***Created By : Vikram Khot***

Contents

[Types of Cloud Computing 3](#_Toc483190967)

[public cloud 4](#_Toc483190968)

[Private cloud 4](#_Toc483190969)

[Hybrid cloud 4](#_Toc483190970)

[Cloud Service 6](#_Toc483190971)

[1. Software as a Service (SaaS 7](#_Toc483190972)

[2. Platform as a Service (PaaS). 7](#_Toc483190973)

[3. Infrastructure as a Service (IaaS). 8](#_Toc483190974)

[10 most powerful PaaS companies 8](#_Toc483190975)

[1. Amazon Web Services 8](#_Toc483190976)

[2. Salesforce.com 8](#_Toc483190977)

[3. Long Jump 8](#_Toc483190978)

[4. Microsoft(Windows Azur) 8](#_Toc483190979)

[5. IBM (IBMSmartCloud) 8](#_Toc483190980)

[6. RedHat (OpenShift) 9](#_Toc483190981)

[7. Cloudy Foundry 9](#_Toc483190982)

[8. Google( GoogleAppEngine) 9](#_Toc483190983)

[9. CloudBees 9](#_Toc483190984)

[10. Engine Yard 9](#_Toc483190985)

[What is Cloud Foundry? 10](#_Toc483190986)

[Elastic Runtime 10](#_Toc483190987)

[Application Lifecycle Management 11](#_Toc483190988)

[Application Execution 11](#_Toc483190989)

[Networking & Configuration 11](#_Toc483190990)

[Logging & Monitoring 11](#_Toc483190991)

[Platform Services 11](#_Toc483190992)

[BOSH 11](#_Toc483190993)

[Cloud Foundry Certified Platforms 12](#_Toc483190994)

[Huawei FusionStage 12](#_Toc483190995)

[IBM Bluemix Cloud Foundry 12](#_Toc483190996)

[Pivotal Cloud Foundry 12](#_Toc483190997)

[SAP Cloud Platform 12](#_Toc483190998)

[Swisscom Application Cloud 12](#_Toc483190999)

[What Is Cloud Foundry? 13](#_Toc483191000)

[Section 6: Cloud Foundry Commands 66](#_Toc483191001)

# [Types of Cloud Computing](http://www.globaldots.com/cloud-computing-types-of-cloud/)

Based on a cloud location, we can classify cloud as:

* public,
* private,
* hybrid
* community cloud

Based on a service that the cloud is offering, we are speaking of either:

* [IaaS](http://www.wikinvest.com/concept/Cloud_Computing) (Infrastructure-as-a-Service)
* [PaaS](http://en.wikipedia.org/wiki/Platform_as_a_service) (Platform-as-a-Service)
* [SaaS](http://www.wikinvest.com/concept/Software_as_a_Service) (Software-as-a-Service)
* or, Storage, Database, Information, Process, Application, Integration, Security, Management, Testing-as-a-service

public cloud, we mean that the whole computing infrastructure is located on the premises of a cloud computing company that offers the cloud service. The location remains, thus, separate from the customer and he has no physical control over the infrastructure.

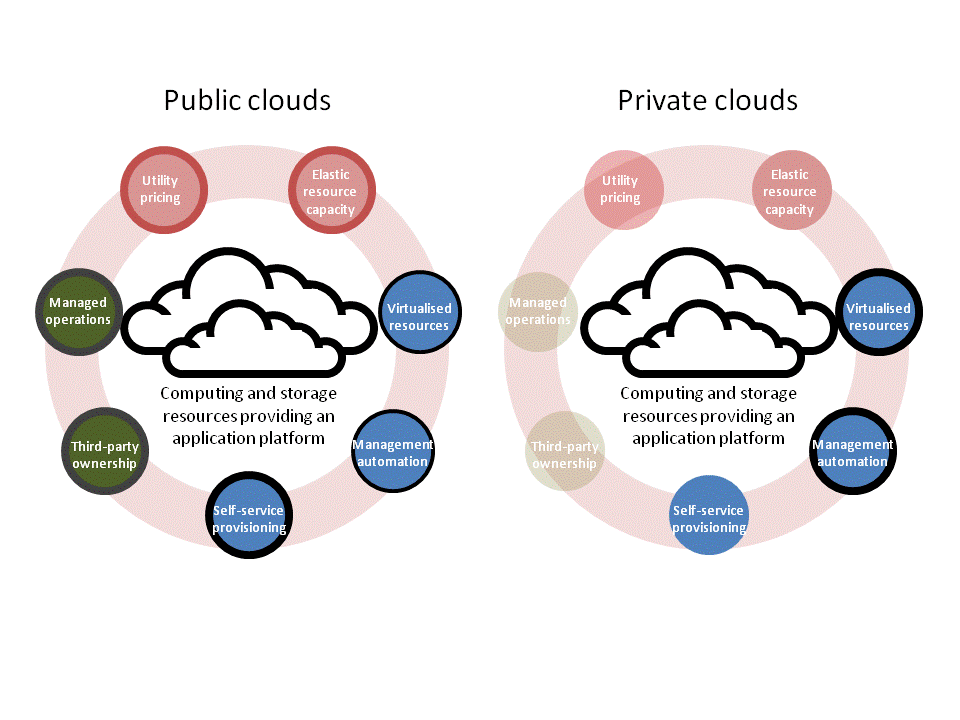
As public clouds use shared resources, they do excel mostly in performance, but are also most vulnerable to various attacks.

