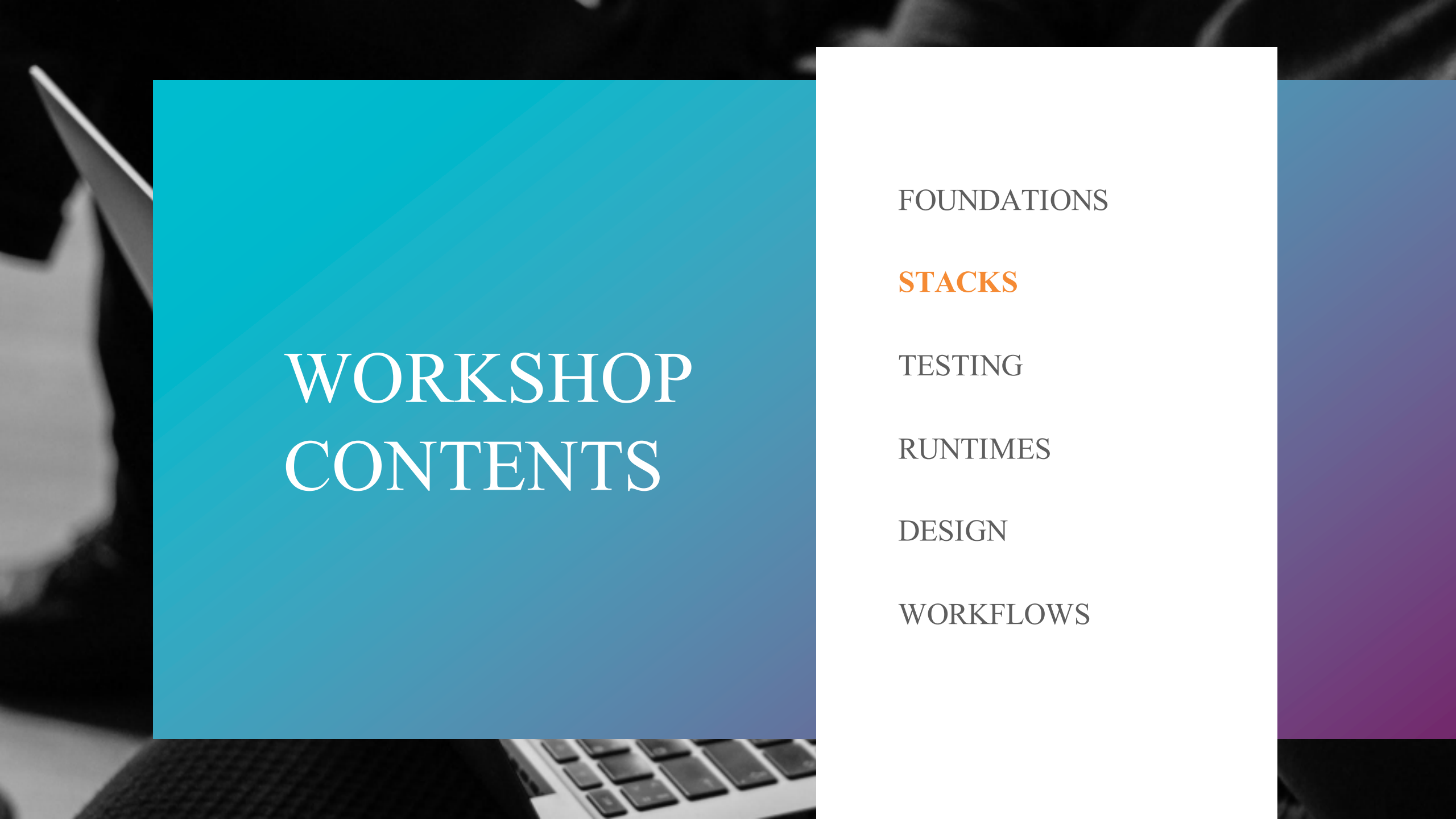
PART 2: STACKS

Infrastructure as Code

Kief Morris @ ThoughtWorks



WORKSHOP CONTENTS

FOUNDATIONS

STACKS

TESTING

RUNTIMES

DESIGN

WORKFLOWS

PART 2

INFRASTRUCTURE STACKS

STACK STRUCTURES

ENVIRONMENTS

CONFIGURING STACKS

SECRETS

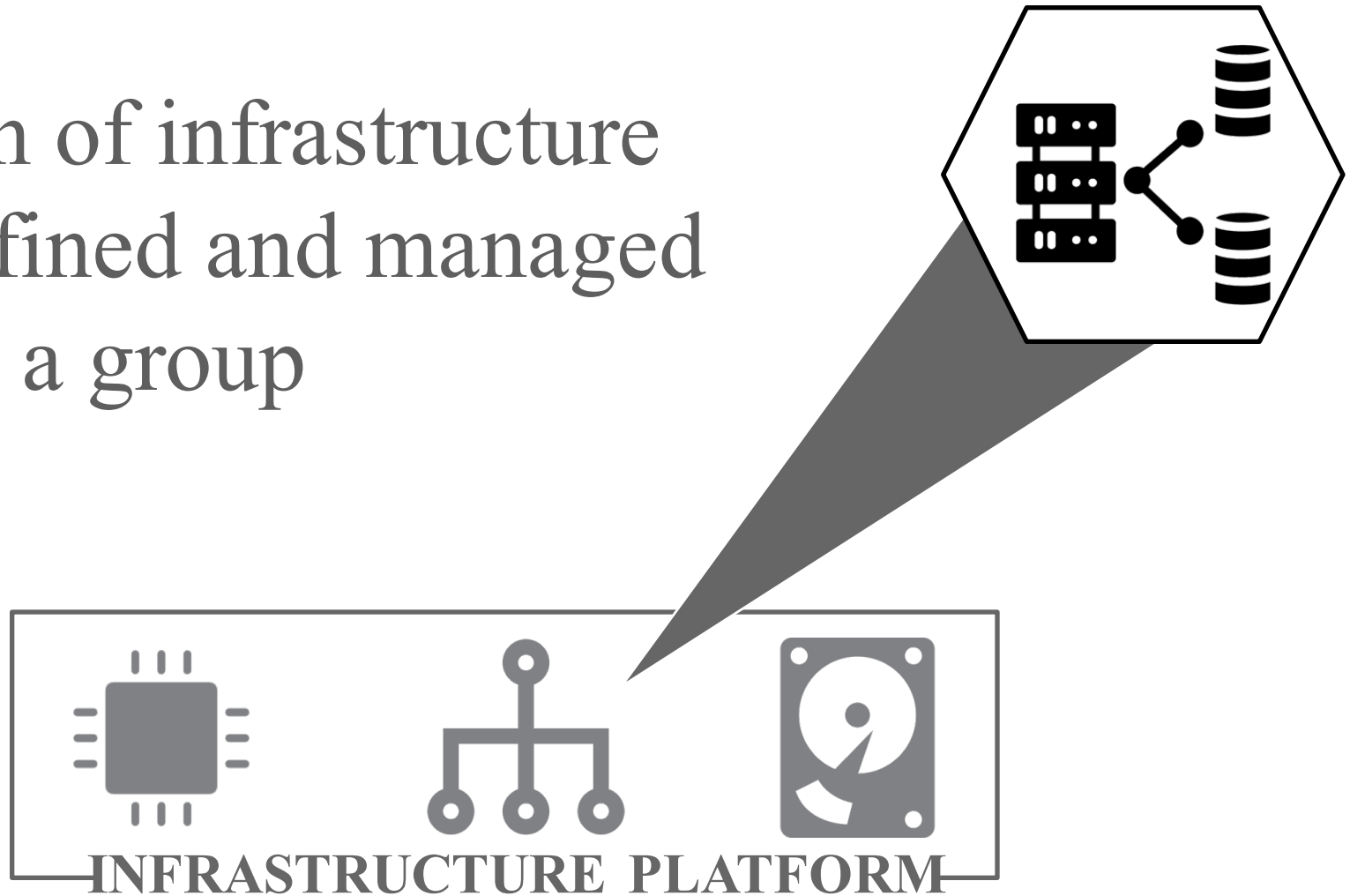
*An "architectural quantum" is an **independently deployable component** with high functional cohesion, which includes all the structural elements required for the system to function properly.*

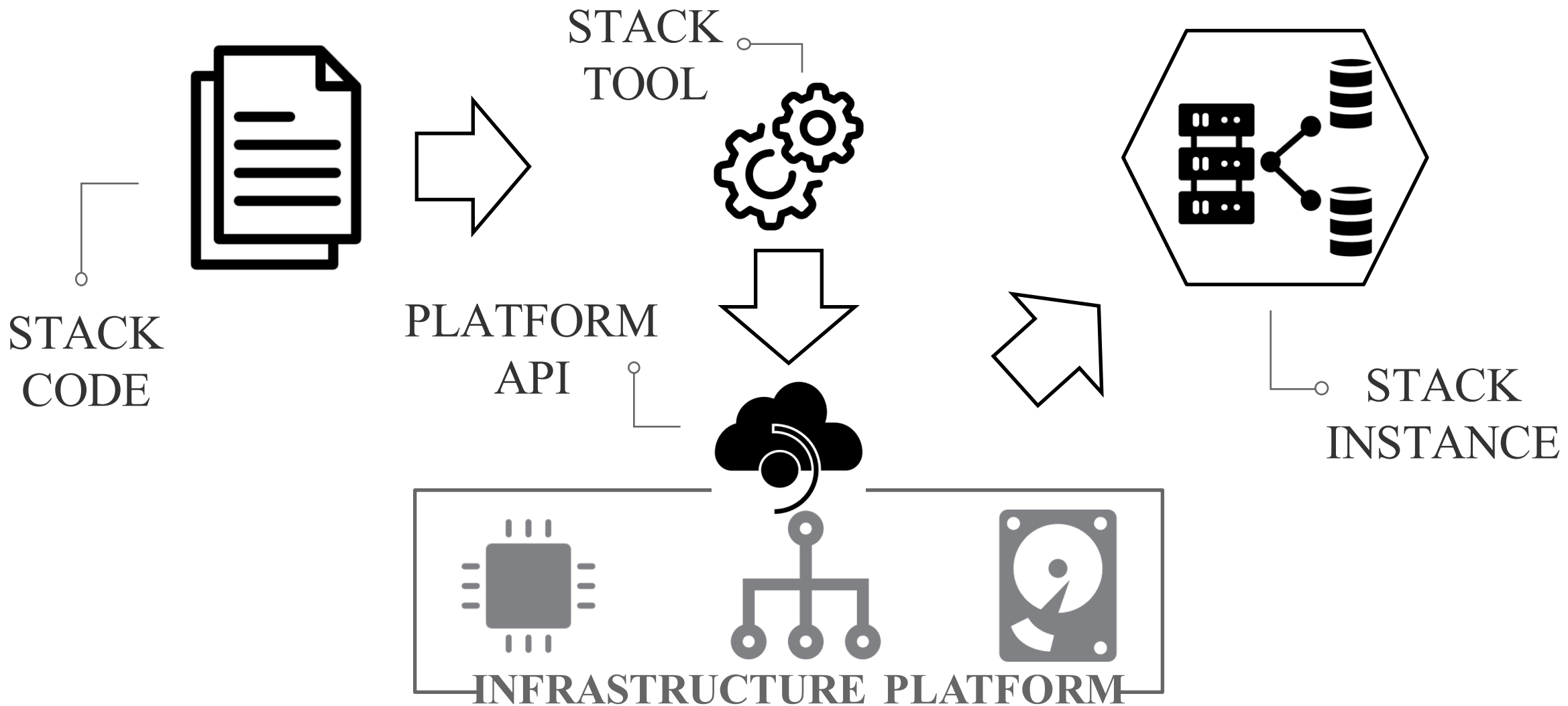
Building Evolutionary Architectures

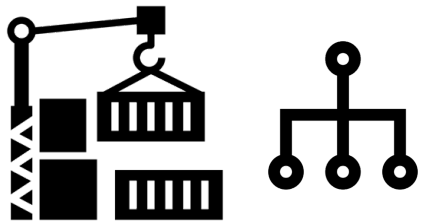
Ford, Parsons, Kua

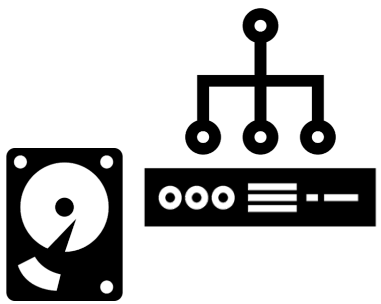
INFRASTRUCTURE STACK

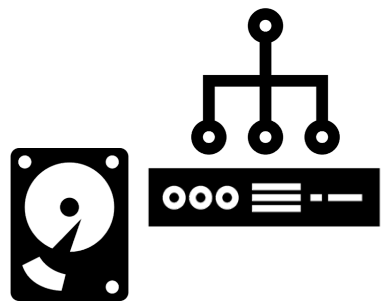
A collection of infrastructure resources defined and managed as a group

