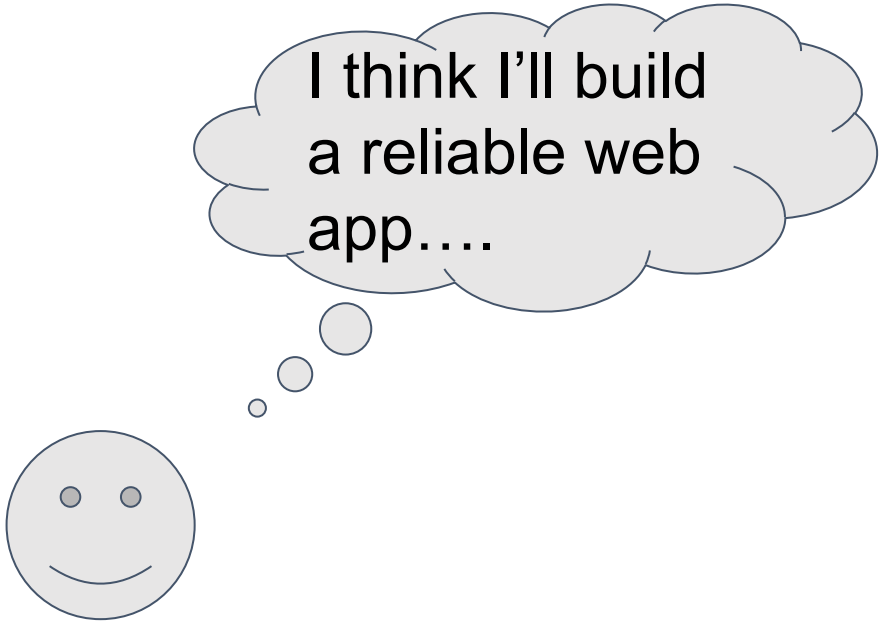
Azure

Metaparticle (https://metaparticle.io)

Microsoft

Azure

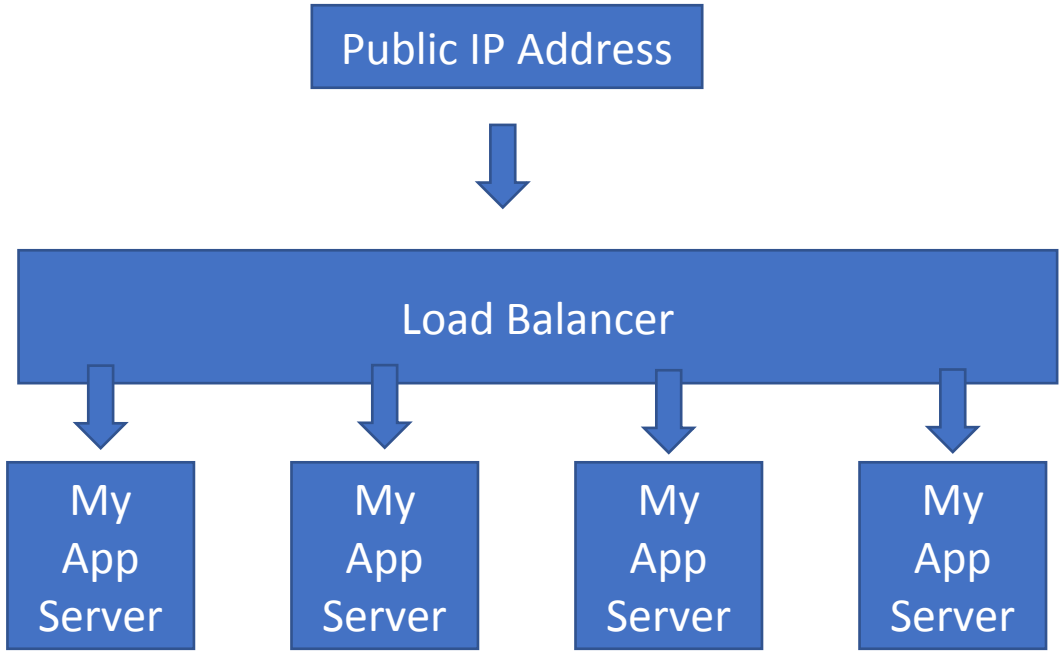
Metaparticle (https://metaparticle.io)