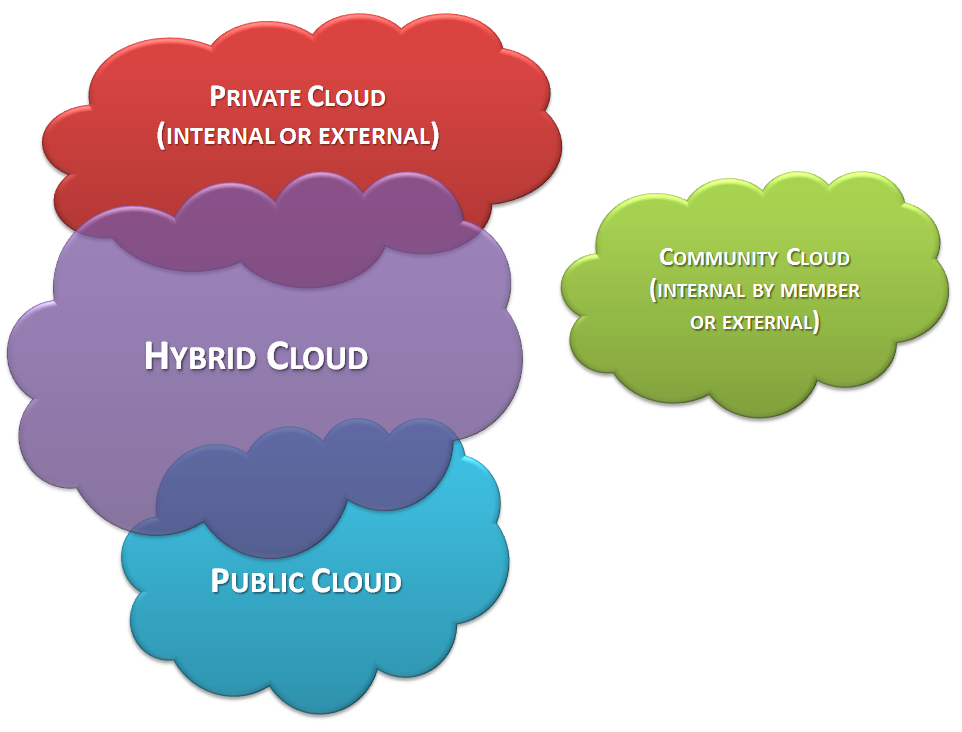
Private cloud means using a cloud infrastructure (network) solely by one customer/organization. It is not shared with others, yet it is remotely located. If the cloud is externally hosted. The companies have an option of choosing an on-premise private cloud as well, which is more expensive, but they do have a physical control over the infrastructure.

The security and control level is highest while using a private network. Yet, the cost reduction can be minimal, if the company needs to invest in an on-premise cloud infrastructure.

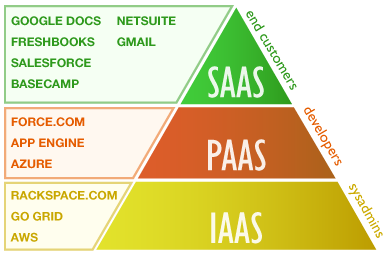
Hybrid cloud, of course, means, using both private and public clouds, depending on their purpose.

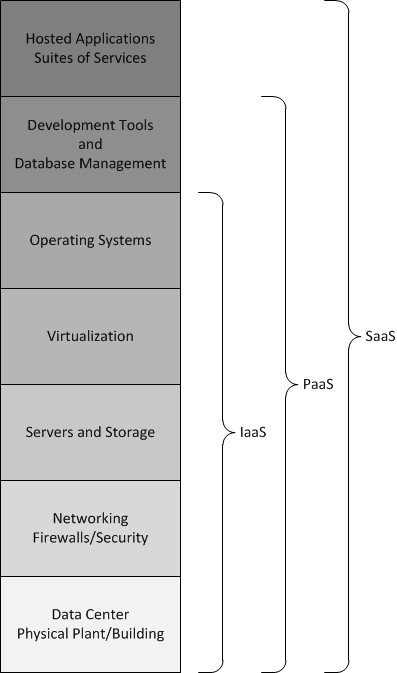
For example, public cloud can be used to interact with customers, while keeping their data secured through a private cloud





# Cloud Service





1. Software as a Service (SaaS***).***The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure2. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.More on [Software as a Service (SaaS)](http://www.service-architecture.com/articles/cloud-computing/software_as_a_service_saas.html)
2. Platform as a Service (PaaS).The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment. More on [Platform as a Service (PaaS)](http://www.service-architecture.com/articles/cloud-computing/platform_as_a_service_paas.html).
3. Infrastructure as a Service (IaaS).The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls). More on [Infrastructure as a Service (IaaS)](http://www.service-architecture.com/articles/cloud-computing/infrastructure_as_a_service_iaas.html).