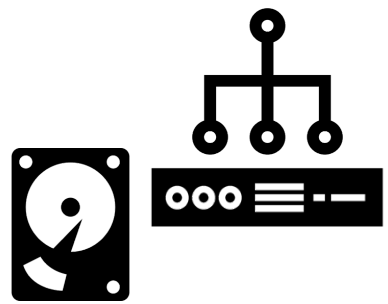


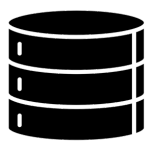

STACK
TOOL
STACK
CODE
PLATFORM
API
STACK
INSTANCE
INFRASTRUCTURE PLATFORM

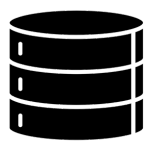


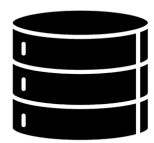








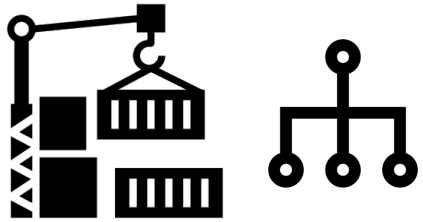




AN EXAMPLE SYSTEM

ONLINE RETAIL PLATFORM

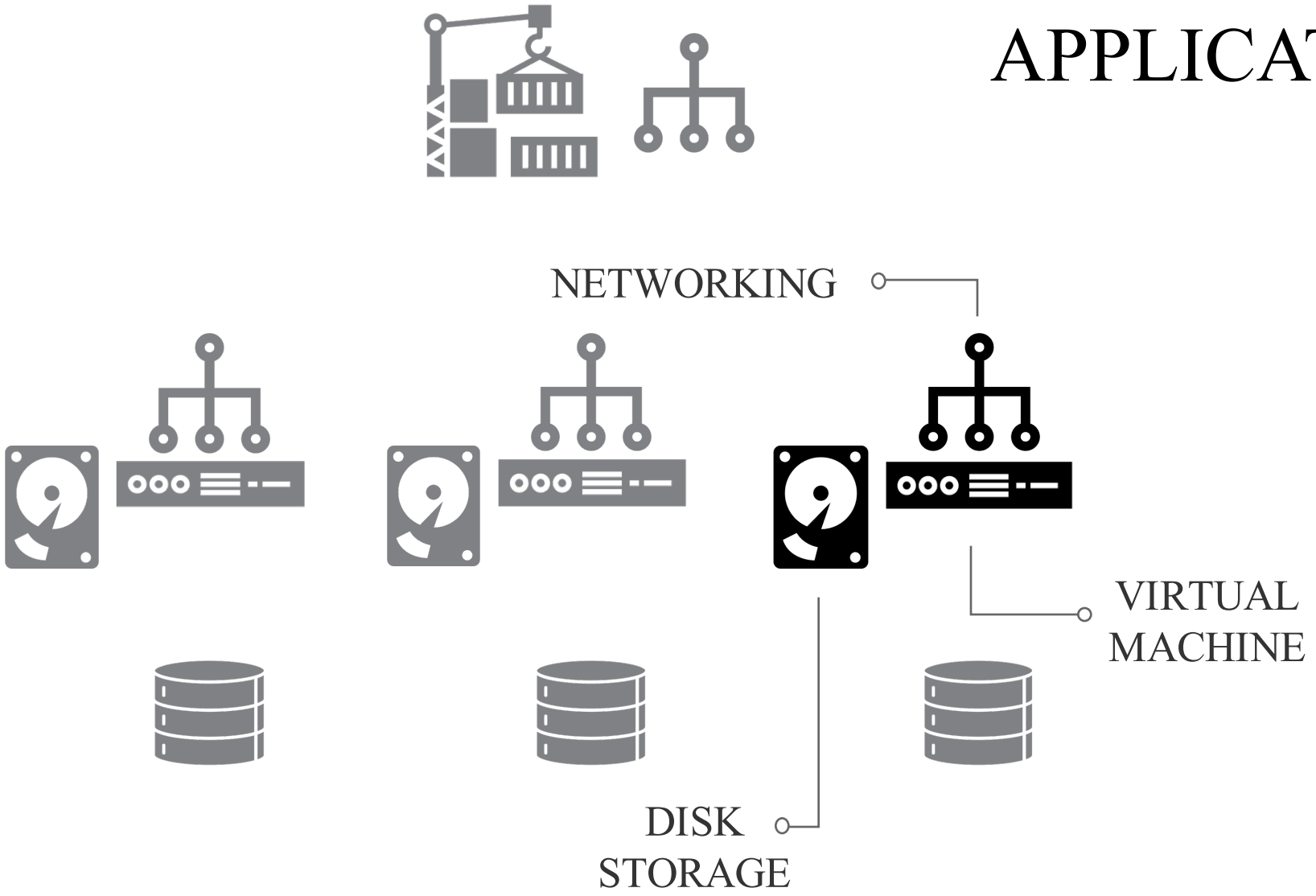
CONTAINER
CLUSTER



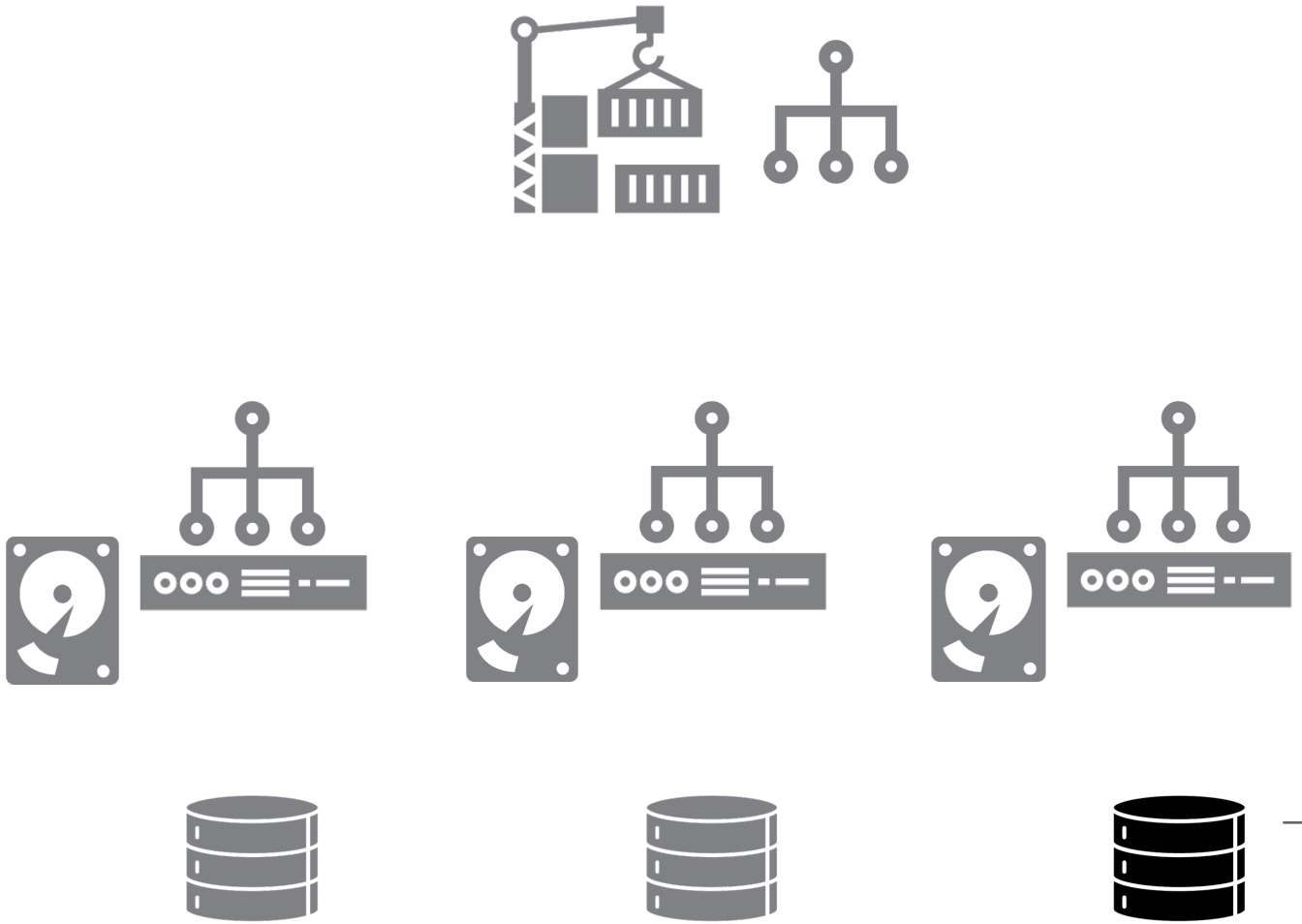
NETWORKING

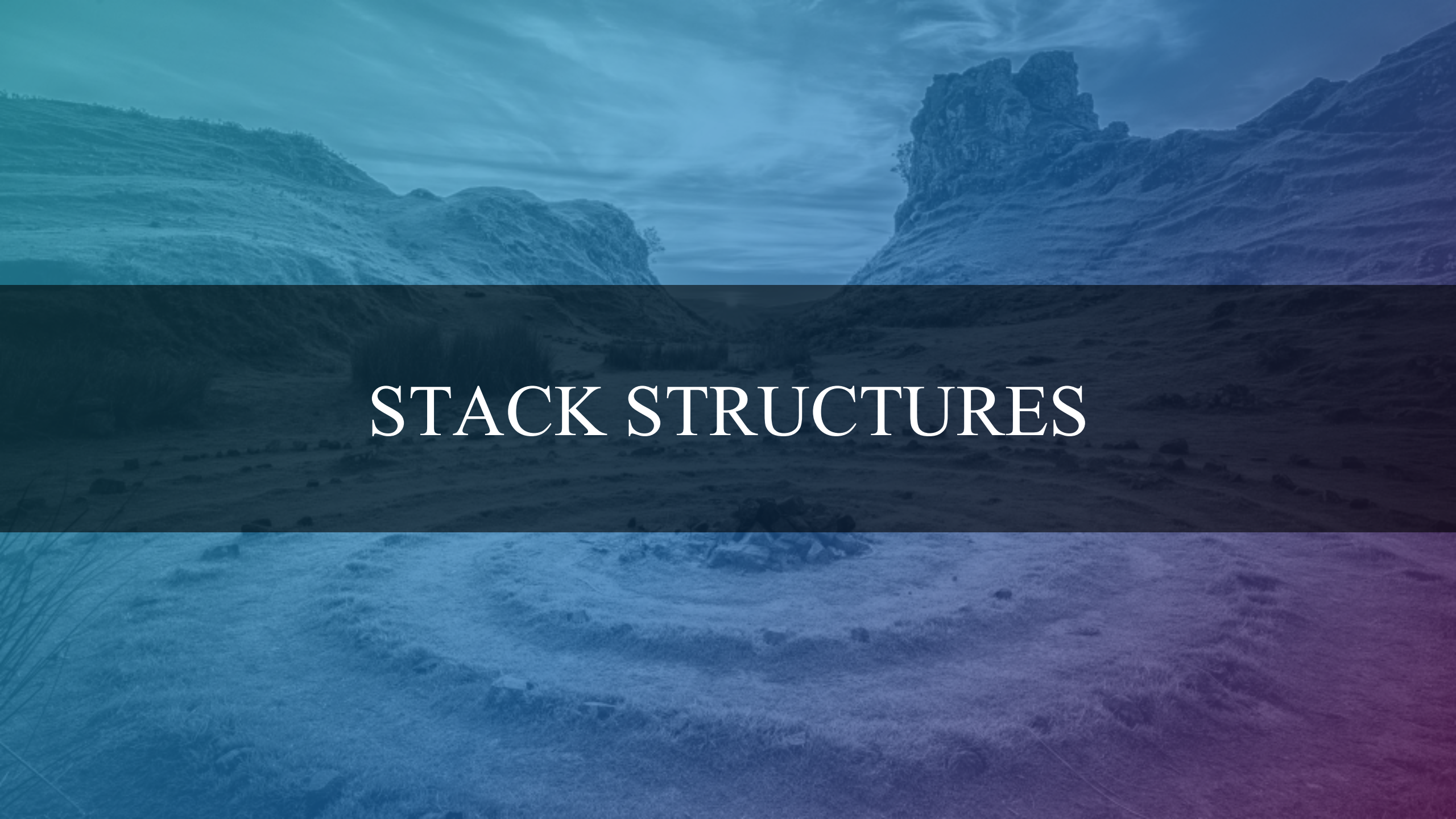
WEB SERVER INSTANCES

APPLICATION SERVERS

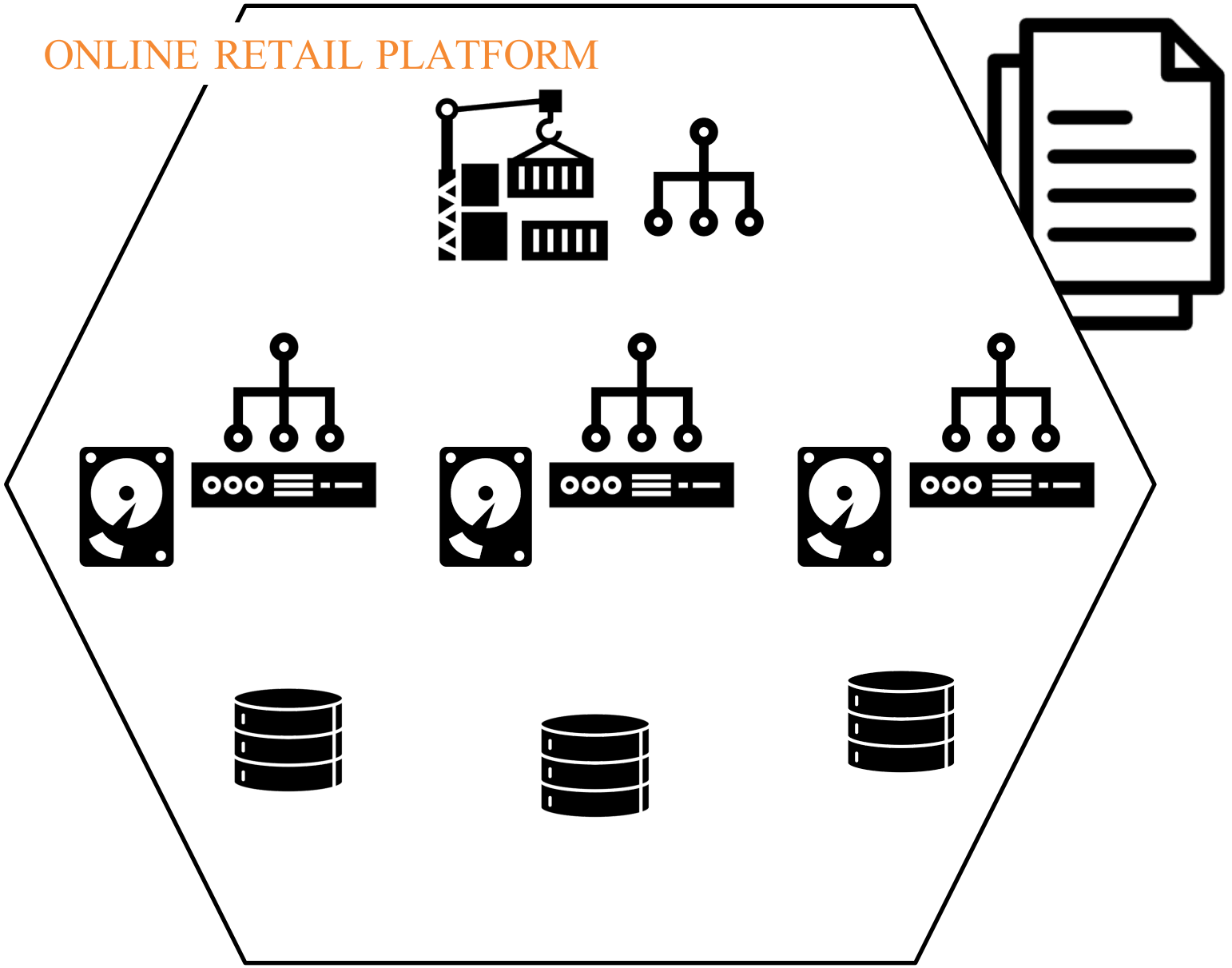