Microsoft

Azure

Metaparticle (https://metaparticle.io)



Microsoft

Azure

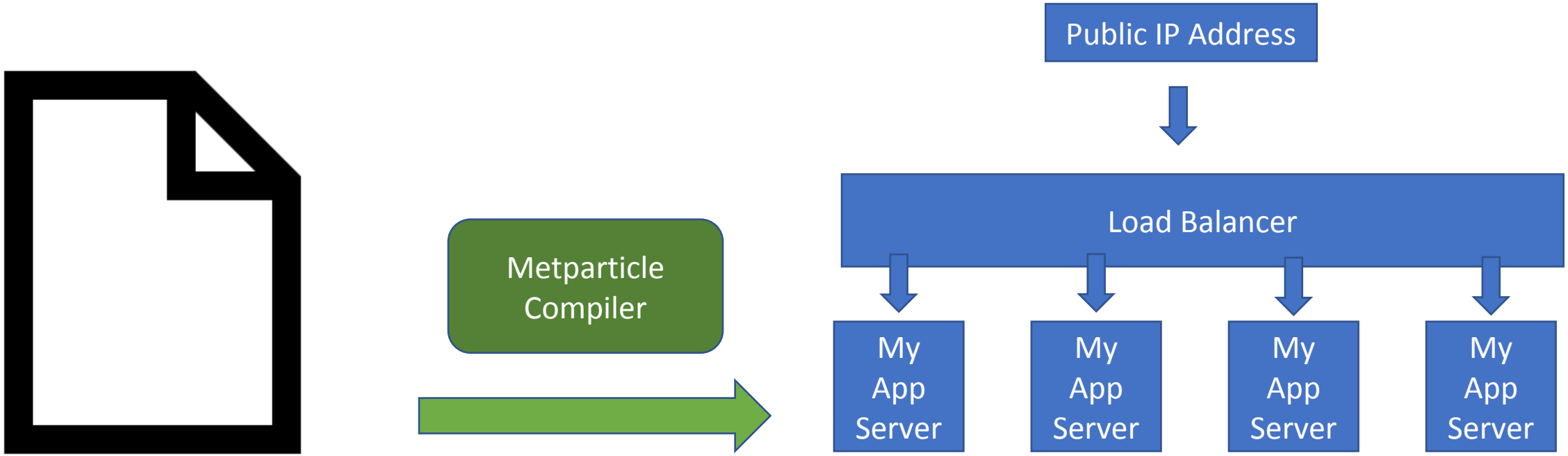
Metaparticle (Python)

```
from metaparticle import containerize

@containerize(
    options={
        'ports': [8080],
        'replicas': 4,
    })

def main():
    Handler = MyHandler
    httpd = SocketServer.TCPServer(("", port), Handler)
    httpd.serve_forever()
```