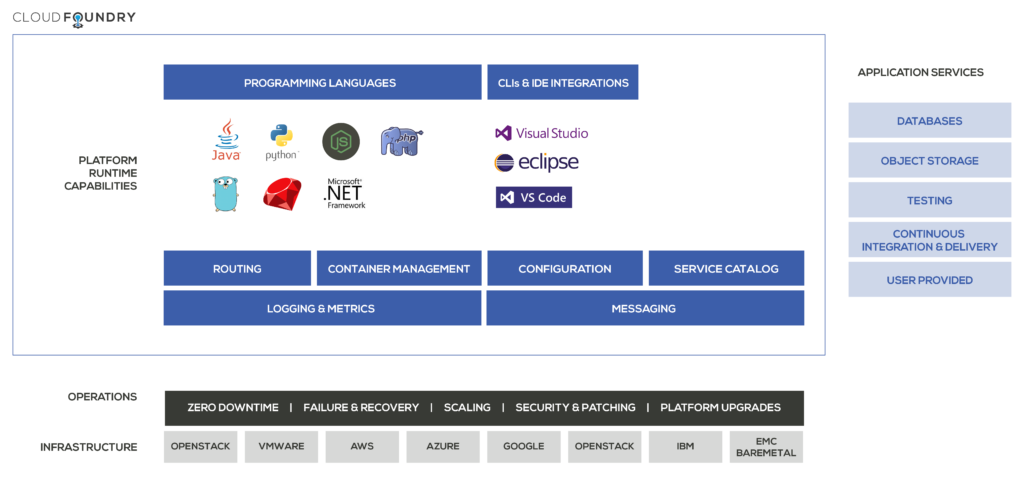
# 10 most powerful PaaS companies

1. Amazon Web Services : Just like Jack planting his magic beans in solid ground, Amazon has built its application runtime PaaS, Amazon Web Service (AWS) Elastic Beanstalk (now in beta), on the solid footing of the very popular Infrastructure as a Service platform, EC2. And while the retailer turned cloud provider doesn’t necessary have a loyal developer following, it’s continually rolling out new tools to entice them. So far the portfolio includes AWS Toolkit for Eclipse (a plug-in for the Eclipse Java Integrated Development Environment), AWS CloudFormation (a service that lets developers create and provision Amazon resources), several cloud-based database options and SDKs for Android and Apple mobile machines, ERuby, Java, PH and .Net.
2. Salesforce.com : The original SaaS giant has successfully parlayed its prowess down the cloud stack with its double one-two PaaS punch of Force.com’s AppExchange and Heruko platform. Right now, the company enjoys the status of market share leader, according to IDC. Salesforce.com touted some impressive numbers to its base at last fall’s DreamForce conference, including the claims that in 2011 3,000 apps were built or installed every 24 hours and that the Force.com platform executes more than 650 million transactions per day.
3. Long Jump : LongJump landed in the PaaS fray very early on in 2008. It has steadily added new features and developer-centric support to attract a customer following that the company says is 600 strong at this point. Long Jump’s biggest coup came in the form of an AT&T partnership announced last November in which the telco rolled out a simplified PaaS service geared toward tech savvy business folks that has the LongJump PaaS stack at its core. Forrester analyst Stefan Reid said this deal could pave the way for more licensing deals for the company.
4. Microsoft(Windows Azur) : There have been rumblings that Microsoft’s 2-year-old Azure PaaS play is not getting the traction the company had hoped. Microsoft’s PaaS portfolio includes the Windows Azure computing environment for applications and persistent storage for both structured and unstructured data; Windows Azure AppFabric a range of services that connects users and on-premises applications to cloud-hosted applications, manages authentication, and implements data management, and SQL Azure, a cloud database service. All Windows centric, of course, but analysts say that Microsoft is making noises about opening that up a bit. And they say you can’t discount the power Microsoft has in its army of .Net developers waiting in the wings to see what the cloud means for them.
5. IBM (IBMSmartCloud) : IBM is a relatively late entry into the PaaS market as it only rolled out SmartCloud Application Services PaaS last October. But the company has long-standing ties to the corporate enterprise and this platform – based on the long-trusted WebSphere middleware -- allows enterprises to build Java-based apps that can run in the public cloud, called IBM Smart Cloud Enterprise, or on premise. IBM is looking to keep their customers in a comfort zone while pushing them out to the cloud.
6. RedHat (OpenShift) : Linux lovers are uniting behind the thought of an open source PaaS like RedHat’s OpenShift. And industry watchers are intrigued by the prospect of portable applications that can be pulled from one infrastructure platform whose fees or contingencies become arduous and easily placed on another one without breakage. It simply remains to be seen how big an ecosystem RedHat can build around OpenShift and how useful it will be corporate developers.
7. Cloudy Foundry : Cloud Foundry is the open source PaaS that was spearheaded in early 2011 by VMware and around which the company plans to build a future commercial product. As with Red Hat’s initiative, VMware is attracting developers who want an open platform that lets them build in the language they want and run on the IaaS they like. According to company officials, the project is gaining significant traction because over 2,100 developers are actively following the changes in the open source code. Analysts have speculated that AppFog, a start-up already offering a comprehensive PaaS based on the Cloud Foundry code, is ripe for the picking should the commercial side of CloudFoundry need a boost.
8. Google( GoogleAppEngine) : Google claims it’s got 200,000 developers building applications on top of its App Engine PaaS. That’s in spite of a rate hike that really got their base riled up back in the September, giving all kinds of fodder to competing products that claim to be both more open and more affordable. But Google doesn’t seem to be bothered and is forging ahead with an upgrade that supports a premium level of service in which customers will receive a 99.95 uptime service-level agreement.
9. CloudBees : CloudBees was first out of the gate with a Java-based PaaS that gave enterprises an easy way to move existing Java applications into the cloud. RUN@cloud is the application runtime side of the CloudBees’ PaaS story, providing traditional application server functionality for web, Java and Spring applications. CloudBees customers choose their underlying IaaS or private cloud. Applications running on RUN@cloud can be built using traditional Java EE development tools or using CloudBees’ second PaaS offering called, DEV@cloud. DEV@cloud is a cloud-based development, build and test environment. CloudBees’ power lies in its understanding that there are lots of sunk costs in existing Java applications from which enterprises are loath to walk away.
10. Engine Yard : Engine Yard is one of the leading PaaS players for Ruby on Rails and PHP developers. These two development languages are most often associated with cool, new greenfield applications running in the cloud. Engine yard, founded in 2006 and still privately held, has a client list that includes Nike, AOL, Apple, Disney, and MTV.

