Modernizing Legacy Applications with Habitat

Tom Finch
Solutions Architect
Chef Software

tfinch@chef.io

CHEF™

All about applications...

Every company is a software company...

Building, managing and deploying software is painful!

Why?

Environments

Why?

Environments

Why?

Environments

Virtual Machines

Containers,k8s, Openshift

Why?

Environments

Virtual Machines	**Containers,k8s, Openshift**	**Bare Metal**

Why?

Environments

Why?

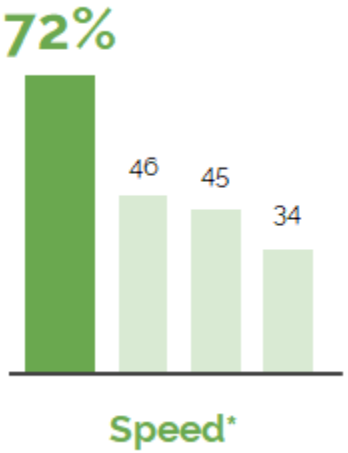
Environments

Each environment requires us to build, deploy and manage in a different way

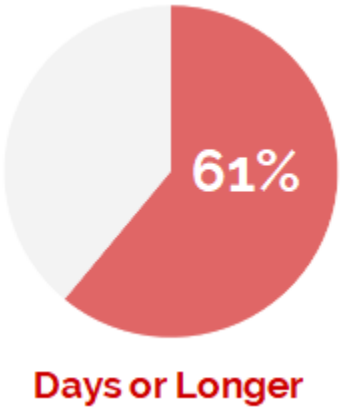
Application Automation allows businesses to build, manage, and deploy software with less pain and more power...

Chef's 2018 'State of Application Delivery' survey

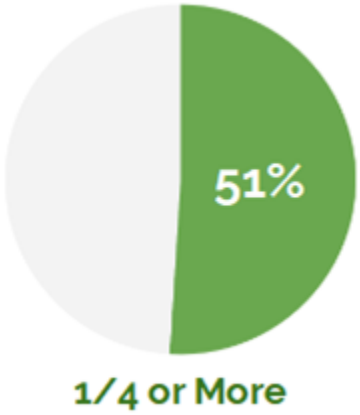
How do you measure app deployment success?



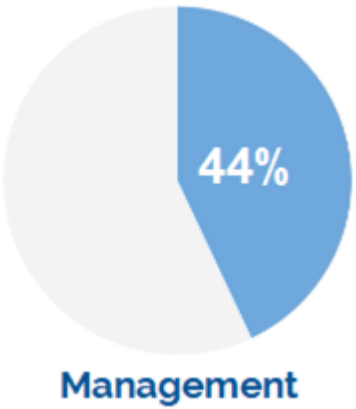
How long does it take to complete the app build process?



In 2 years, what % of your apps will be deployed on container platforms?



Which is the most challenging aspect of the application lifecycle?



347 application and operations professionals participated

Introducing..

What is Habitat?

At its core:

- A technology for packaging software
- A technology for running software

What does Habitat bring to the Enterprise?

What does an Enterprise look like today?