DATABASES





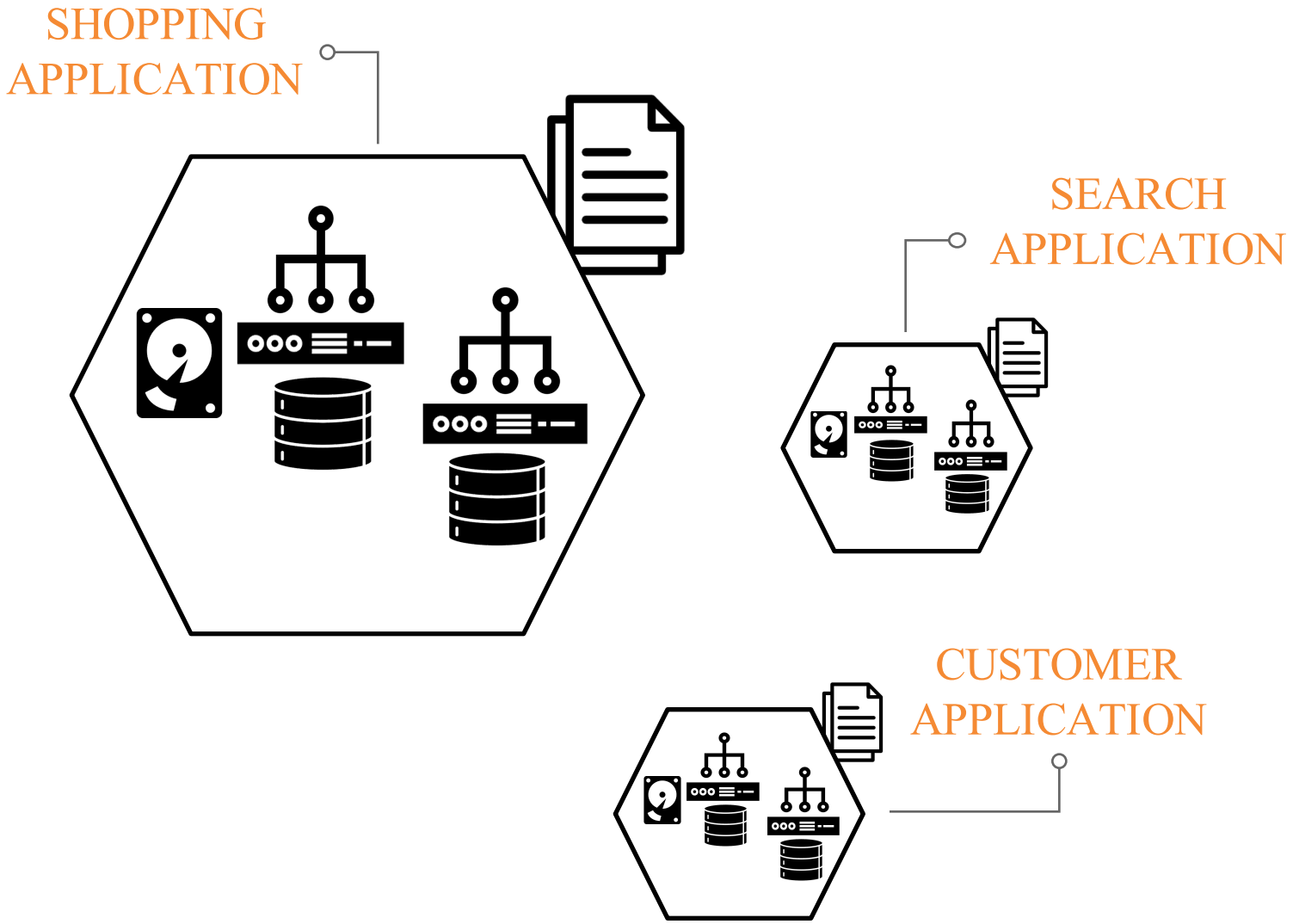
STACK STRUCTURES



MONOLITHIC STACK

EVERYTHING IN ONE STACK

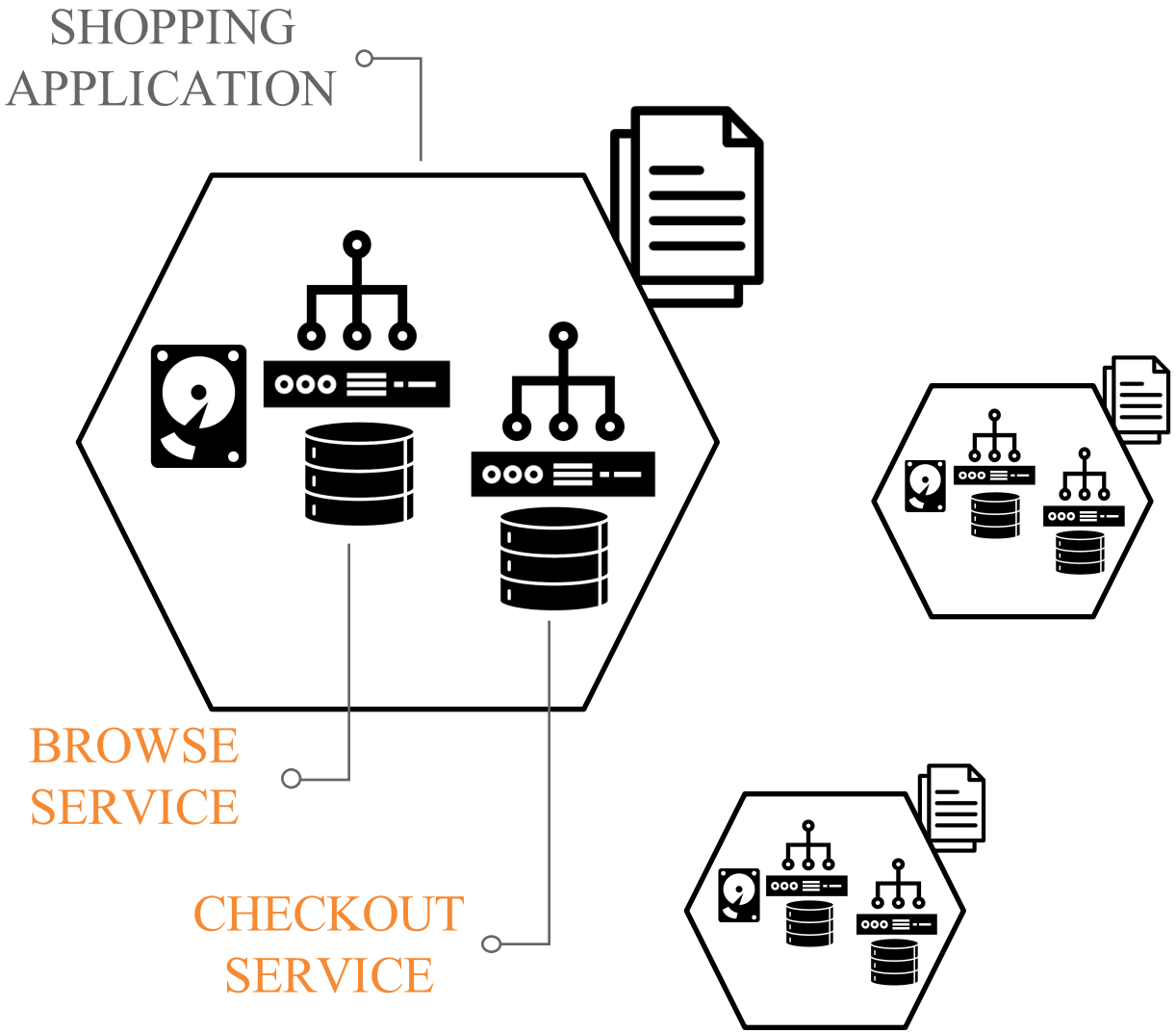
Changes are difficult and risky



APPLICATION GROUP STACK

GROUPED APPLICATIONS

Each stack includes all of the services / processes for a given application

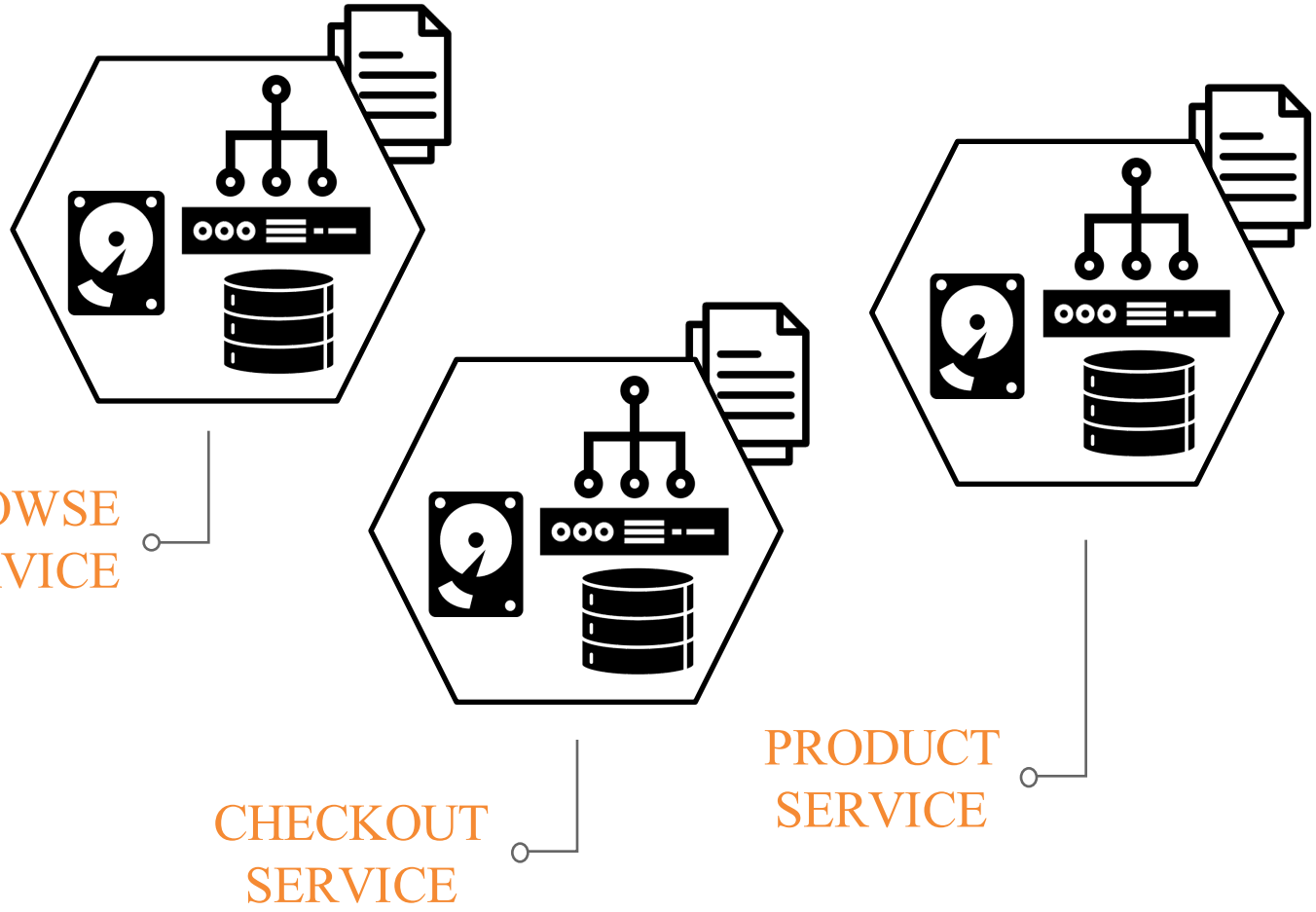


APPLICATION GROUP STACK

GROUPED APPLICATIONS

Each stack includes all of the services / processes for a given application

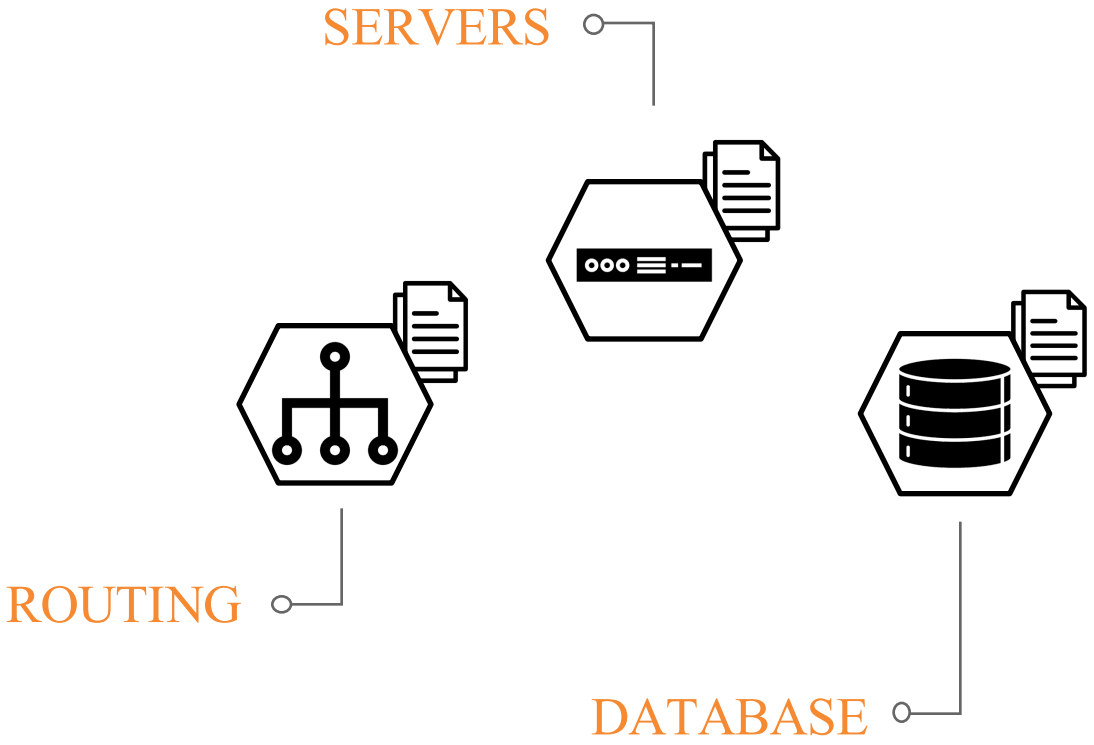
SHOPPING APPLICATION



SERVICE STACK