## What is Cloud Foundry?

Cloud Foundry is one of the most mature container-based application platforms. The Cloud Foundry engineering community works with other open source communities on shared industry standards, and integrates technologies maintained within the Open Container Initiative and Cloud Native Computing Foundation.

The platform leverages containers to deploy applications and enables businesses to take advantage of the latest innovations from projects such as Docker and Kubernetes to increase the ease and velocity of managing production-grade applications.



### Elastic Runtime



### **Application Lifecycle Management**

When a developer pushes an application to Cloud Foundry, they are interacting directly with the Cloud Controller. On entering the command cf-push, the Elastic Runtime executes a series of operations that start with bundling the app, staging it, storing any large binary files in the blob storage (such as code packages, buildpacks or droplets), and then running the application as is or pulling a container from a container registry. In addition, the Cloud Foundry platform also has a series of logging and health monitoring components that work together to ensure that the application keeps running.

### **Application Execution**

The Cloud Controller directs the Diego Brain to coordinate individual Diego cells and initiate the process of building, staging and running the application.

### **Networking & Configuration**

There are various components that assist in configuring the Cloud Foundry platform. For example, the router directs incoming traffic to the appropriate component, either a Cloud Controller component or a hosted application running on a Diego cell. The User Account and Authentication Server (UAA) is responsible for identity management.

### **Logging & Monitoring**

The metrics collector gathers statistics from the platform components. Operations can use this to monitor applications deployed on Cloud Foundry or can integrate with existing application logging and monitoring solutions like Splunk, Logstash, New Relic, Datadog and Dynatrace.

### **Platform Services**

Applications typically depend on services such as databases or third-party SaaS providers. When a developer provisions and binds a service to an application, the open service broker for that service is responsible for providing the service instance.

Cloud Foundry’s Service Broker API accelerates the expansion of the global cloud ecosystem, including projects in Google and Microsoft, by providing a single path for developers to add services to applications. Now developers can write and configure against a single API, and reach many developers across multiple platforms. Learn more at [**Open Service Broker API**](https://www.openservicebrokerapi.org/).

### BOSH

BOSH is an open source tool for release engineering, deployment, lifecycle management, and monitoring of distributed system

The Cloud Foundry Foundation also hosts the [BOSH](http://bosh.io/) project as the default operations and deployment layer for the Cloud Foundry platform.

# Cloud Foundry Certified Platforms

## Huawei FusionStage

Huawei “FusionStage Cloud Foundry” is a Cloud Foundry Foundation Certified PaaS offering based on latest Cloud Foundry open source release with Diego as a runtime engine and fully compliant with the CF CLI. It has a fully functional portal for users to deploy and manage their cloud native applications and/or services.

## IBM Bluemix Cloud Foundry

The Bluemix cloud platform is not just about creating new apps or migrating existing ones, on-prem or off-prem implementations, or offering IaaS and PaaS cloud services. It’s designed to bring all of these aspects together to help you solve your real, complex business problems in the cloud.

## Pivotal Cloud Foundry

Pivotal Cloud Foundry® is the world’s most powerful cloud-native platform to build and run software.

## SAP Cloud Platform

SAP Cloud Platform is the enterprise platform-as-a-service with comprehensive application development services and capabilities.

## Swisscom Application Cloud

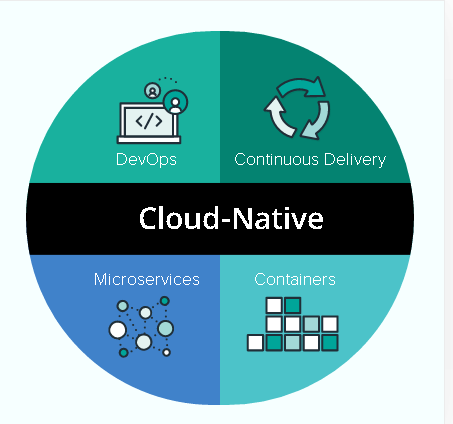
Swisscom Cloud Foundry is provided from the most modern and secure data centres in Switzerland. Corporate customers particularly value our Virtual Private offering and managed enterprise-grade services.