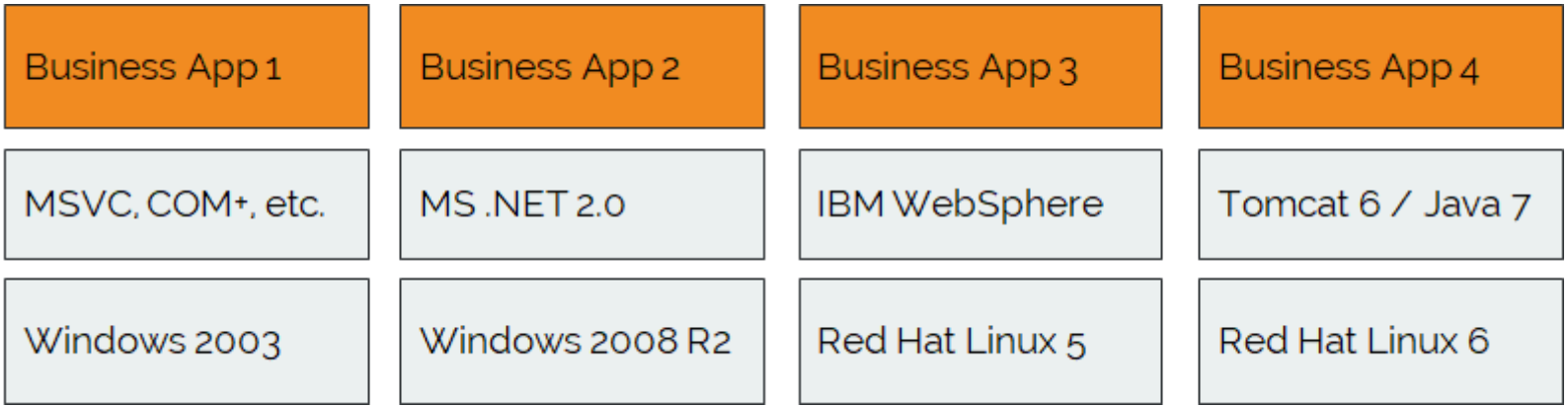
- Hundreds or thousands of applications (both internally written and off the shelf) that must be managed
- Dependencies of these hundreds or thousands of applications also must be managed
- Most of these applications are **legacy apps**

A **legacy application**
is a **critical business application** that
is **long lived**

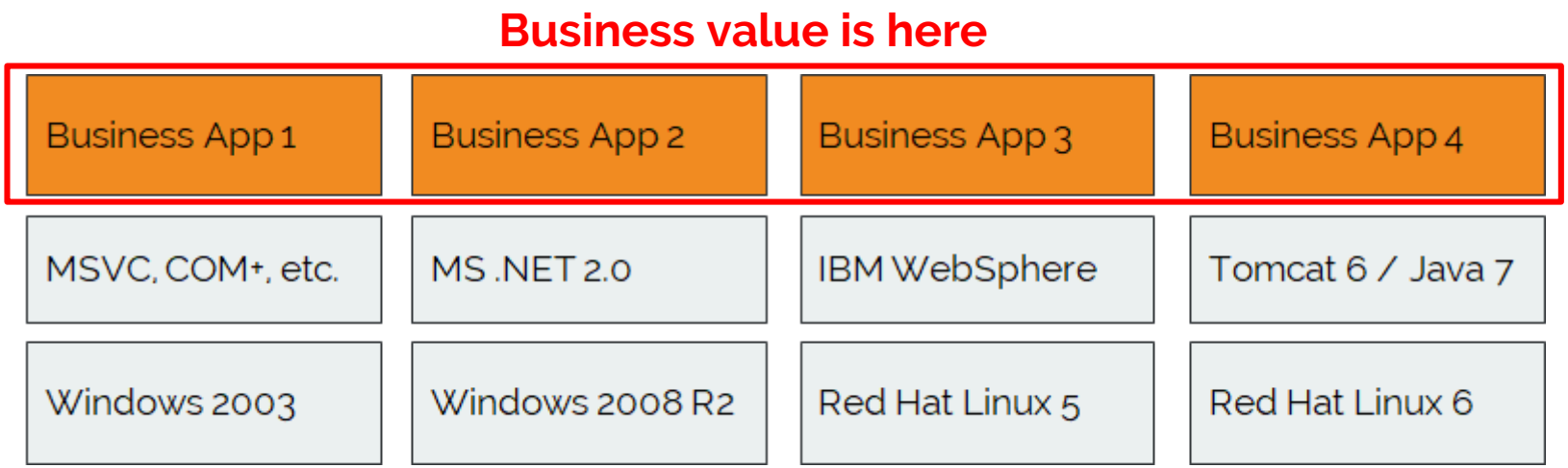
Why is it so painful?