ONE STACK PER SERVICE

Includes all of the infrastructure related to a particular process

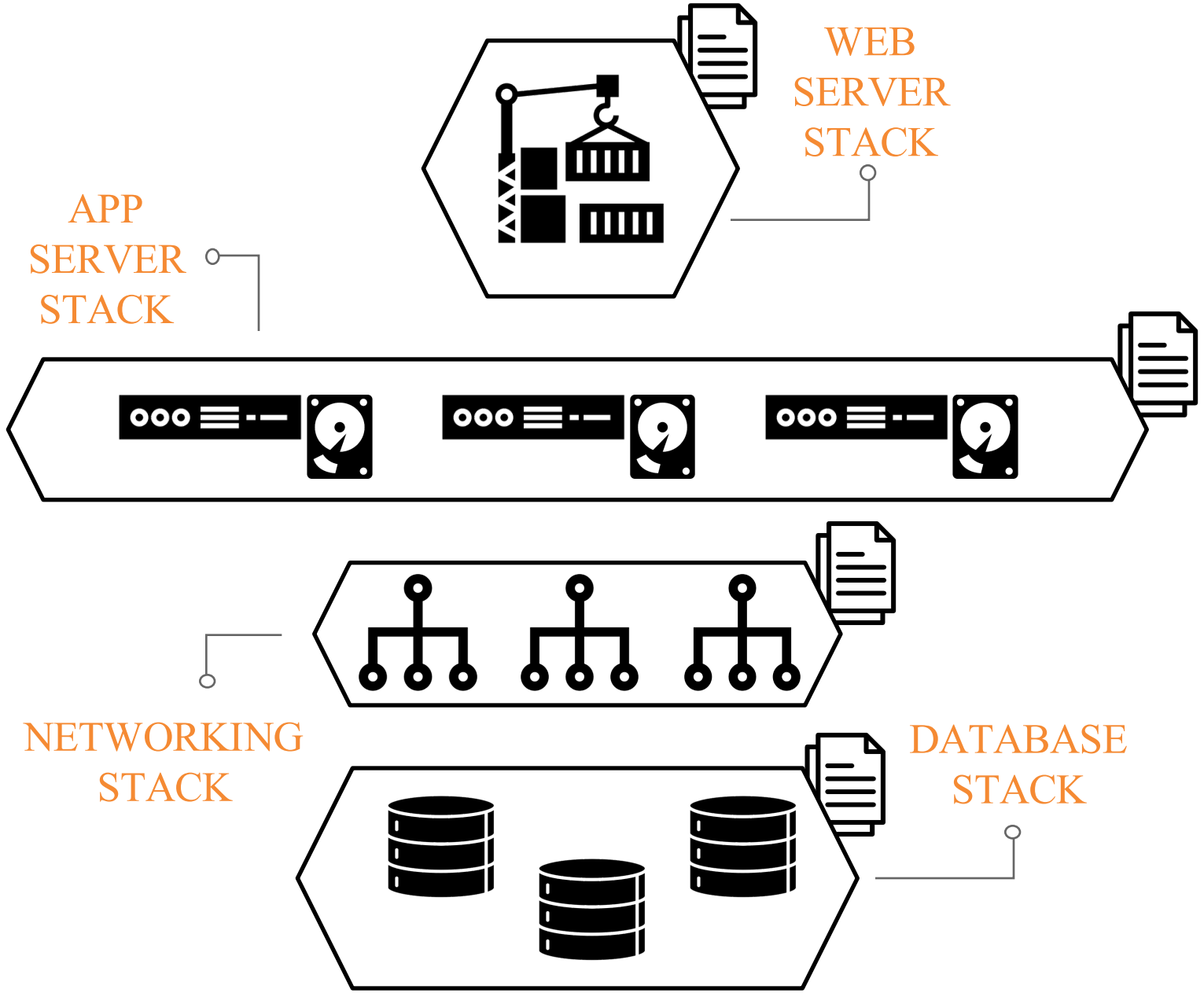


BROWSE SERVICE

MICRO-STACK

MULTIPLE STACKS PER SERVICE

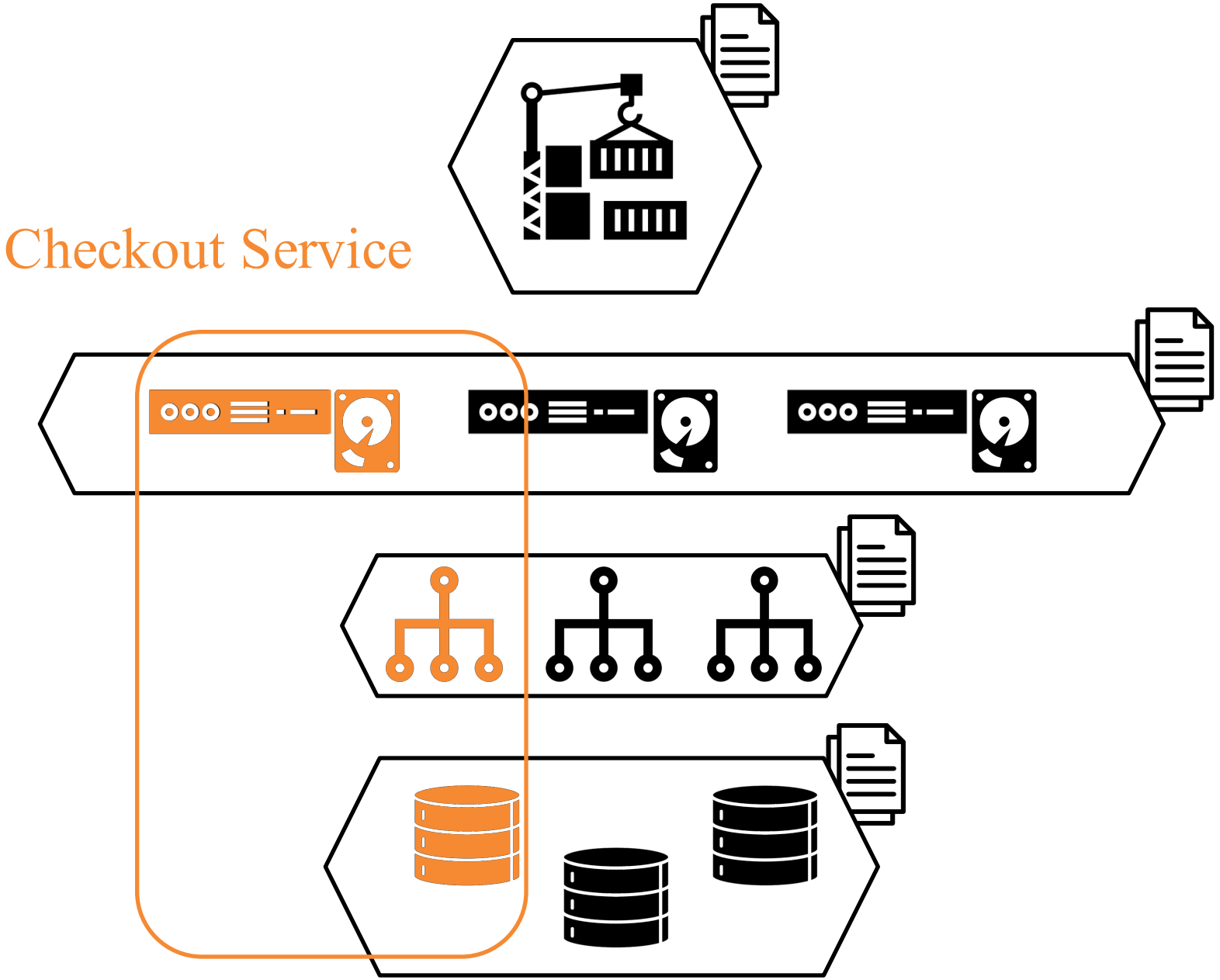
Different elements of a service's infrastructure are managed in different stacks



CROSS-SERVICE STACKS

ANTIPATTERN

Stacks include parts of multiple applications



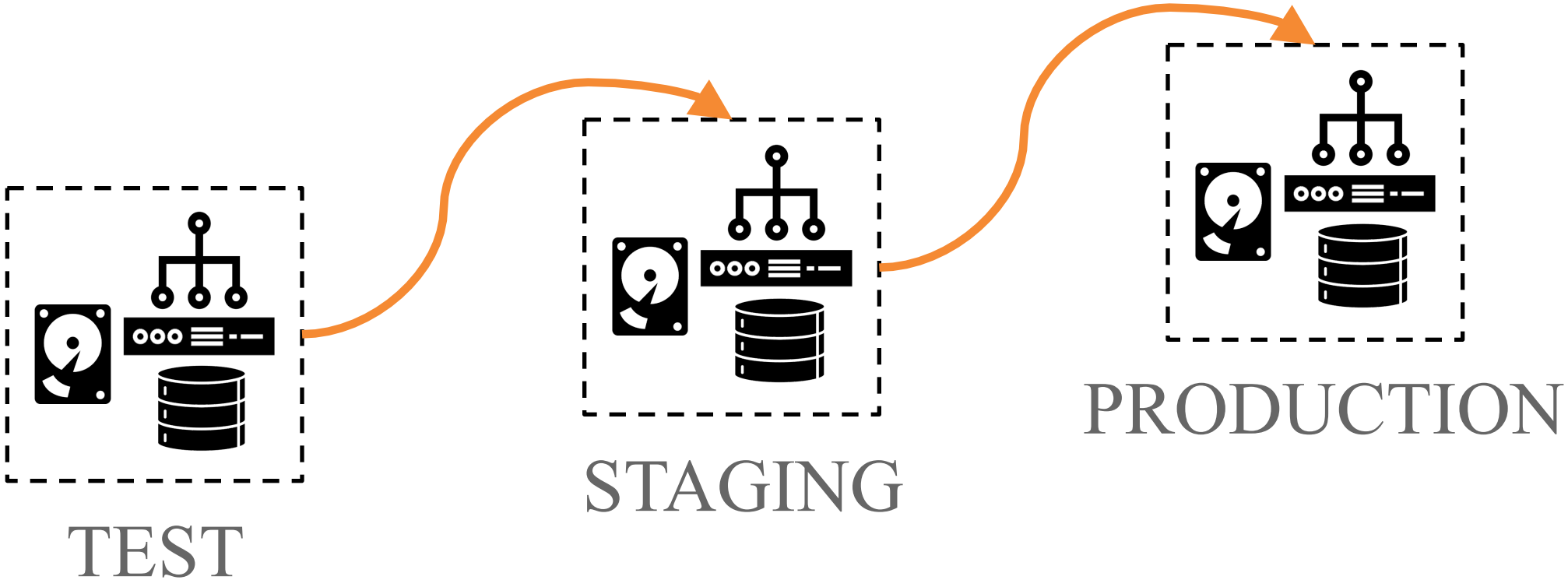
CROSS-SERVICE STACKS

ANTIPATTERN

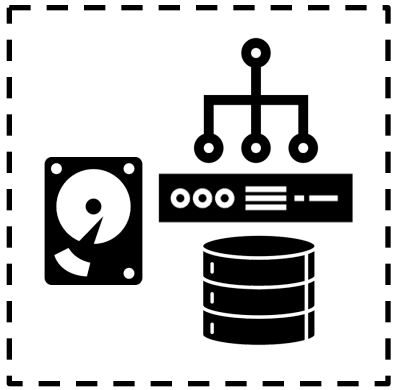
Stacks include parts of multiple applications