## What Is Cloud Foundry?

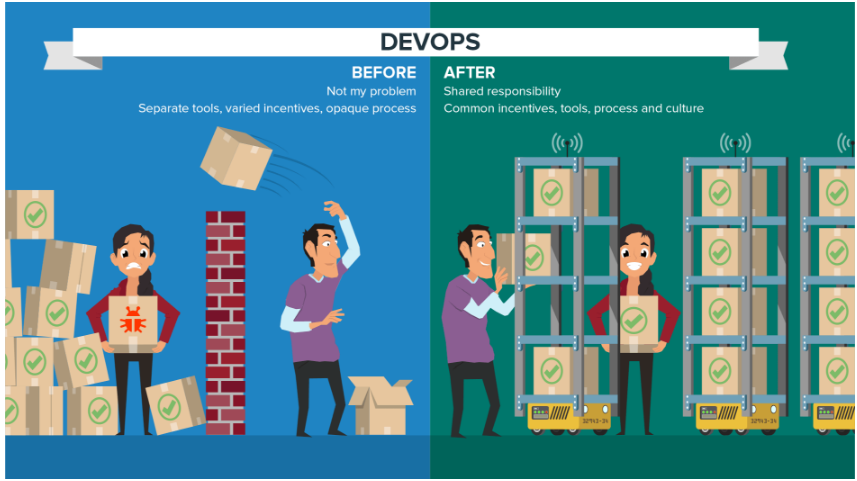
What are Cloud-Native Applications?

Cloud-native is an approach to building and running applications that fully exploits the advantages of the cloud computing model. The cloud has redefined the competitive landscape across virtually all industries by eliminating the focus on capital investment and staff to run an enterprise data center, replacing it with limitless computing power on-demand and pay-as-you-go. Reduced IT spend means a lower barrier to entry with competitive advantage becoming a function of the speed teams can bring new ideas to market, which is why software is eating the world and startups are using cloud-native approaches to disrupt traditional industries.

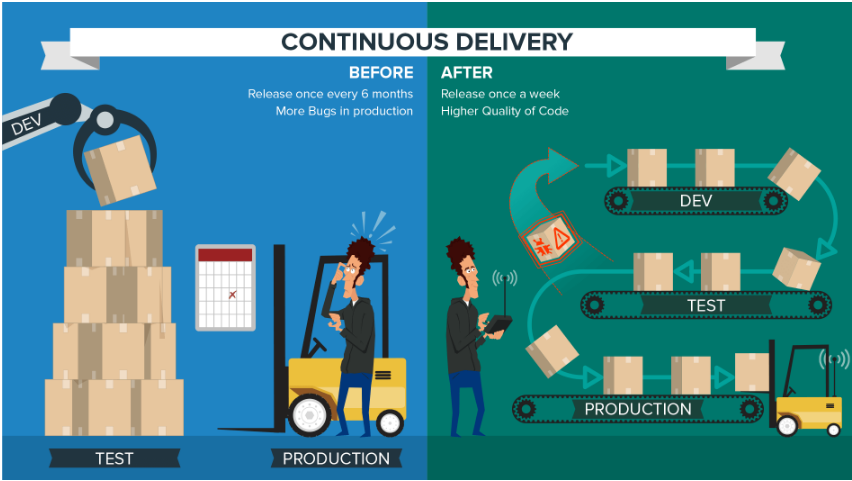
Yet organizations require a platform for building and operating cloud-native applications and services that automates and integrates the concepts of DevOps, continuous delivery, microservices, and containers:



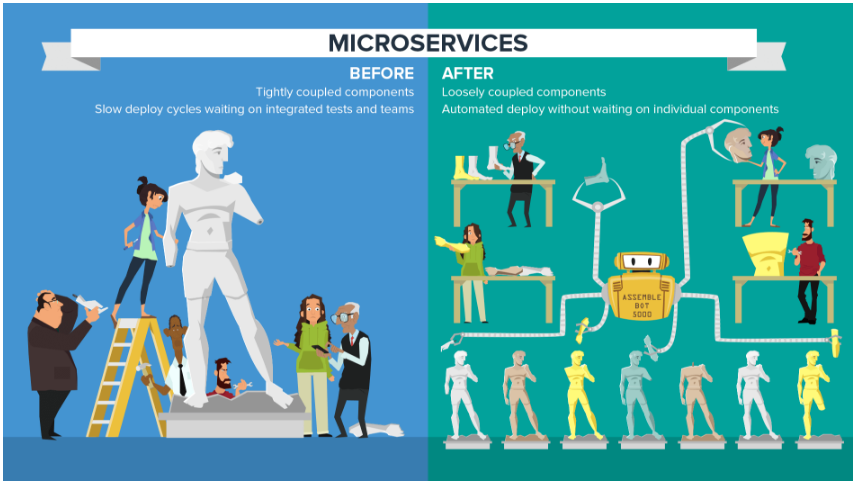
[**DevOps**](https://pivotal.io/devops)is the collaboration between software developers and IT operations with the goal of automating the process of software delivery and infrastructure changes. It creates a culture and environment where building, testing and releasing software can happen rapidly, frequently, and more reliably.