- Maintaining software
  - Keeping your application and its dependencies up to date quickly becomes unmanageable
- Adapting to new platforms
  - Migrating software to the cloud is difficult and costly
  - Application needs change over time, and infrastructure must evolve
- Supporting old technology stacks
  - Adapting to new ways of software delivery and management is not prioritised.
  - Legacy technology must be supported for longer than is necessary

Legacy Applications



Legacy Applications

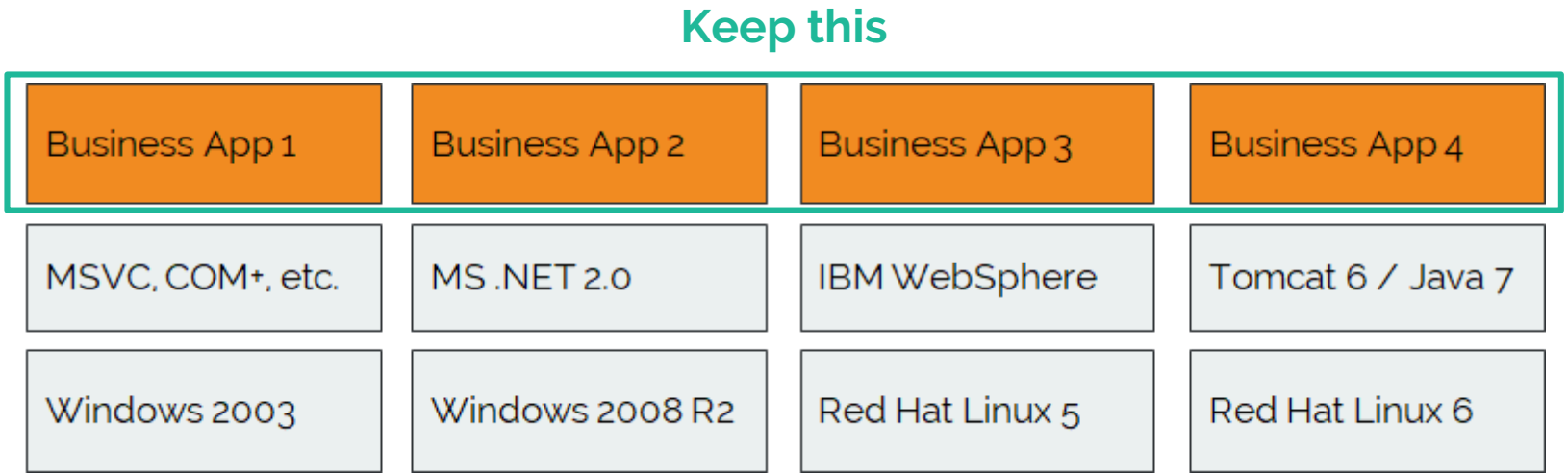


Legacy Applications

Costly burden of support and dependencies to maintain

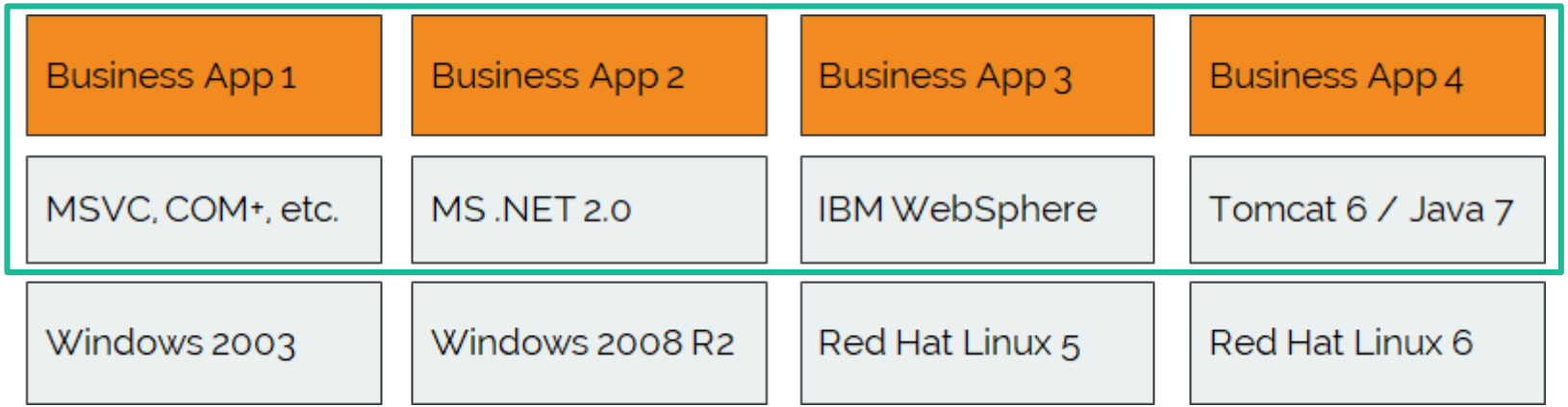
What if?