Microsoft

Azure

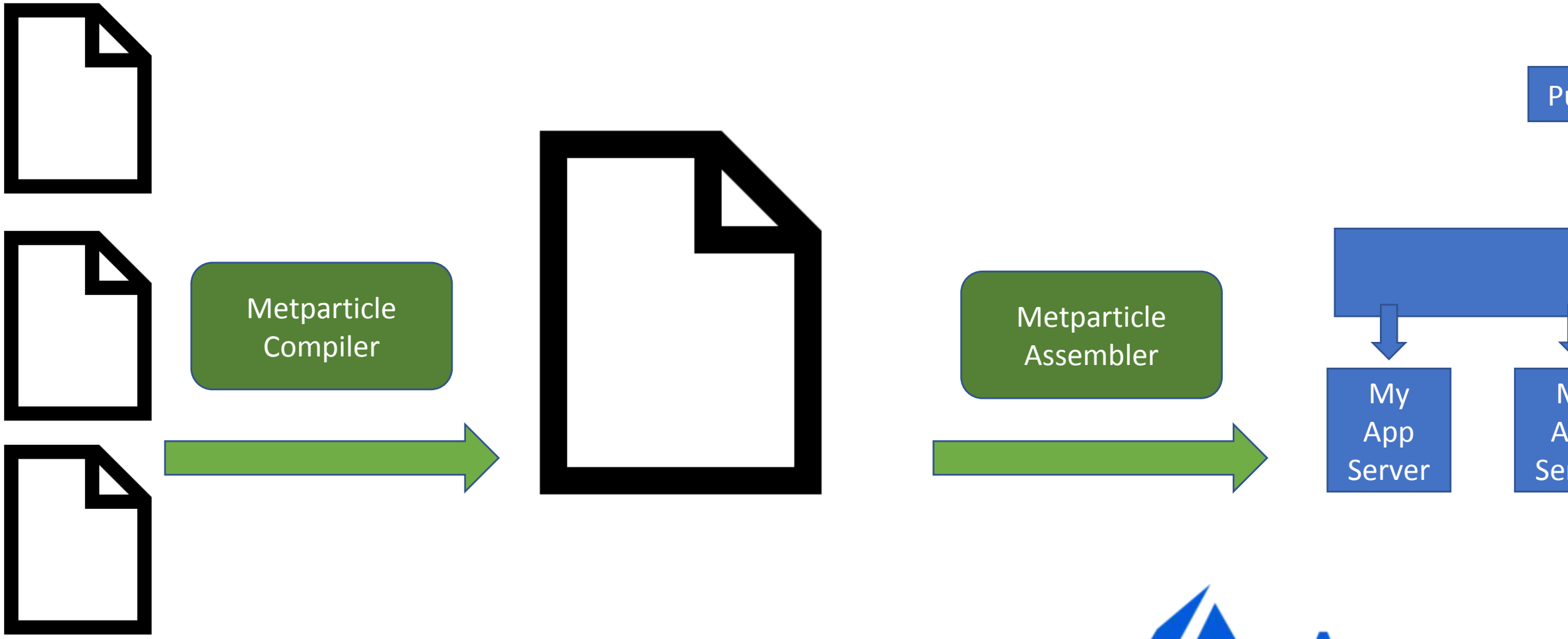
Metaparticle



Microsoft

Azure

Metaparticle



Microsoft

Azure

Metaparticle (Javascript)

```
const mp = require('@metaparticle/package');

const server = http.createServer((request, response) => {
   response.end(`Hello World: hostname: ${os.hostname()}\n`);
});

mp.containerize(
   {
      'replicas': 4
   },
   () => {
      server.listen(port, () => {
         console.log(`server up on ${port}`);
      });
   }
);
```

Microsoft

Azure

Metaparticle Languages

- Python
- Javascript
- Java
- C#
- Go
- Rust
- Ruby
- <your favorite language here>

Microsoft

Azure

Metaparticle: Demo

Microsoft

Azure

more containers, more problems...