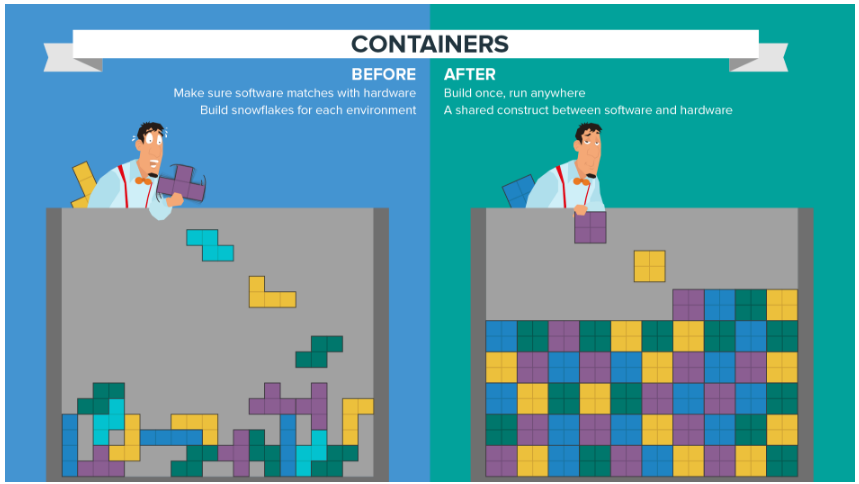
**Continuous Delivery** makes an individual application change ready for release as soon as it is ready, without waiting for bundling with other changes into a release or an event such as a maintenance window. Continuous delivery makes the act of releasing dull and reliable, so organizations can deliver frequently, at less risk, and get feedback faster from end users until deployment becomes an integral part of the business process and enterprise competitiveness.



[Microservices](https://pivotal.io/microservices) is an architectural approach to developing an application as a collection of small services; each service implements business capabilities, runs in its own process and communicates via an HTTP API. Each microservice can be deployed, upgraded, scaled, and restarted independent of other services in the application, typically as part of an automated system, enabling frequent updates to live applications without impacting end customers.



**Containers** offer both efficiency and speed compared with standard virtual machines (VMs). Using operating system (OS)-level virtualization, a single OS instance is dynamically divided among one or more isolated containers, each with a unique writable file system and resource quota. The low overhead of creating and destroying containers combined with the high packing density in a single VM makes containers the perfect compute vehicle for deploying individual microservices.



The Big Differences: Cloud-Native Vs Traditional Enterprise Applications

|  |  |
| --- | --- |
| **CLOUD-NATIVE APPLICATIONS** | **TRADITIONAL ENTERPRISE APPLICATIONS** |
| **Predictable**. Cloud-native applications conform to a framework or “contract” designed to maximize resilience through predictable behaviors. The highly automated, container-driven infrastructure used in cloud platforms drives the way software is written. A good example of such a “contract” is illustrated by the 12 principles first documented as the twelve factor app. | **Unpredictable**. Traditional applications can’t realize all of the benefits of running on a cloud-native platform due to the way they’re architected or developed. This type of application often takes longer to build, is released in big batches, can only scale gradually, and has more single points of failure. |
| **OS abstraction**. Cloud-native application architecture requires developers to use a platform as a means for abstracting away from underlying infrastructure dependencies to enable simple migration and scaling of the application. The most efficient means of abstraction is a formalized platform, for example, Pivotal Cloud Foundry which is ideal for operating on cloud-based infrastructure such as Google Cloud Platform (GCP), Microsoft Azure, or Amazon Web Services (AWS). | **OS dependent.** Traditional application architecture allows developers to build close dependencies between the application and underlying OS, hardware, storage, and backing services. These dependencies make migrating and scaling the application across new infrastructure complex and risky, working against the cloud model. |
| **Right-sized capacity**. A cloud-native application platform automates infrastructure provisioning and configuration, dynamically allocating and reallocating resources at deploy time based on the ongoing needs of the application. Building on a cloud-native runtime optimizes application lifecycle management, including scaling to meet demand, resource utilization, orchestration across available resources, and recovery from failures to minimize downtime. | **Over-sized capacity**. Traditional IT designs a dedicated, custom infrastructure solution (“snowflake”) for an application, delaying deployment of the application. The solution is often over-sized based on worst-case capacity estimates with little capability to scale beyond to meet demand. |
| **Collaborative**. Cloud-native facilitates DevOps, a combination of people, process, and tools, resulting in a close collaboration between development and operations functions to speed and smooth the transfer of finished application code into production. | **Siloed**. Traditional IT operates an ‘over the wall’ handoff of finished application code from developers to operations, which then runs it in production. Organizational priorities take precedence over customer value, resulting in internal conflict, slow and compromised delivery and poor staff morale. |
| **Continuous delivery**. IT teams make individual software updates available for release as soon as they are ready. Organizations that release software rapidly get a tighter feedback loop and can respond more effectively to customer needs. Continuous delivery works best with other related approaches including test-driven development and continuous integration. | **Waterfall development**. IT teams release software periodically, typically weeks or months apart, when code has been built into a ‘release’ despite the fact that many of the components of the release are ready earlier and have no dependency other than the artificial release vehicle. Features that customers want or need are delayed and the business misses opportunities to compete, win customers, and grow revenue. |
| **Independent**. Microservices architecture decomposes applications into small, loosely coupled independently operating services. These services map to smaller, independent development teams and make possible frequent, independent updates, scaling, and failover/restart without impacting other services. | **Dependent**. Monolithic architectures bundle many disparate services into a single deployment package causing unnecessary dependencies between services and leading to a loss of agility during development and deployment. |
| **Automated scalability**. Infrastructure automation at scale eliminates downtime due to human error. Computer automation faces no such challenge, consistently applying the same set of rules across any size of deployment. Cloud-native also goes beyond the ad-hoc automation built on top of traditional virtualization-oriented orchestration. A fully cloud-native architecture includes automation and orchestration that works for teams instead of requiring them to write automations as custom recipes. In other words, the automation makes it easy to build and run applications that can be easily managed. | **Manual scaling**. Manual infrastructure includes human operators that manually craft and manage server, network, and storage configurations. At scale, operators are slow to correctly diagnose issues and easily fail to correctly implement at scale due to the level of complexity. Hand-crafted automation recipes have the potential to hard-code human errors into the infrastructure. |
| **Rapid recovery.** The container runtime and orchestrator provides a dynamic, high-density virtualization overlay on top of a VM, ideally matched to hosting microservices. Orchestration dynamically manages placement of containers across a cluster of VMs to provide elastic scaling and recovery/restart in the event of failure. | **Slow recovery**. VM-based infrastructure is a slow and inefficient foundation for microservice-based applications because individual VMs are slow to startup/shutdown and come with large overhead even before deploying application code to them. |
|  |  |