Legacy Applications



Legacy Applications

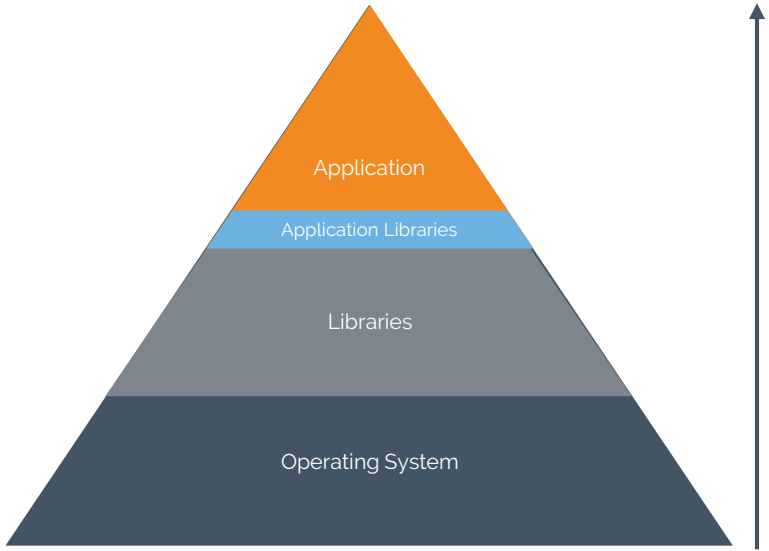
Bundle the bare minimum with the app in one package



Legacy Applications

Remove dependency on this

Legacy Applications - The Old World



- Baggage
- Build, Deploy and Manage differently in each environment
- Coupled to OS
- Updates are hard

Legacy Applications - The New World

- Strict dependencies (no baggage)
- Build, Deploy and Manage the same way
- Decoupled from the OS
- Updates are easy

This gives us
Application portability

Why is it so painful?

- Maintaining software
  - Keeping your application and its dependencies up to date quickly becomes unmanageable
- Adapting to new platforms
  - Migrating software to the cloud is difficult and costly
  - Application needs change over time, and infrastructure must evolve
- Supporting old technology stacks
  - Adapting to new ways of software delivery and management is not prioritised
  - Legacy technology must be supported for longer than is necessary

Application portability relieves this pain

Benefits of Application Portability

- **Agnostic to operating system and operating system version**

Benefits of Application Portability

- Agnostic to operating system and operating system version

- **Allows you to move your application to any infrastructure – regardless of where it ran before**

Application portability
Future proofs applications

Why is it so painful?

- Maintaining software
  - Keeping your application and its dependencies up to date quickly becomes unmanageable
- Adapting to new platforms
  - Migrating software to the cloud is difficult and costly
  - Application needs change over time, and infrastructure must evolve
- Supporting old technology stacks
  - Adapting to new ways of software delivery and management is not prioritised
  - Legacy technology must be supported for longer than is necessary

Application portability also relieves this pain

Containers???

Moving to containers will not solve manageability issues

When you package with Habitat...

- I **know** what is in my containers!
  - Dependencies
  - Easy to audit
  - Happy auditors!

- Consistent configuration interface for **all** services
  - No one-off config or format for any services
  - Runtime lifecycle management(init, run, health check)
  - Service discovery (binds)

Habitat complements container technologies, and makes them easier to work with and switch between

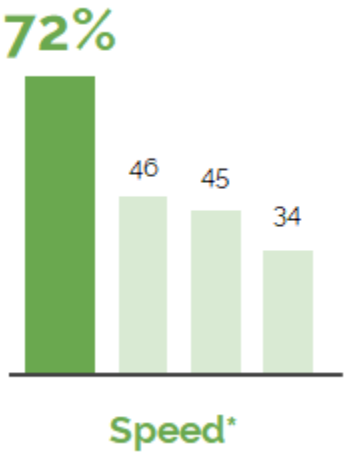
When you package with Habitat...