Adds barriers and risks for changes

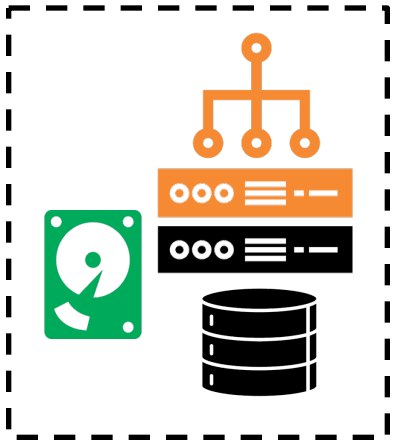
BUILDING ENVIRONMENTS WITH STACKS



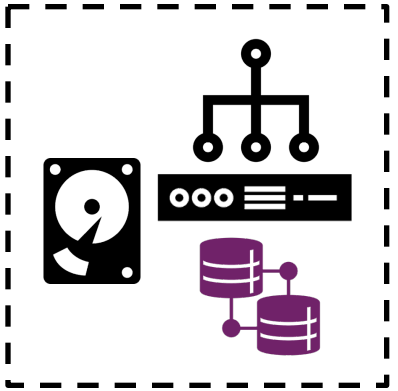
DELIVERY ENVIRONMENTS



TEST

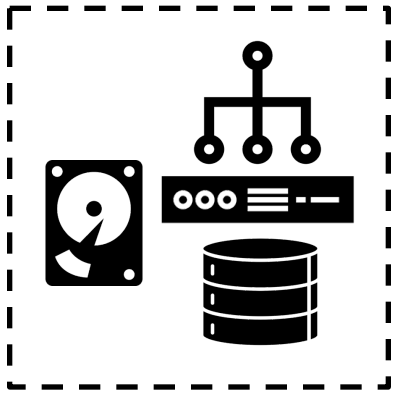


STAGING

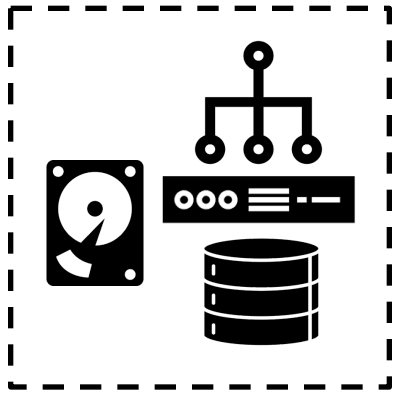


PRODUCTION

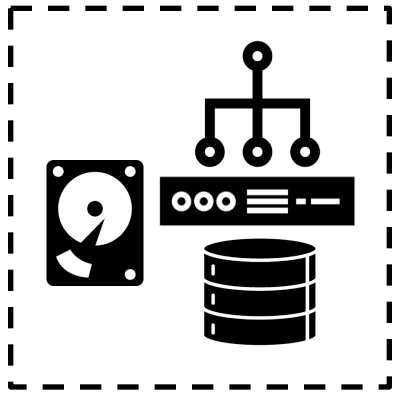
CHALLENGE:
INCONSISTENCY



US

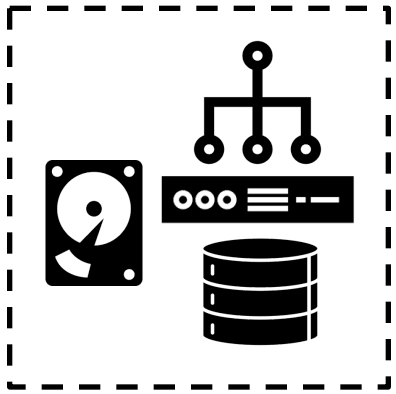


EUROPE

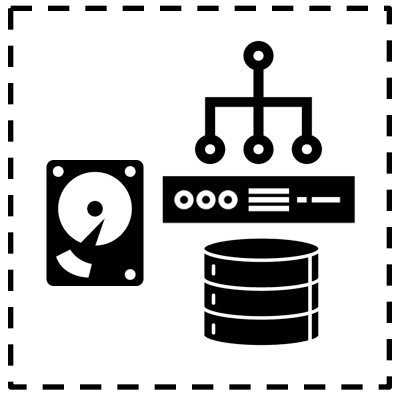


APAC

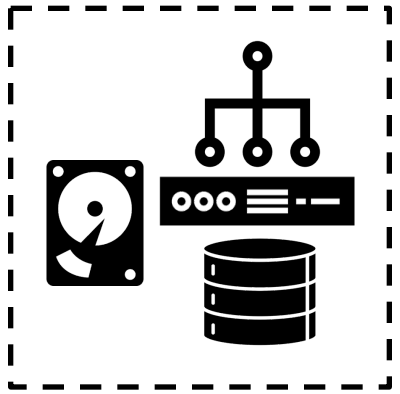
MULTIPLE PRODUCTION ENVIRONMENTS



US



EUROPE



APAC

CHALLENGE: MAINTENANCE

GOALS

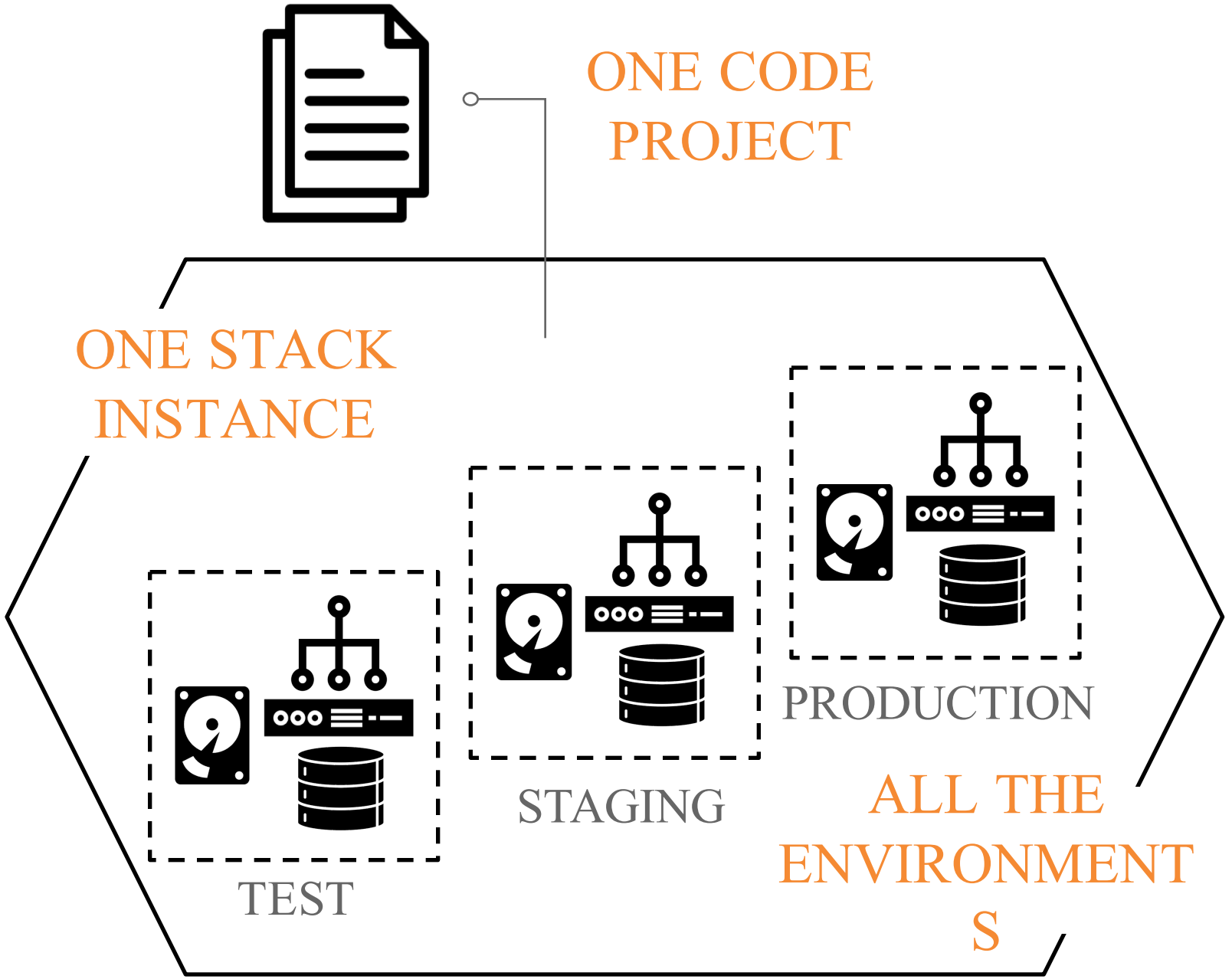
Make changes across environments

- Easily and quickly
- Safely and reliably
- Consistently

MULTI-ENVIRONMENT STACK

ANTIPATTERN

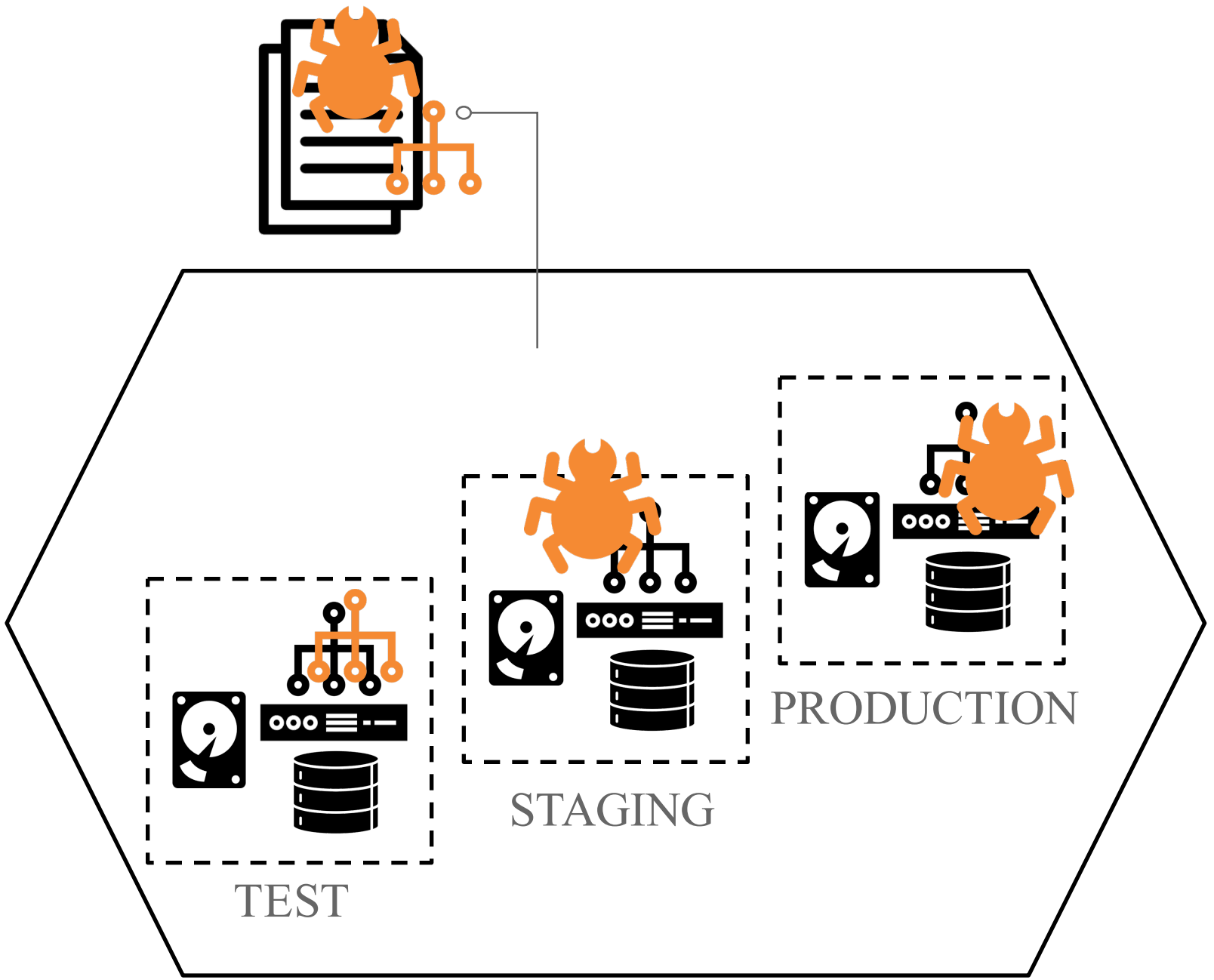
All your environments in a single code project

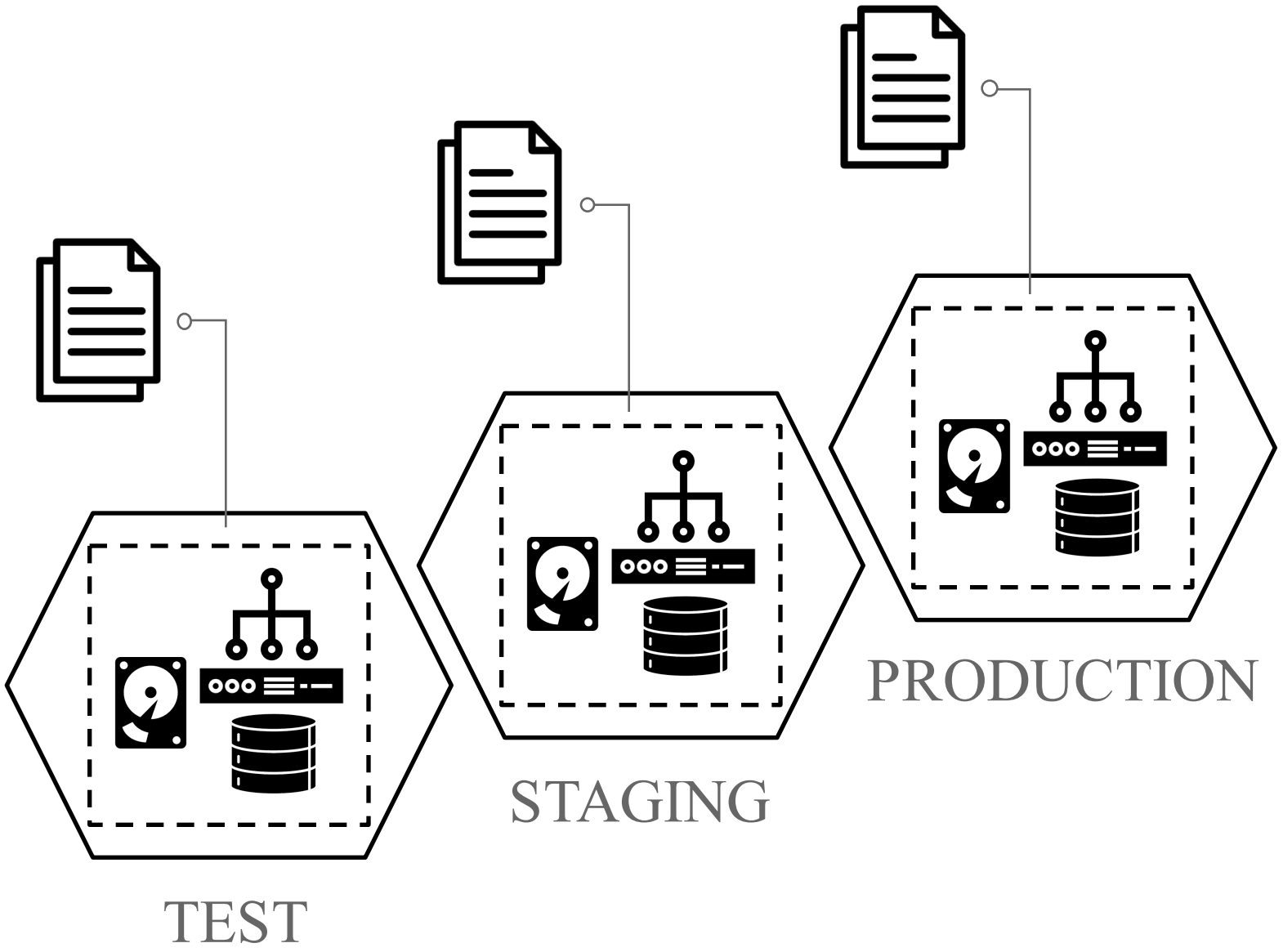


MULTI-ENVIRONMENT STACK

ANTIPATTERN

The **blast radius** for a change includes all of the environments

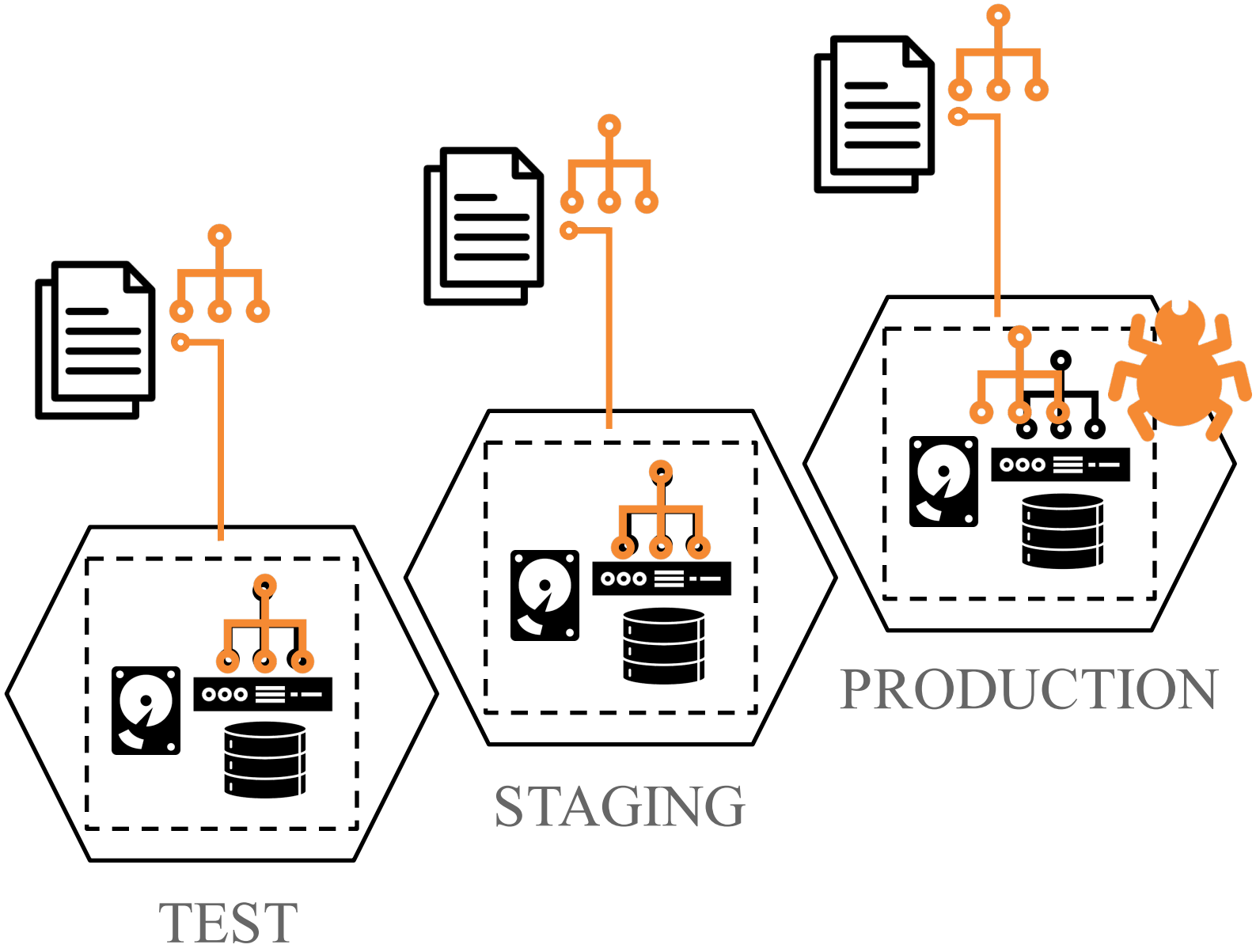




COPY-PASTE ENVIRONMENTS

ANTIPATTERN

- A separate stack code project for each environment



COPY-PASTE ENVIRONMENTS

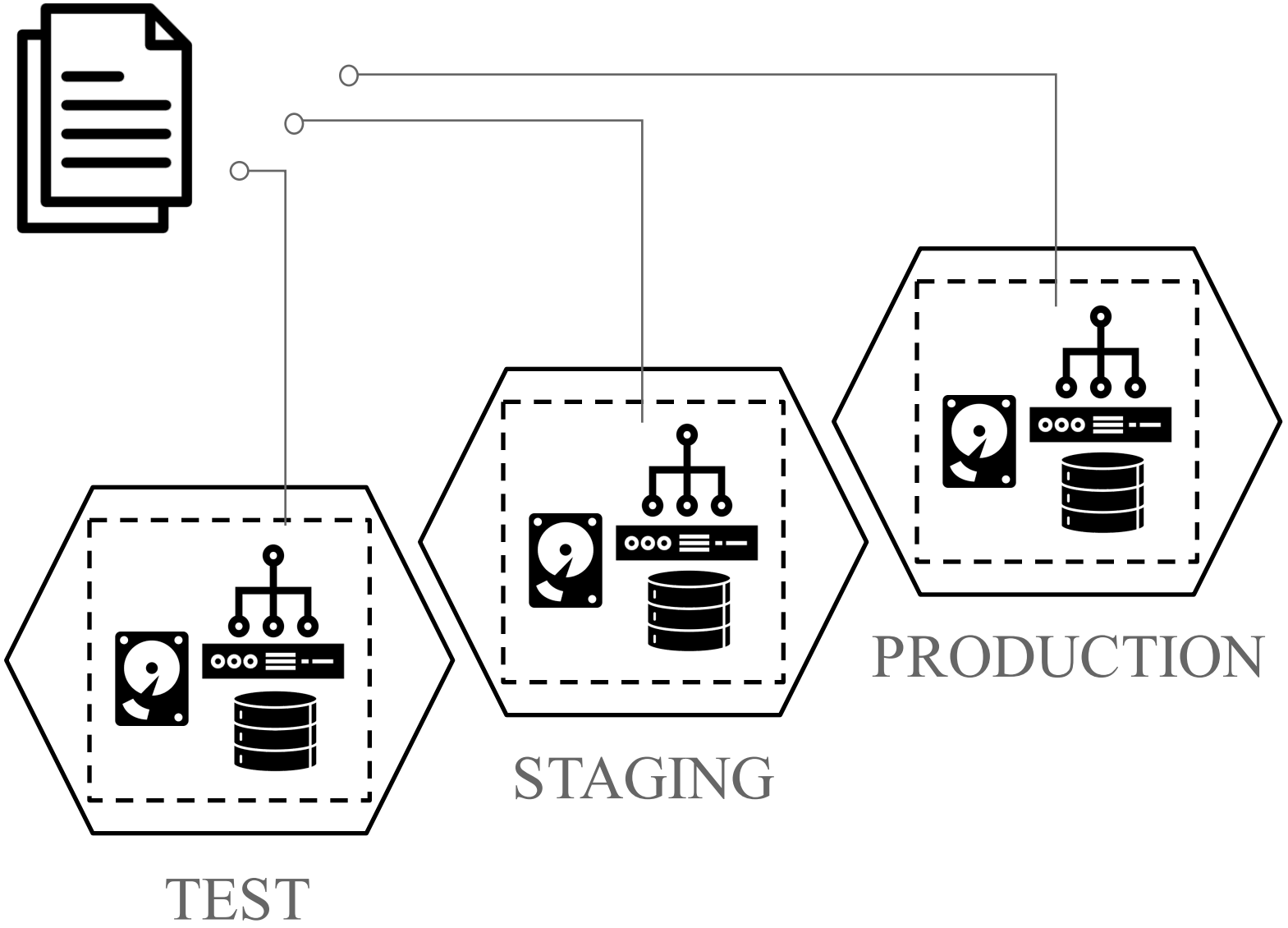
ANTIPATTERN

- Changes are copied and pasted from one environment stack to the next
- Creates opportunity for error