## Section 6: Cloud Foundry Commands

GETTING STARTED:

help Show help

version Print the version

login Log user in

logout Log user out

passwd Change user password

target Set or view the targeted org or space

api Set or view target api url

auth Authenticate user non-interactively

APPS:

apps List all apps in the target space

app Display health and status for app

push Push a new app or sync changes to an existing app

scale Change or view the instance count, disk space limit, and memory limit for an app

delete Delete an app

rename Rename an app

start Start an app

stop Stop an app

restart Stop all instances of the app, then start them again. This may cause downtime.

restage Recreate the app's executable artifact using the latest pushed app files and the latest environment (variables, servic

e bindings, buildpack, stack, etc.)

restart-app-instance Terminate the running application Instance at the given index and instantiate a new instance of the application with the same index

run-task Run a one-off task on an app

tasks List tasks of an app

terminate-task Terminate a running task of an app

events Show recent app events

files Print out a list of files in a directory or the contents of a specific file of an app running on the DEA backend

logs Tail or show recent logs for an app

env Show all env variables for an app

set-env Set an env variable for an app

unset-env Remove an env variable

stacks List all stacks (a stack is a pre-built file system, including an operating system, that can run apps)

stack Show information for a stack (a stackis a pre-built file system, including an operating system, that can run apps)

copy-source Copies the source code of an application to another existing application (and restarts that application)

create-app-manifest Create an app manifest for an app that has been pushed successfully

get-health-check Show the type of health check performed on an app

set-health-check Change type of health check performedon an app

enable-ssh Enable ssh for the application

disable-ssh Disable ssh for the application

ssh-enabled Reports whether SSH is enabled on an application container instance

ssh SSH to an application container instance

SERVICES:

marketplace List available offerings in the market

place

services List all service instances in the targ

et space

service Show service instance info

create-service Create a service instance

update-service Update a service instance

delete-service Delete a service instance

rename-service Rename a service instance

create-service-key Create key for a service instance

service-keys List keys for a service instance

service-key Show service key info

delete-service-key Delete a service key

bind-service Bind a service instance to an app

unbind-service Unbind a service instance from an app

bind-route-service Bind a service instance to an HTTP rou

te

unbind-route-service Unbind a service instance from an HTTP

route

create-user-provided-service Make a user-provided service instance

available to CF apps

update-user-provided-service Update user-provided service instance

ORGS:

orgs List all orgs

org Show org info

create-org Create an org

delete-org Delete an org

rename-org Rename an org

SPACES:

spaces List all spaces in an org

space Show space info

create-space Create a space

delete-space Delete a space

rename-space Rename a space

allow-space-ssh Allow SSH access for the space

disallow-space-ssh Disallow SSH access for the space

space-ssh-allowed Reports whether SSH is allowed in a sp

ace

DOMAINS:

domains List domains in the target org

create-domain Create a domain in an org for later us

e

delete-domain Delete a domain

create-shared-domain Create a domain that can be used by al

l orgs (admin-only)

delete-shared-domain Delete a shared domain

router-groups List router groups

ROUTES:

routes List all routes in the current space o

r the current organization

create-route Create a url route in a space for late

r use

check-route Perform a simple check to determine wh

ether a route currently exists or not

map-route Add a url route to an app

unmap-route Remove a url route from an app

delete-route Delete a route

delete-orphaned-routes Delete all orphaned routes (i.e. those

that are not mapped to an app)