- Application can be deployed **anywhere**
  - Bare metal
  - Virtual machines
  - Containers
  - Cloud
- Applications can be exported into **any format**
  - Docker
  - Kubernetes
  - Cloud Foundry
  - Mesosphere
  - ...many more!
- As the application evolves, the infrastructure can **easily change**

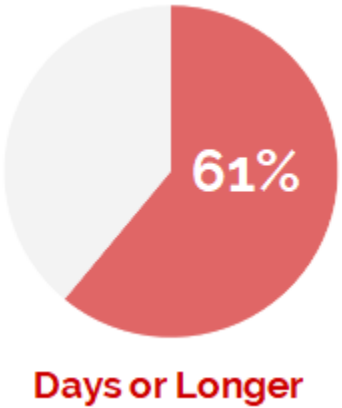
Chef's 2018 'State of Application Delivery' survey

Habitat can help enable this

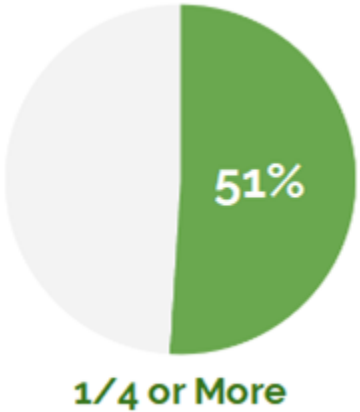
How do you measure app deployment success?



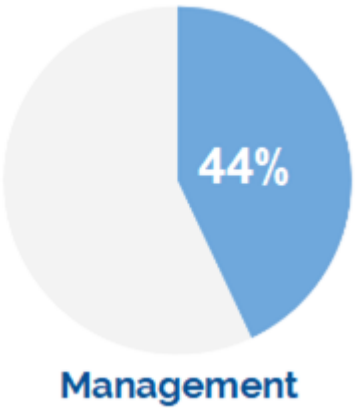
How long does it take to complete the app build process?



In 2 years, what % of your apps will be deployed on container platforms?



Which is the most challenging aspect of the application lifecycle?



347 application and operations professionals participated

How it works...

Build, Deploy, Manage

Build

plan.sh

```
pkg_name=national-parks
pkg_description="A sample JavaEE Web app deployed in the Tomcat8 package"
pkg_origin=devopslifter
pkg_version=6.3.0
pkg_maintainer="Tom Finch <tfinch@chef.io>"
pkg_license=('Apache-2.0')
pkg_deps=(core/tomcat8 core/jre8)
pkg_build_deps=(core/jdk8/8u131 core/maven)
pkg_svc_user="root"
pkg_binds=(
 [database]="port"
)
pkg_exports=(
 [port]=tomcat_port
)
pkg_exposes=(port)
```

Plan.sh (continued)

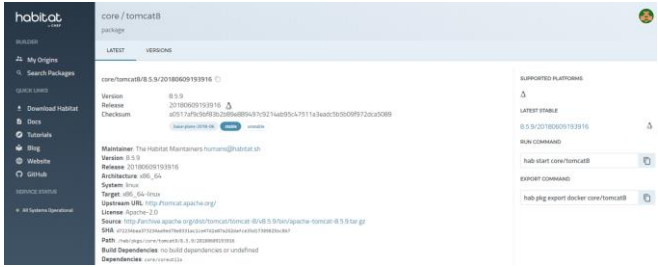
```
do_build()
{
   cp -r $PLAN_CONTEXT/../ $HAB_CACHE_SRC_PATH/$pkg_dirname
   cd ${HAB_CACHE_SRC_PATH}/${pkg_dirname}
   mvn package
}

do_install()
{
   cp ${HAB_CACHE_SRC_PATH}/${pkg_dirname}/target/${pkg_name}.war ${PREFIX}/
}
```