RE-USABLE STACK

PATTERN

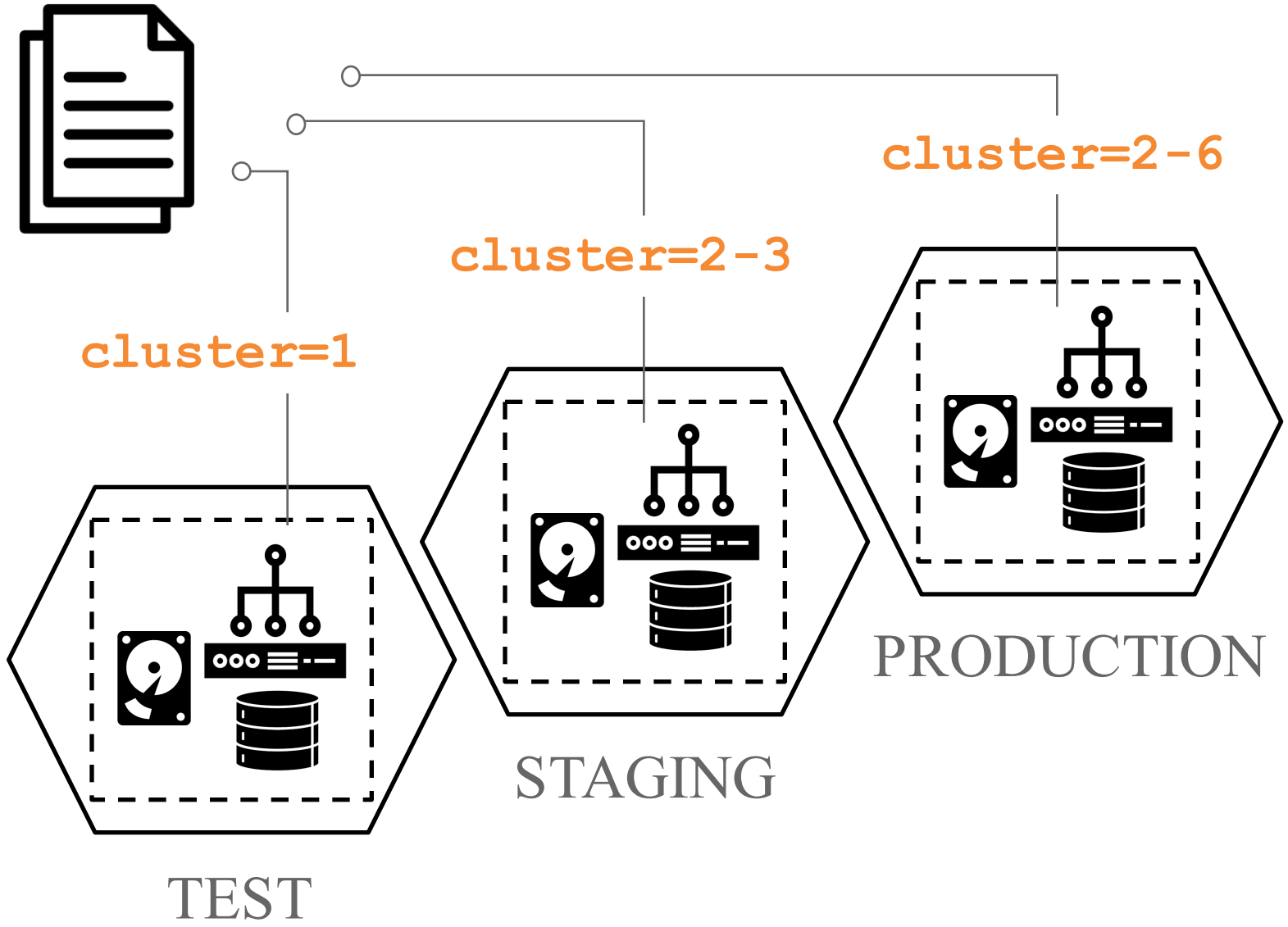
- Single stack project code is applied to each environment in turn



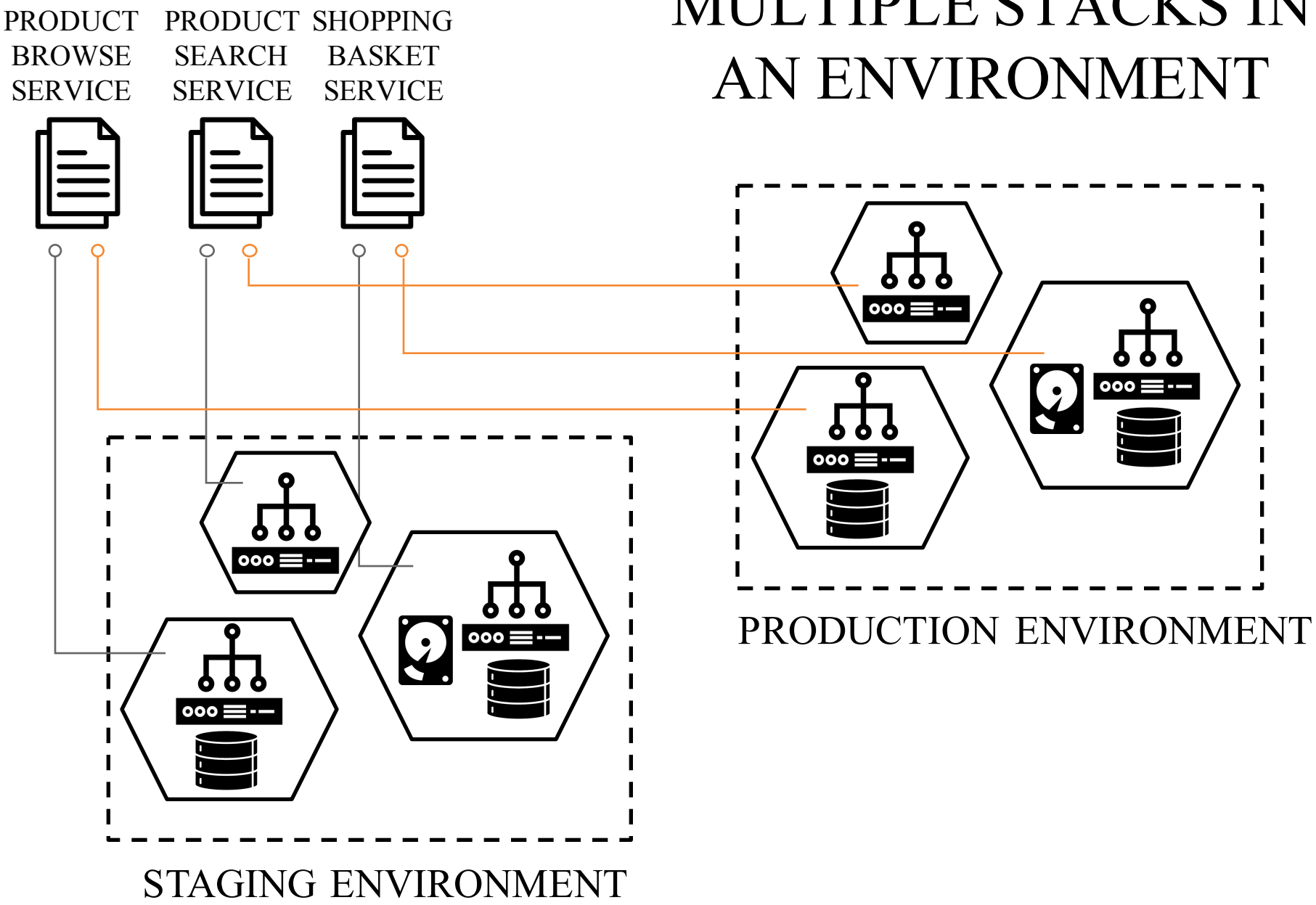
RE-USABLE STACK

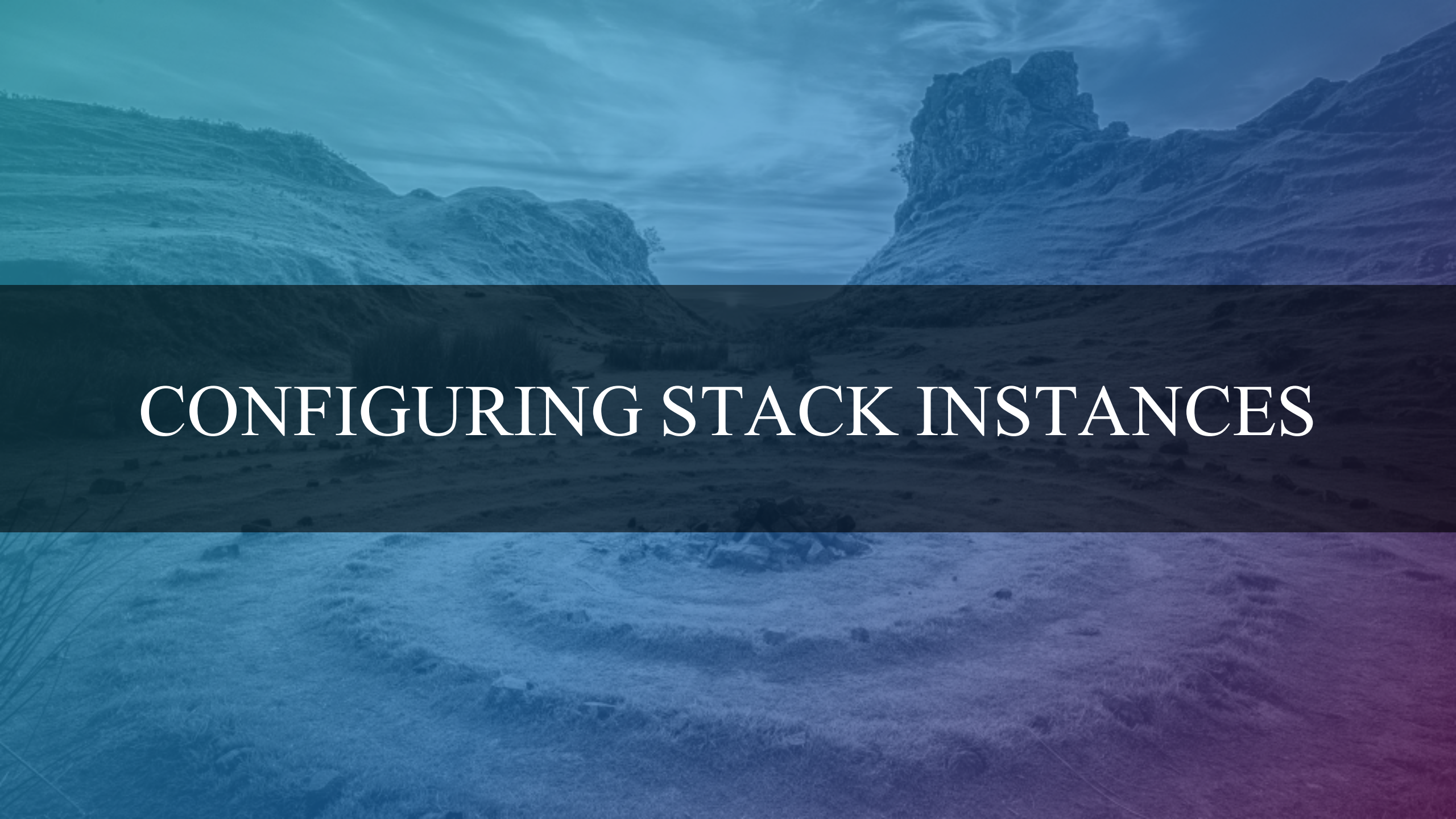
PATTERN

- Use parameters to customize each instance



MULTIPLE STACKS IN AN ENVIRONMENT





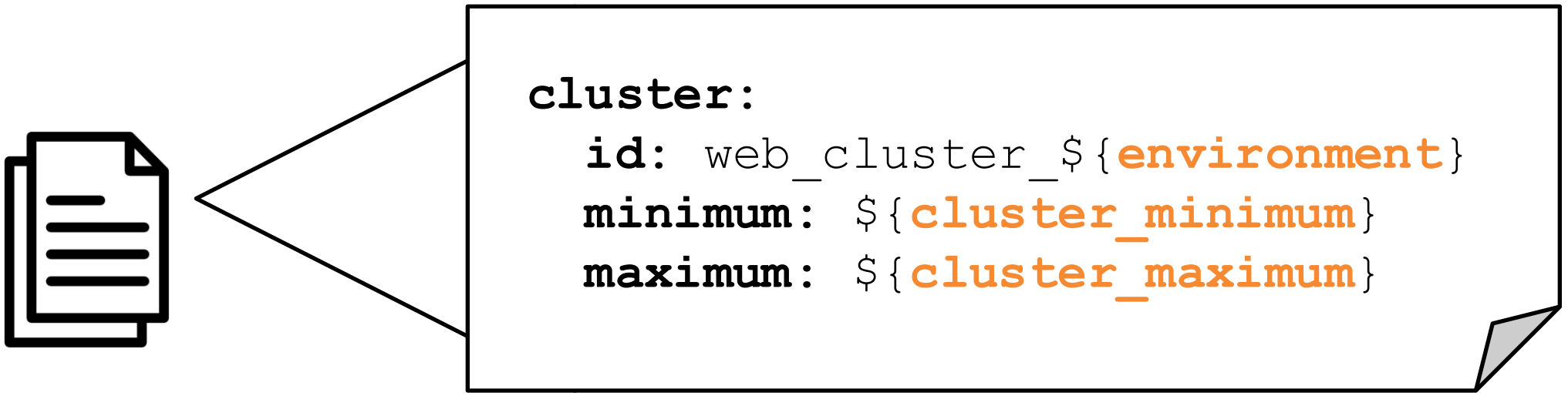
CONFIGURING STACK INSTANCES

GOALS

WHY

- Support variations between environments
- Name and tag resources by environment
- Avoid resource name and ID clashes