BUILDPACKS:

buildpacks List all buildpacks

create-buildpack Create a buildpack

update-buildpack Update a buildpack

rename-buildpack Rename a buildpack

delete-buildpack Delete a buildpack

USER ADMIN:

create-user Create a new user

delete-user Delete a user

org-users Show org users by role

set-org-role Assign an org role to a user

unset-org-role Remove an org role from a user

space-users Show space users by role

set-space-role Assign a space role to a user

unset-space-role Remove a space role from a user

ORG ADMIN:

quotas List available usage quotas

quota Show quota info

set-quota Assign a quota to an org

create-quota Define a new resource quota

delete-quota Delete a quota

update-quota Update an existing resource quota

share-private-domain Share a private domain with an org

unshare-private-domain Unshare a private domain with an org

SPACE ADMIN:

space-quotas List available space resource quotas

space-quota Show space quota info

create-space-quota Define a new space resource quota

update-space-quota Update an existing space quota

delete-space-quota Delete a space quota definition and un

assign the space quota from all spaces

set-space-quota Assign a space quota definition to a s

pace

unset-space-quota Unassign a quota from a space

SERVICE ADMIN:

service-auth-tokens List service auth tokens

create-service-auth-token Create a service auth token

update-service-auth-token Update a service auth token

delete-service-auth-token Delete a service auth token

service-brokers List service brokers

create-service-broker Create a service broker

update-service-broker Update a service broker

delete-service-broker Delete a service broker

rename-service-broker Rename a service broker

migrate-service-instances Migrate service instances from one ser

vice plan to another

purge-service-offering Recursively remove a service and child

objects from Cloud Foundry database without making requests to a service broker

purge-service-instance Recursively remove a service instance

and child objects from Cloud Foundry database without making requests to a servi

ce broker

service-access List service access settings

enable-service-access Enable access to a service or service

plan for one or all orgs

disable-service-access Disable access to a service or service

plan for one or all orgs

SECURITY GROUP:

security-group Show a single security group

security-groups List all security groups

create-security-group Create a security group

update-security-group Update a security group

delete-security-group Deletes a security group

bind-security-group Bind a security group to a particularspace, or all existing spaces of an org

unbind-security-group Unbind a security group from a space

bind-staging-security-group Bind a security group to the list of security groups to be used for staging applications

staging-security-groups List security groups in the staging set for applications

unbind-staging-security-group Unbind a security group from the set of security groups for staging applications

bind-running-security-group Bind a security group to the list of security groups to be used for running applications

running-security-groups List security groups in the set of security groups for running applications

unbind-running-security-group Unbind a security group from the set of security groups for running applications

ENVIRONMENT VARIABLE GROUPS:

running-environment-variable-group Retrieve the contents of the running environment variable group

staging-environment-variable-group Retrieve the contents of the staging environment variable group

set-staging-environment-variable-group Pass parameters as JSON to create a staging environment variable group

set-running-environment-variable-group Pass parameters as JSON to create a running environment variable group

FEATURE FLAGS:

feature-flags Retrieve list of feature flags with status of each flag-able feature

feature-flag Retrieve an individual feature flag with status

enable-feature-flag Enable the use of a feature so that users have access to and can use the feature

disable-feature-flag Disable the use of a feature so that users have access to and can use the feature

ADVANCED:

curl Executes a request to the targeted API

endpoint

config Write default values to the config

oauth-token Retrieve and display the OAuth token for the current session

ssh-code Get a one time password for ssh clients

ADD/REMOVE PLUGIN REPOSITORY:

add-plugin-repo Add a new plugin repository

remove-plugin-repo Remove a plugin repository

list-plugin-repos List all the added plugin repositories

repo-plugins List all available plugins in specifie

d repository or in all added repositories

ADD/REMOVE PLUGIN:

plugins List all available plugin commands

install-plugin Install CLI plugin

uninstall-plugin Uninstall the plugin defined in comman

d argument

INSTALLED PLUGIN COMMANDS:

ENVIRONMENT VARIABLES:

CF\_COLOR=false Do not colorize output

CF\_DIAL\_TIMEOUT=5 Max wait time to establish a connection, including name resolution, in seconds

CF\_HOME=path/to/dir/ Override path to default config directory

CF\_PLUGIN\_HOME=path/to/dir/ Override path to default plugin config dir

ectory

CF\_TRACE=true Print API request diagnostics to stdout

CF\_TRACE=path/to/trace.log Append API request diagnostics to a log file

https\_proxy=proxy.example.com:8080 Enable HTTP proxying for API requests

GLOBAL OPTIONS:

--help, -h Show help

-v Print API request diagnostics to stdout

**Reference links:**

How PCF Works   
<https://docs.pivotal.io/pivotalcf/concepts>

PCF Documentation   
<https://docs.pivotal.io/pivotalcf/installing/pcf-docs.html>

Installing PCF (IaaS-specific guides for installing PCF)   
<https://docs.pivotal.io/pivotalcf/installing/>

Learn more about the Spring Framework   
<https://spring.io/guides>

Explore and download more Cloud Foundry sample apps   
<https://github.com/cloudfoundry-samples/>