Deploy

Deploying software with **Habitat**

Deploying software with **Habitat**

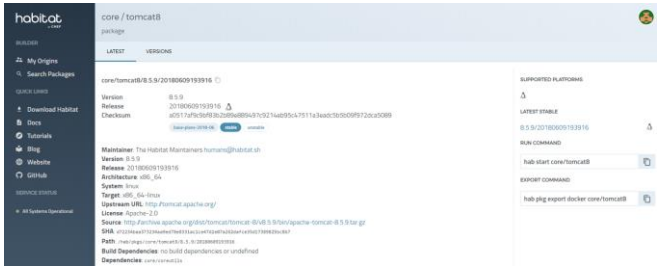
Habitat pulls
Package + deps
From Builder Depot



Place to run software

Deploying software with **Habitat**

Habitat pulls
Package + deps
From Builder Depot



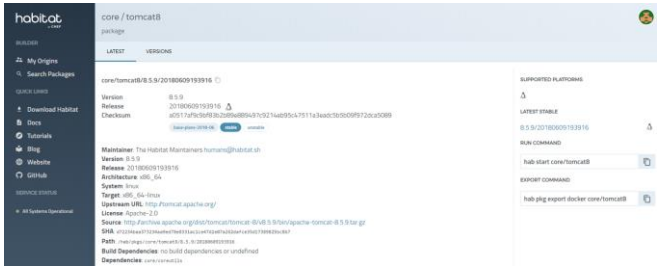
Place to run software

Habitat start
Habitat Supervisor

Habitat Supervisor

Deploying software with **Habitat**

Habitat pulls
Package + deps
From Builder Depot



Place to run software

Habitat start
Habitat Supervisor

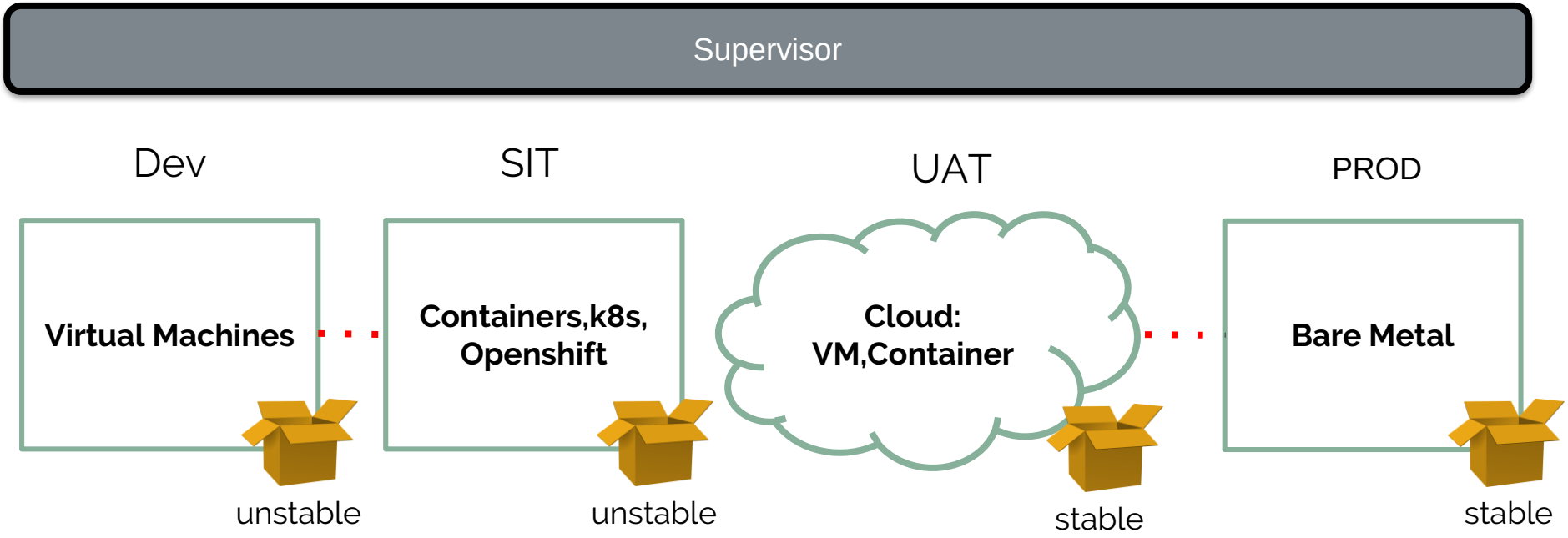
Habitat Supervisor

Habitat unpacks,
initialises and runs
packages software

Manage

No matter where an application packaged with Habitat is running, it's **configuration is managed the same way**