Microsoft

Azure

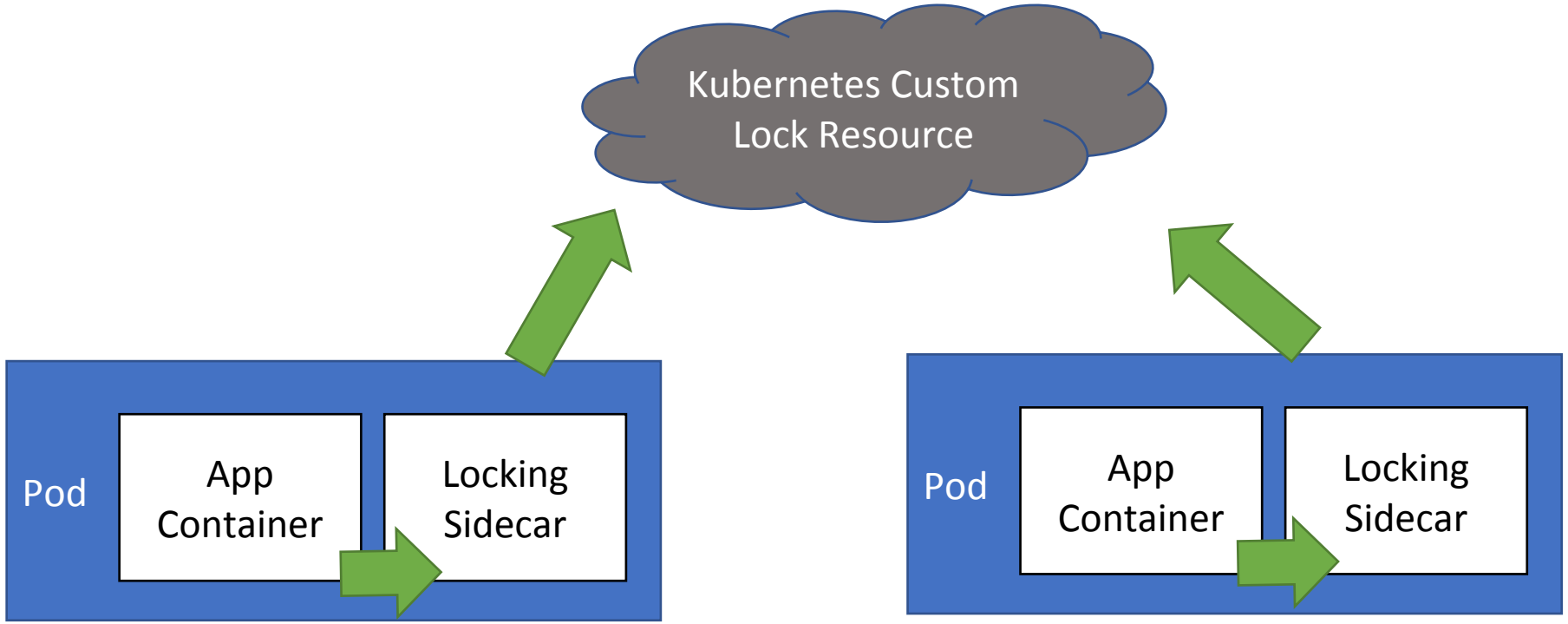
Metaparticle: Synchronization

```
import io.metaparticle.sync.Lock;

public class Main {
    public static void main(String[] args) throws InterruptedException {
        Lock lock = new Lock("test");

        lock.lock();
        Thread.sleep(45 * 1000);
        lock.unlock();
    }
}
```

Microsoft

Azure

Metaparticle Synchronization



Microsoft

Azure

Metaparticle: Synchronization

```
import metaparticle_sync

def master_fn():
    print('I am the master')

def lost_master_fn():
    print('I lost the master')

el = metaparticle_sync.Election('test', master_fn, lost_master_fn)
el.run()
```