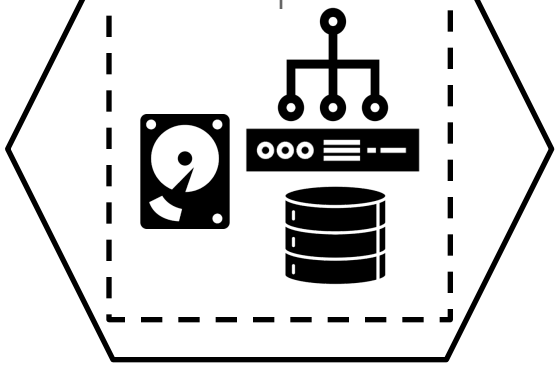
AN EXAMPLE OF STACK PARAMETERS



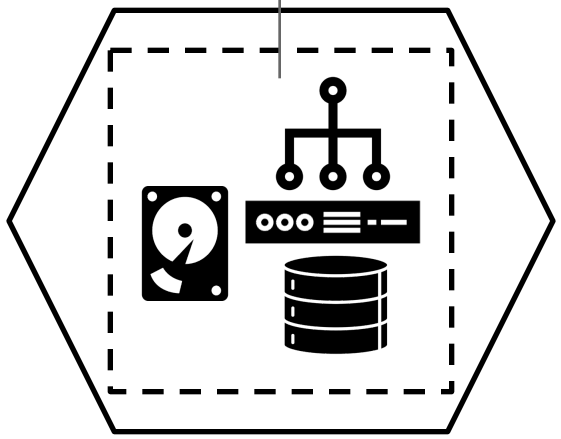
```
environment      = "production"
cluster_minimum  = 2
cluster_maximum  = 5
```

```
environment      = "staging"
cluster_minimum  = 2
cluster_maximum  = 3
```

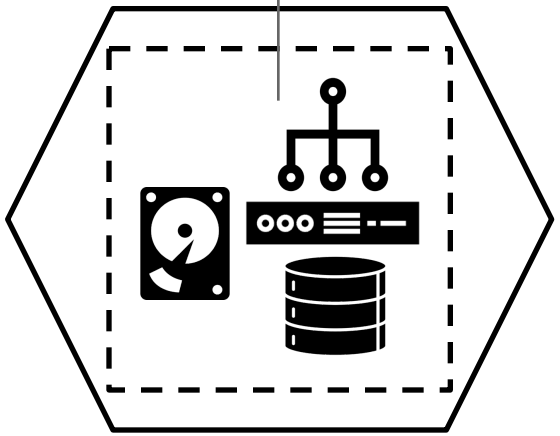
```
environment      = "test"
cluster_minimum  = 1
cluster_maximum  = 1
```



TEST



STAGING



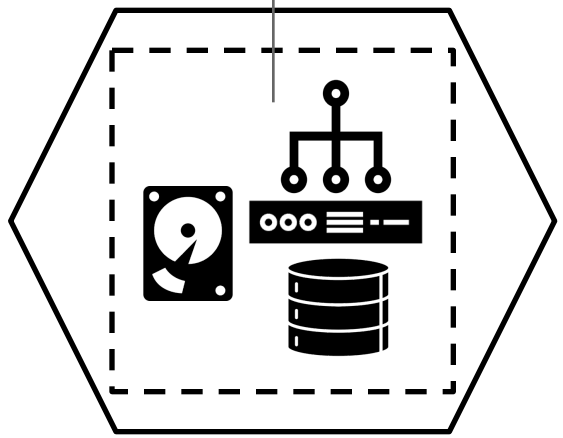
PRODUCTION

MANUAL STACK PARAMETERS

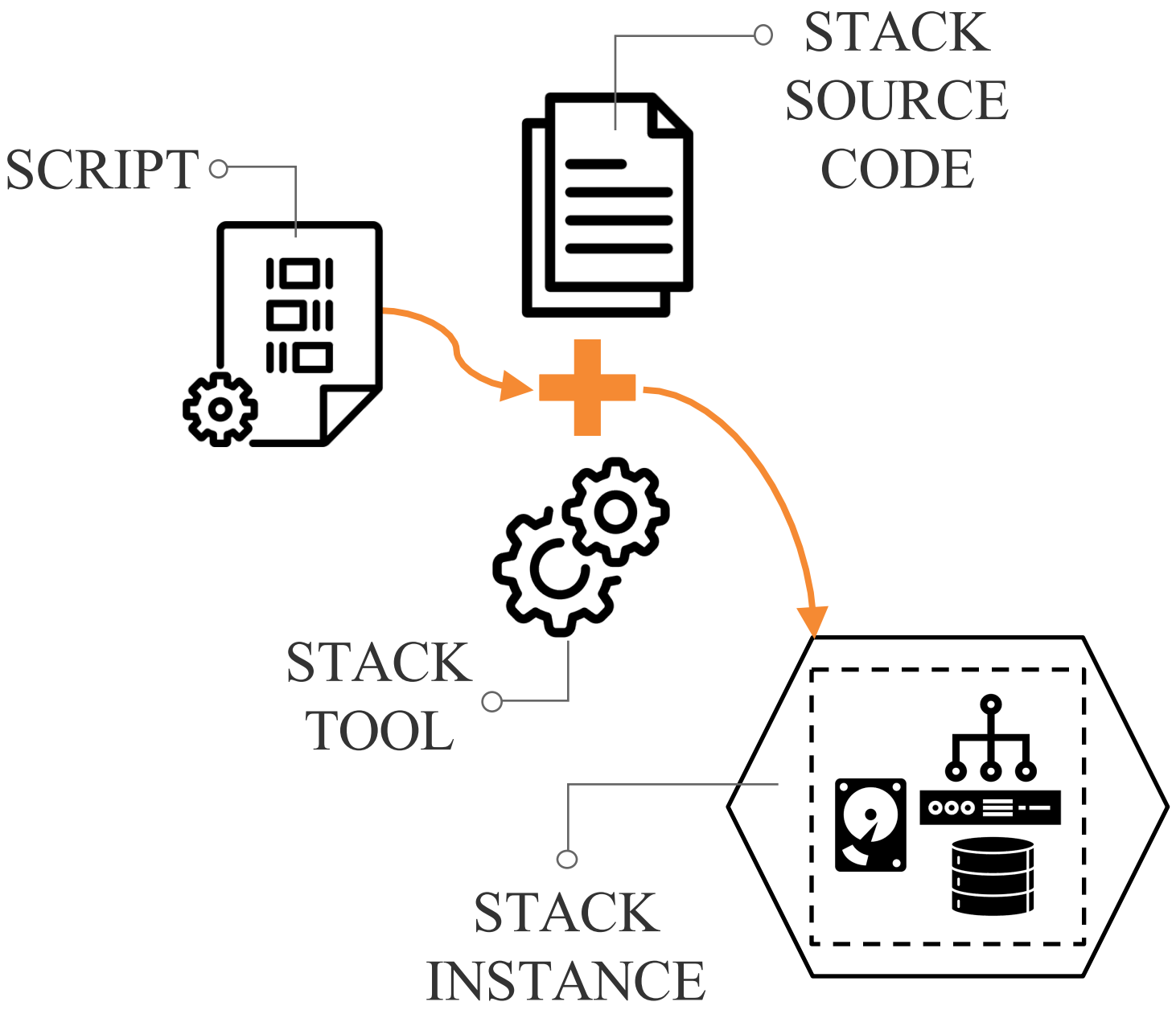
ANTIPATTERN

- Difficult to reproduce
- Difficult to maintain
- Risky

```
stack up \
  environment=test \
  cluster_maximum=1
```



TEST



SCRIPTED STACK PARAMETERS

PATTERN

- A script runs the stack tool
- The parameters hard-coded in the script

```
if $ENV = "test"
  stack up cluster_maximum=1 env=$ENV
elsif $ENV = "staging"
  stack up cluster_maximum=3 env=$ENV
elsif $ENV = "production"
  stack up cluster_maximum=5 env=$ENV
end
```

SCRIPTED STACK PARAMETERS

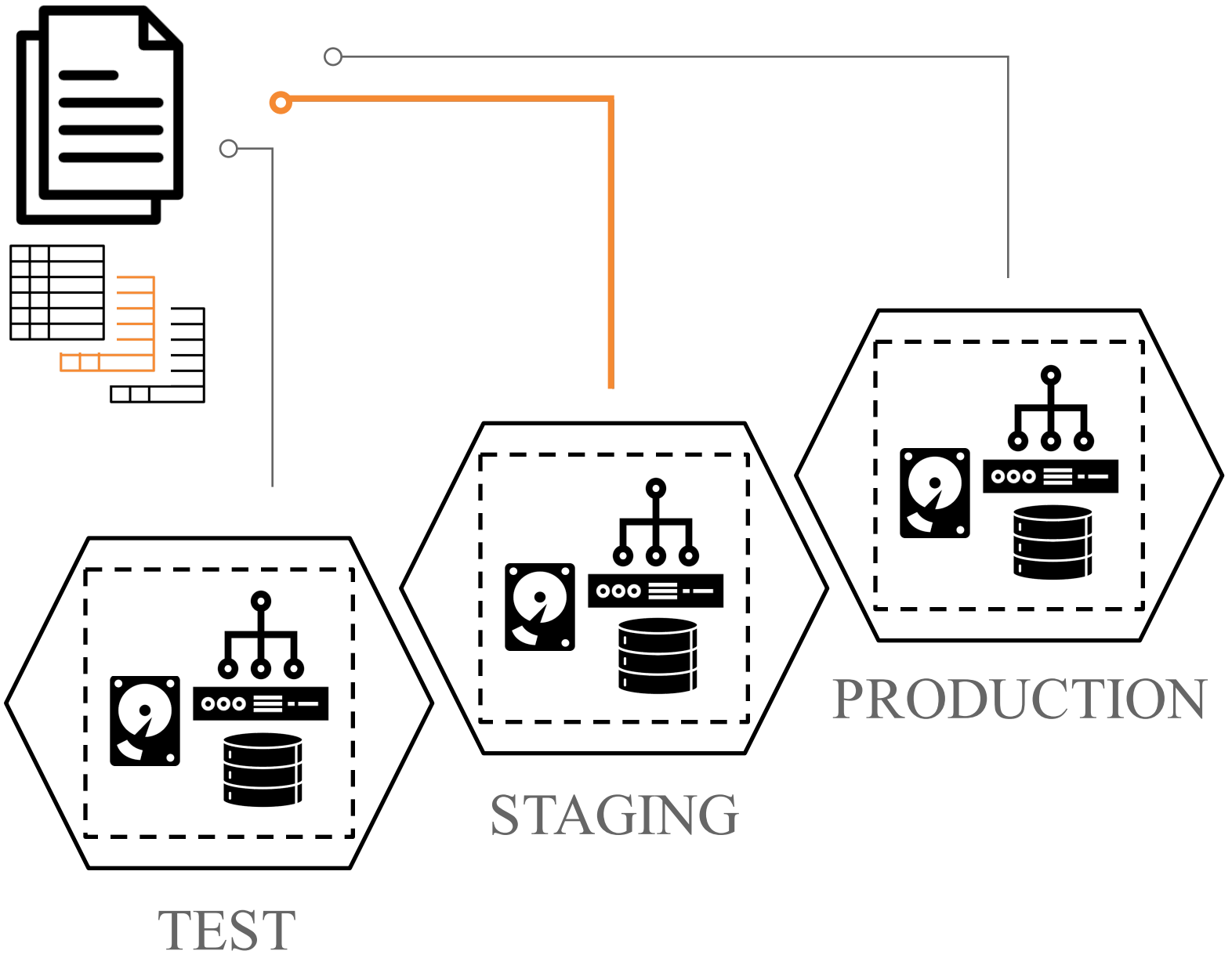
PATTERN

- Easier to reproduce
- Difficult to maintain

STACK CONFIGURATION FILES

PATTERN

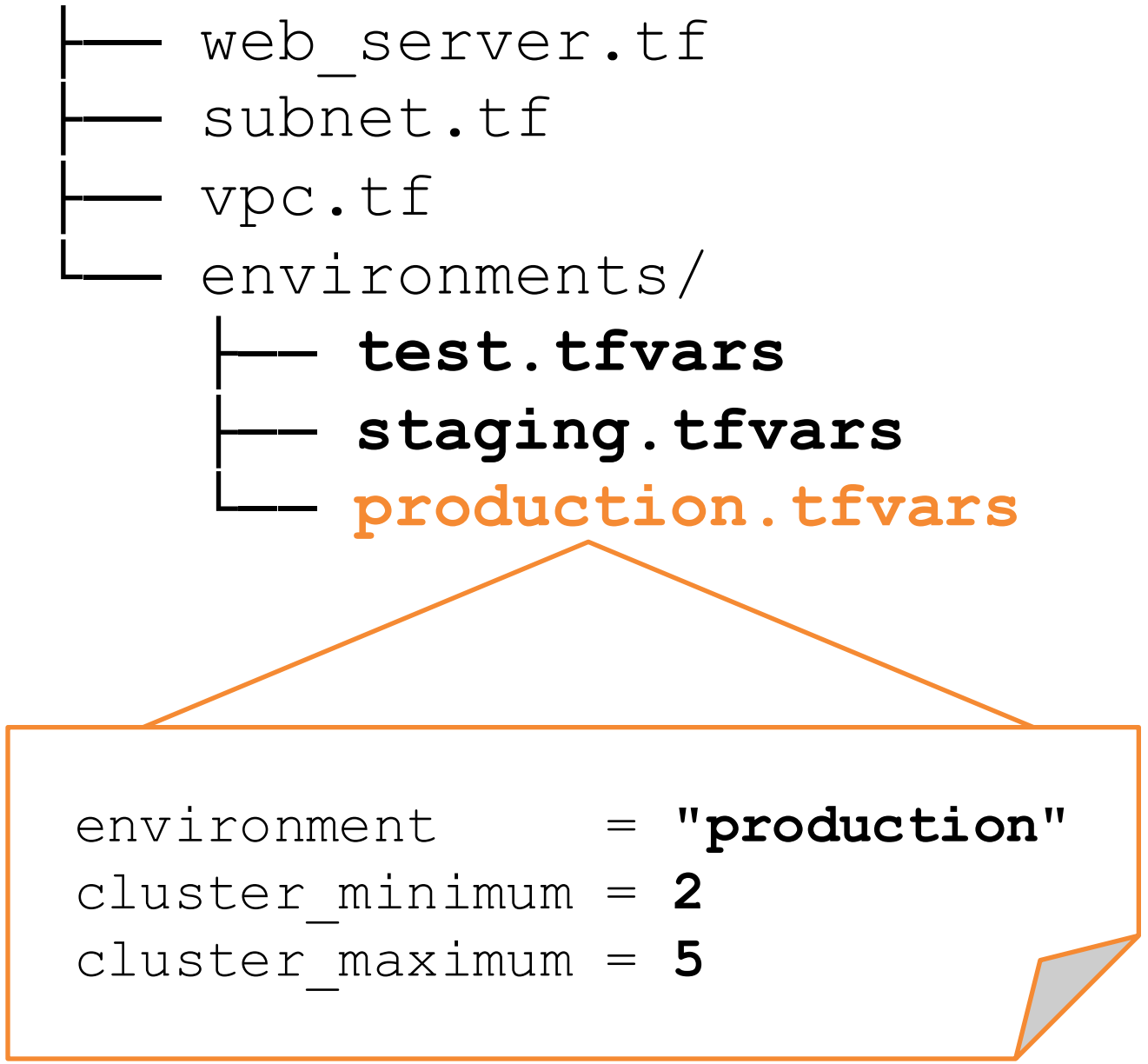
- Separate configuration file per environment
- Files checked into source control with stack code