Managing software with **Habitat**



Managing software with **Habitat**

- **Habitat artifacts remain immutable in every environment**

Managing software with **Habitat**

- Habitat artifacts remain immutable in every environment
- **Habitat manages config and bindings specific to environment**

Managing software with **Habitat**

- Habitat artifacts remain immutable in every environment
- Habitat manages config and bindings specific to environment
- **Automation travels with the app - Shift left!**

Show me!

All about applications...

Habitat gives one consistent, testable, portable and **immutable** way to build, deploy and manage your applications.

There is no one true way to develop applications

It depends on the business
and application's needs
at that particular time

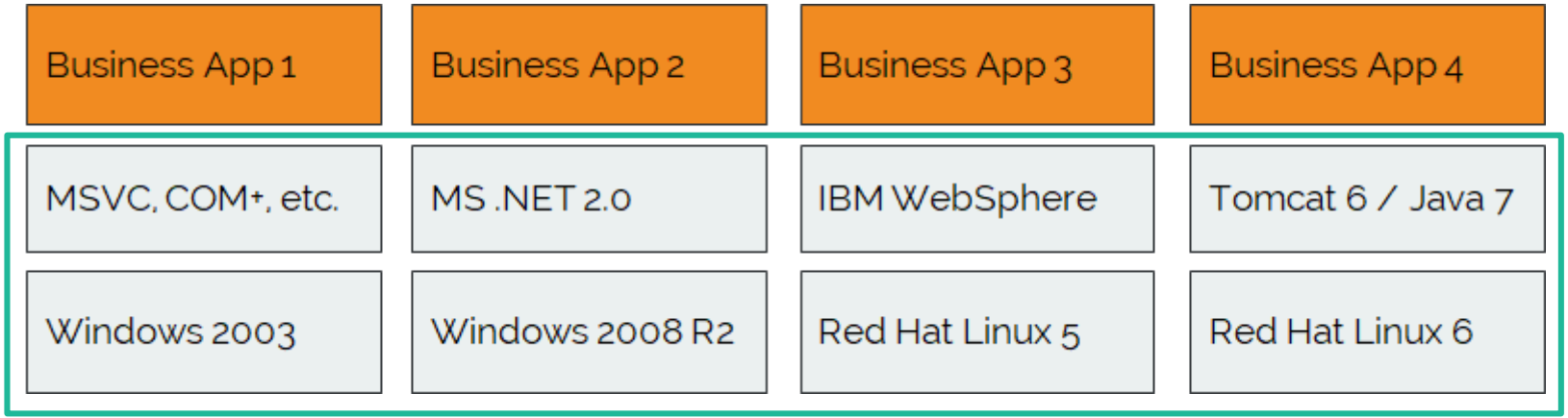
Those needs will change throughout the life of the application and the business

Legacy Applications

Focus here - Value!

Business App 1	Business App 2	Business App 3	Business App 4
MSVC, COM+, etc.	MS .NET 2.0	IBM WebSphere	Tomcat 6 / Java 7
Windows 2003	Windows 2008 R2	Red Hat Linux 5	Red Hat Linux 6

Legacy Applications



Let Habitat handle this!

Learn more about Habitat

- www.habitat.sh
- www.learn.chef.io
- www.slack.habitat.sh
- Come and see me today!
- Talk to us! - tfinch@chef.io

CHEF CONF 2019

London | June 19-20

AUTOMATE ALL THE THINGS.
CHEFCONF 2019 IS HERE.

chefconf.chef.io

CHEF™