Microsoft

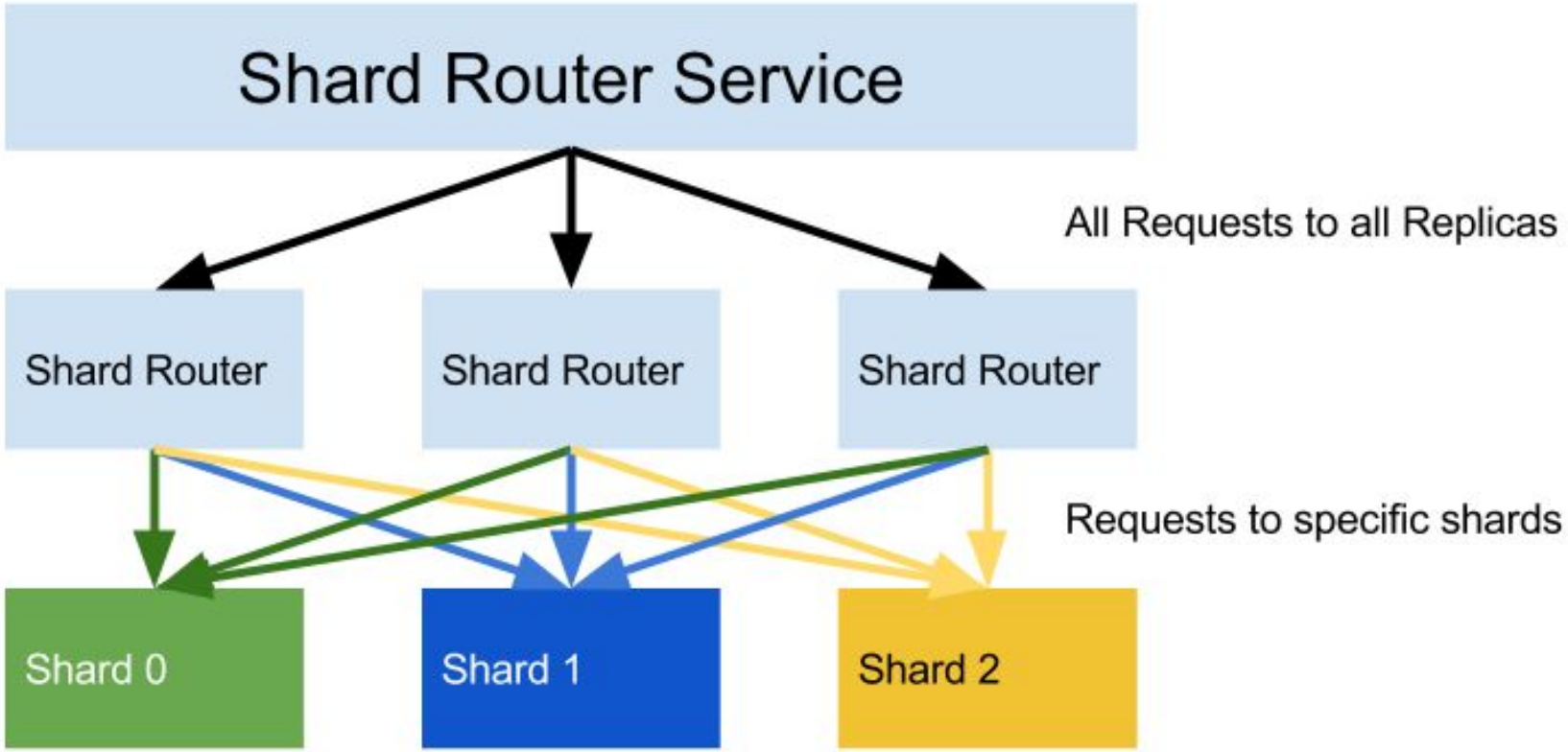
Azure

Moving on…

Microsoft

Azure

Metaparticle: Sharding



Microsoft

Azure

Metaparticle: Sharding

```
@Runtime(
   shards = 4,
   urlShardPattern = "^\\/users\\/([^\\/]*)\\/.*")
@Package(
   repository="brendanburns",
   jarFile="target/metaparticle-package-0.1-SNAPSHOT-jar-with-dependencies.jar")

public static void main(String[] args) {
   Containerize(() -> {
      try {
         HttpServer server = HttpServer.create(new InetSocketAddress(8080), 0);
         server.createContext("/", new HttpHandler() {
               . . .
         });
   });
}
```

Microsoft

Azure

Metaparticle: Demo

Microsoft

Azure

Taking it further: MetaML

Microsoft

Azure

MetaML Example

https://aka.ms/metaml

```
@Train(
  package={
    'name': 'some_image_name',
    'repository': 'some_docker_repo',
    'publish': True},
  strategy=HyperparameterTuning({
    'learning_rate': random.normalvariate(0.5, 0.5)},
     parallelism=3),
  tensorboard={ 'log_dir': FLAGS.log_dir, } )

def my_training_function(learning_rate):
   # some training logic ...
```

Microsoft

Azure

MetaML Example

```
from metaml.serve import Serve

@Serve(package={
    'name': 'simple-serve',
    'repository': 'wbuchwalter',
    'publish': True})

def func():
  return "hello world"

func()
```

Microsoft

Azure

Where to next?

- https://metaparticle.io
- https://github.com/metaparticle-io/

- https://aka.ms/metaml

Microsoft

Azure