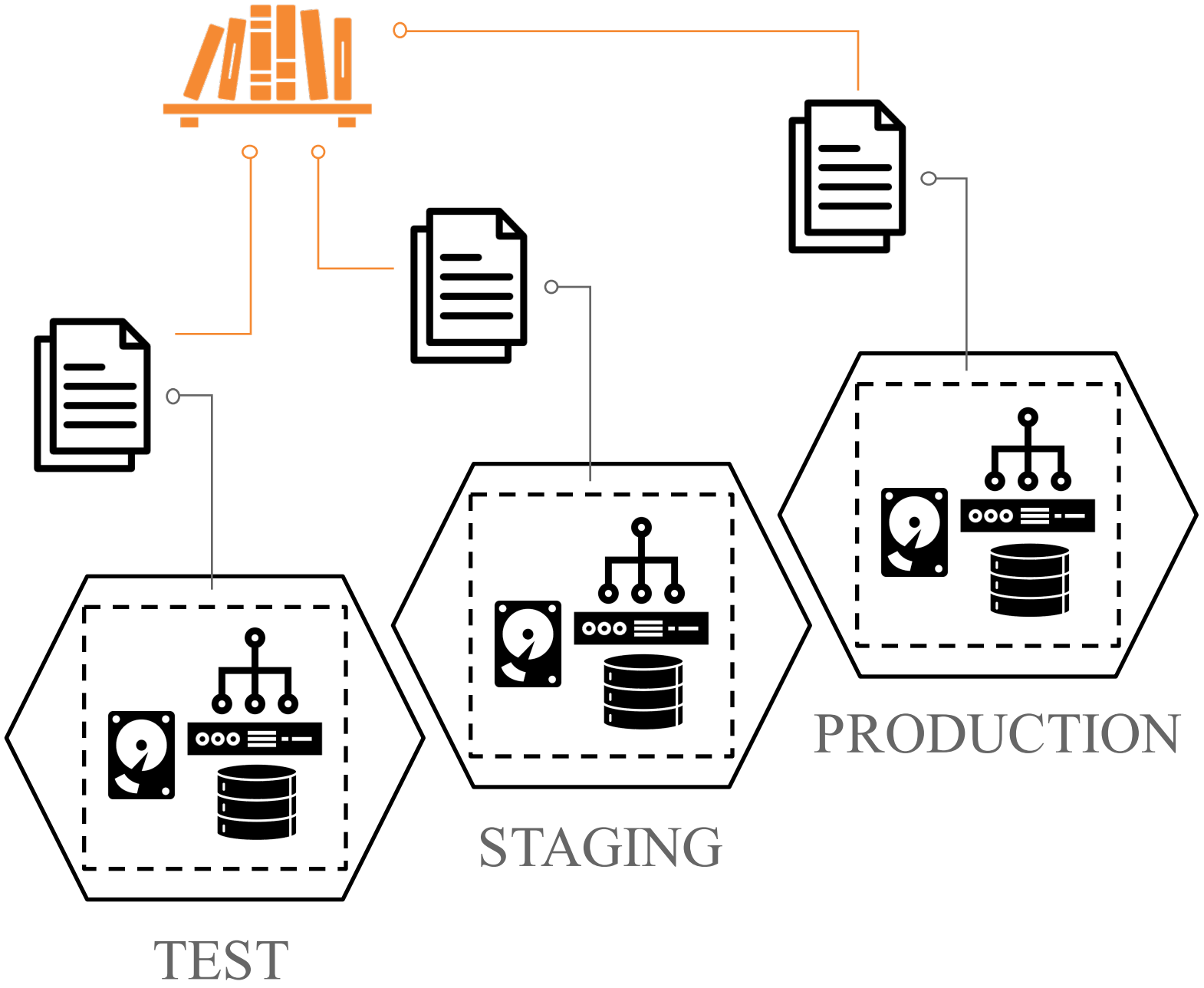


STACK CONFIGURATION FILES

PATTERN

- Good maintainability and reproducibility
- You need to change the source code to change an environment





WRAPPER STACK

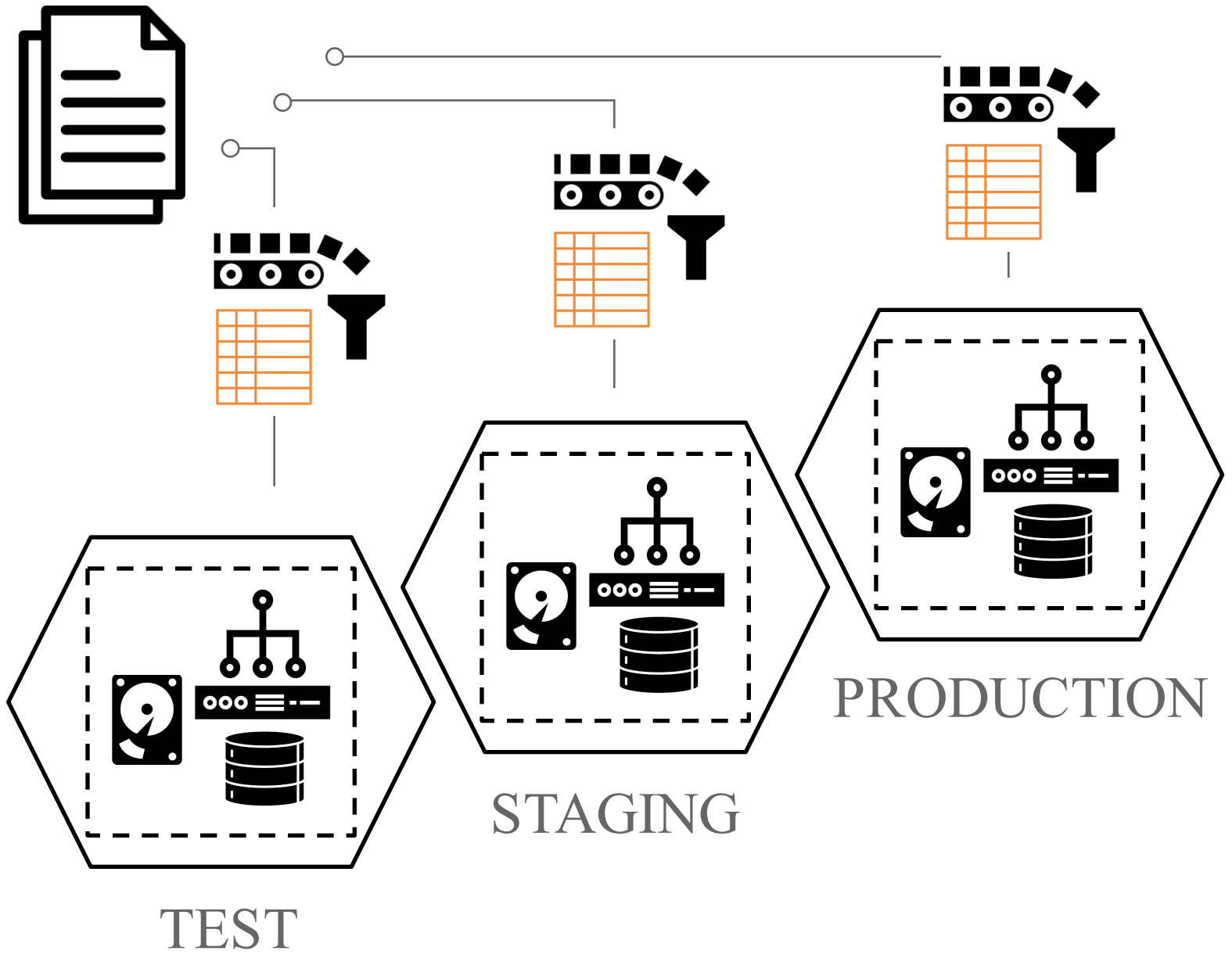
PATTERN

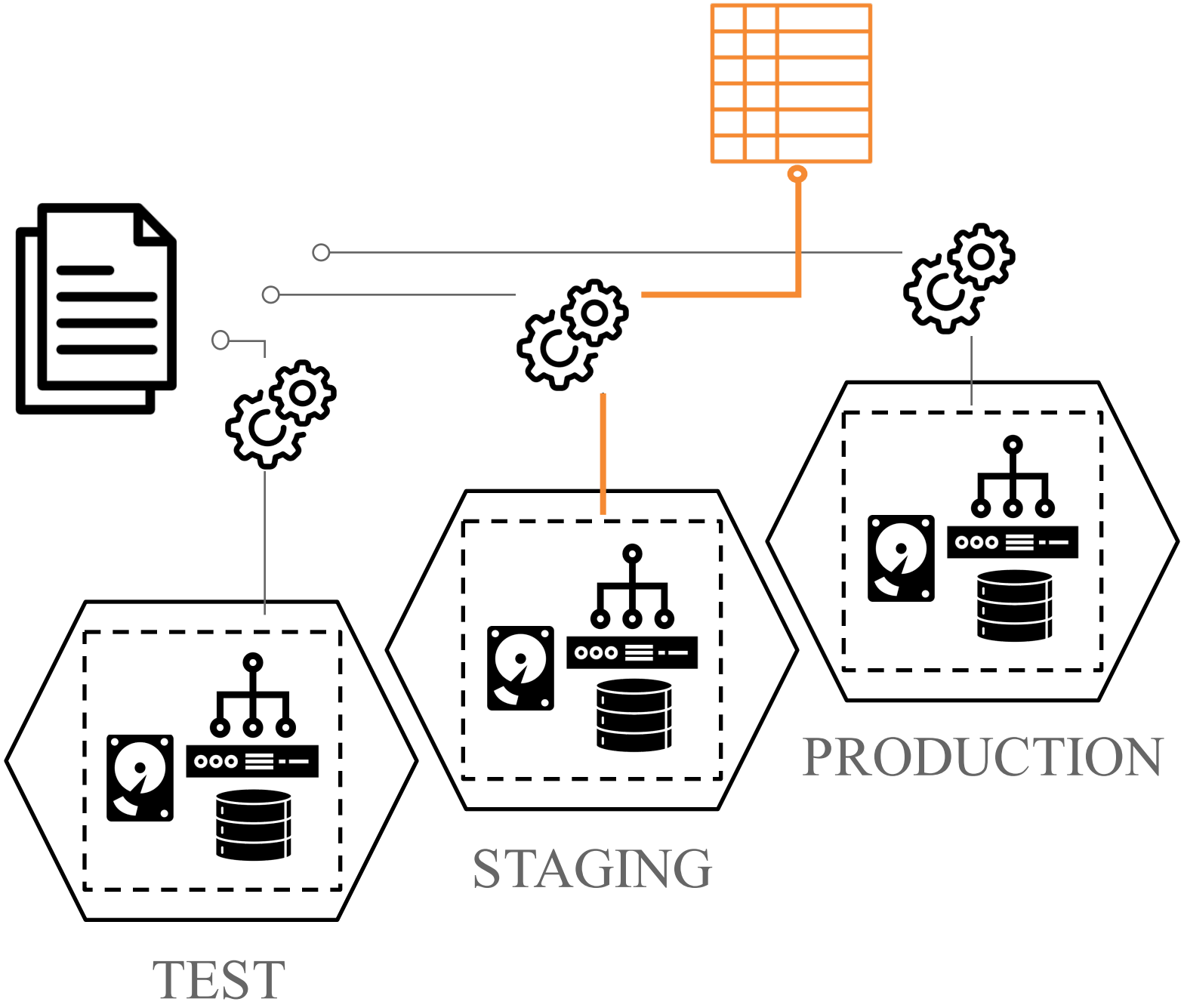
- Each instance has a separate stack code project with configuration
- The wrapper project imports a library that defines the stack

PIPELINE STACK PARAMETERS

ANTIPATTERN

- Pipeline stage or CI job includes environment configuration values
- Couples provisioning to the pipeline or CI tool





STACK PARAMETER REGISTRY

PATTERN

- Values for each instance are stored in a central location
- Stack tool retrieves those values
- Maintains code and configuration apart


SECRETS

Encrypted secrets in source

Who has the secret that decrypts the secrets?

- git-crypt
- blackbox
- sops
- transcrypt

Inject Secrets at Runtime

Apply stack instance parameter patterns

- Local secrets file
- Environment variables
- Secrets management service

Secretless Authorization

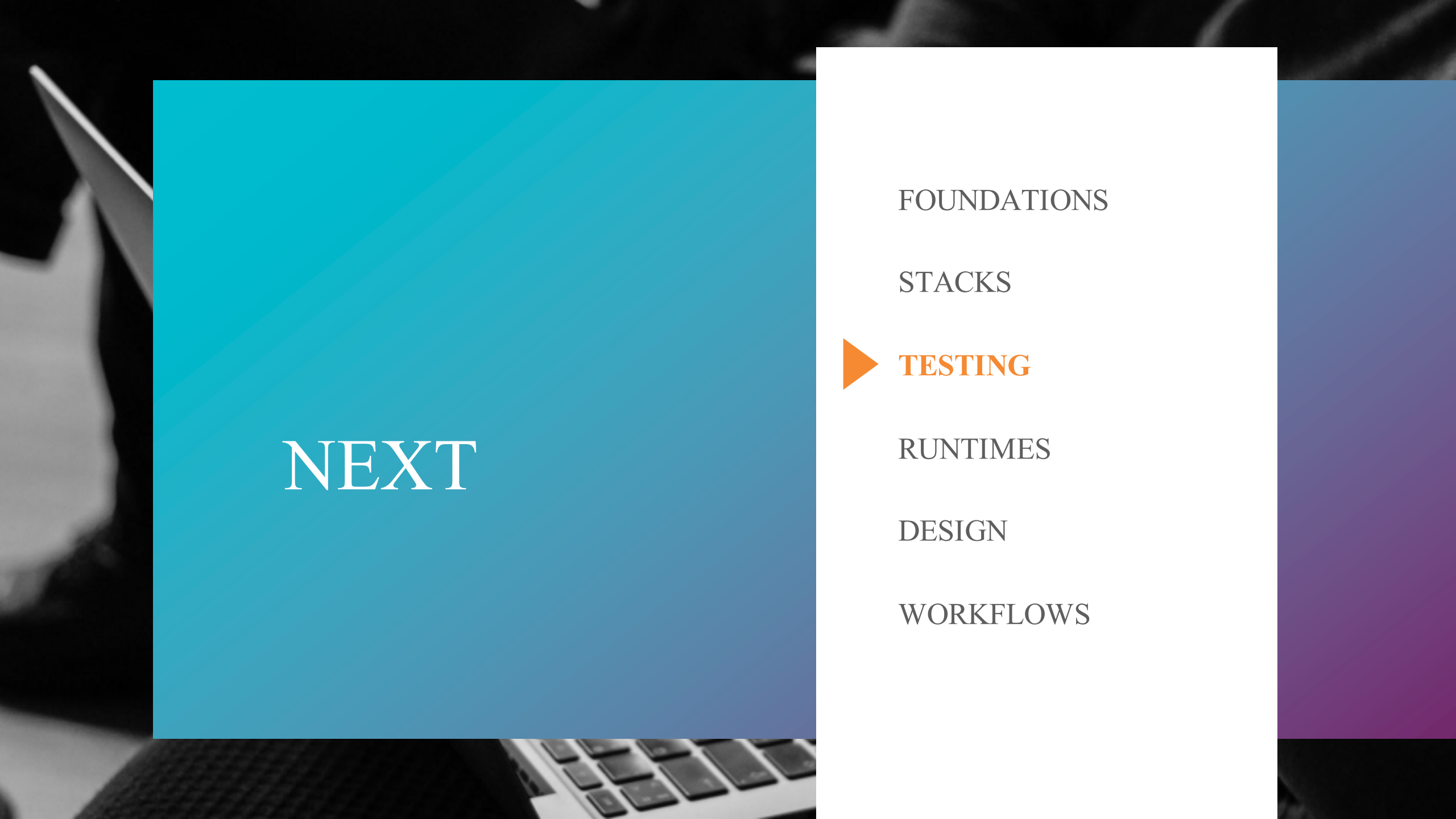
Platform handles it for you

- For example, AWS IAM Profiles

Disposable Secrets

Create and assign on the fly

- Useful for testing
- Connections between services
- Vault and similar services can support this



NEXT

- FOUNDATIONS
- STACKS
- **TESTING**
- RUNTIMES
- DESIGN
- WORKFLOWS