

# Portabilidade

## Rodando Aplicações em Múltiplos Provedores de Cloud Pública



Evitando vendor lock in com uma  
estratégia multi-cloud com Crossplane

Jackson Oliveira, Diego Pacheco e Marcelo Serpa

## Sumário

<b>O Impacto da Cloud no mercado</b> .....	<b>3</b>
Preocupações para considerar quando migrando para uma Cloud Pública.....	4
Vendor lock in.....	4
Disponibilidade.....	4
Acelerar ciclos de inovação. ....	4
Padrões Cloud-Native Emergentes.....	4
Containers.....	4
Kubernetes.....	5
Estratégias de Arquitetura Cloud-native.....	5
Multi-cloud.....	5
Polycloud.....	6
Crossplane- Um painel de controle multi-cloud.....	6
Separação de Papéis.....	7
Como Crossplane funciona?.....	7
Um exemplo de aplicação Multi-cloud.....	7
Conclusão.....	8

## O Impacto da Cloud no Mercado

A computação em cloud tem transformado a forma como empresas operam. Há não muito tempo atrás, as empresas precisam investir em datacenters próprios para suportar a sua operação bem como ciclos de inovação. Este movimento de expansão computacional era e ainda é caro e lento tendo como uma das principais consequências um alto lead-time entre identificar novas oportunidades de mercado e torná-las disponíveis para o seus clientes.

Nem todas as empresas são capazes de investir em datacenters próprios, isto de alguma forma restringiu por muito tempo a capacidade de inovar à empresas com maior capital e pessoal especializado a disposição.

Com a popularização da cloud pública o cenário mudou. Empresas não precisam mais manter data centers próprios para ter capacidade computacional. Se valendo do modelo **Pay as you go**, empresas investem somente pelos recursos que precisam e quando precisam, delegando a responsabilidade de manter a infraestrutura para o provedor de cloud.

Usando a elasticidade da cloud pública, empresas não mais precisam esperar semanas ou às

vezes até meses para ter recurso computacional a disposição, na realidade está sempre disponível, basta requisitar.

Neste modelo, infraestrutura deixa de ser um gargalo. Temos visto startups sacudindo as suas áreas de atuação e uma das razões e a forma de como a computação em cloud é oferecida para o público geral.

Os benefícios da Computação em Cloud para as empresas são inquestionáveis. Empresas que vão nesta direção não querem voltar atrás, embora esta seja uma longa jornada. Apesar das vantagens, algumas avaliações precisam ser feitas pelos tomadores de decisão. Uma vez tomada a decisão de migrar para um Cloud provider específico é difícil voltar atrás e escolher outro e começar novamente. É uma decisão de longo prazo difícil de ser desfeita.

Dependendo do contexto, isto pode não ser um problema, mas falando de Flexibilidade, empresas talvez queiram avaliar soluções de outros Cloud Providers ou serviços específicos que cada um ofereça. Algumas vezes esta não é uma estratégia possível ou muita cara de executar. Algumas das razões são descritas nas seções seguintes:

# Preocupações para considerar quando migrando para uma Cloud Pública.

## Vendor lock in

Esta é de longe o cenário mais comum na adoção de uma Cloud Pública. Há casos em que acontece por design (exemplo: quanto mais usa, mais descontos recebe) ou por acidente. *Vendor lock in* se caracteriza por uma arquitetura que só pode rodar uma plataforma de um Cloud provider específico. Na maioria dos casos, são encontrados os seguintes elementos na solução:

- *Uso de Managed services.*
- *Dependência em um sistema de virtualização específico de um Cloud Provider.*
- *Soluções de armazenamento de dados baseada serviços oferecidos por Cloud Providers (exemplo: S3, Cloud storage, MS Azure Blob Storage)*

Evitar *vendor lock in* é desafiador. Dependendo do cenário, vai demandar dos times um esforço grande de engenharia dos profissionais envolvidos que muita empresas não estão dispostas a pagar. Um Exemplo é adoção de Managed Services (exemplo: banco de dados) aumenta muito a produtividade dos times, uma vez que maioria das tarefas operacionais para manter solução é responsabilidade do Cloud Provider que oferece o serviço. Do contrário, times dedicados precisam existir para manter, suportar e evoluir soluções específicas. Tal cenário vai exigir investimento uma vez que exige profissionais dedicados e qualificados dentro dos times. Apesar desta abordagem ajudar evitar *vendor lock in* nem todas as empresa estão dispostas a investir nisto.

Outro fator que pesa a favor do uso de *Managed Services* é que Cloud Providers geralmente oferecem vantagens financeiras tangíveis aderindo a esta modalidade. Por exemplo, tráfego de rede interno geralmente não é cobrado quando comunicando com *Managed Services*. Neste cenário, apenas considerar custo como variável, vai ter efeitos direto na Arquitetura da solução

sendo difíceis de mudar no futuro.

## Disponibilidade

Usando um único Cloud Provider, o uptime das soluções vão depender diretamente do uptime do Provider. Falhas que comprometem serviços de zonas ou regiões inteiras não são muito comuns, entretanto, quando acontecem, geralmente trazem para a mesa debates se usar ou não mais de um Cloud Provider é algo que precisa ser considerado.

Verdade é que abordagens Multi-cloud são complexas e caras de implementar. Aplicações onde indisponibilidade pode causar prejuízos consideráveis são candidatos a se considerar para esta abordagem, mas definitivamente não é a maioria dos casos.

## Acelerar ciclos de inovação.

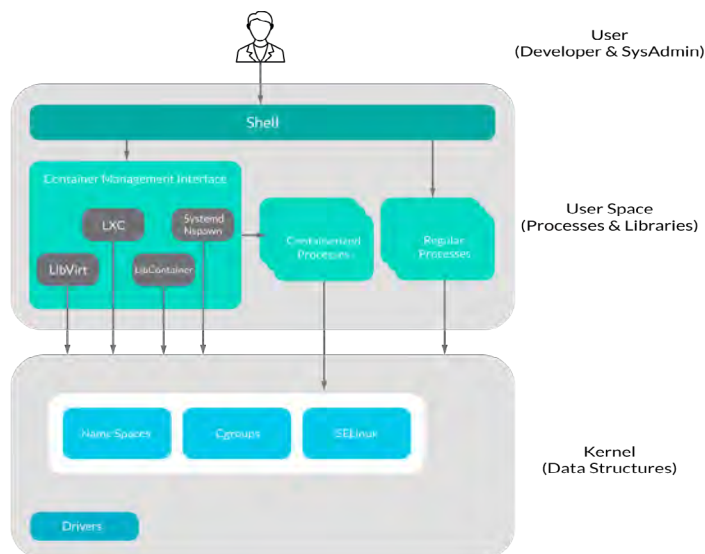
Para acelerar o ciclo de inovação, empresas precisam ser aptas a experimentar e/ou usar as técnicas e/ou ferramentas mais recentes disponíveis. Dependendo do cenário, um único Cloud Provider não ofereça as melhores soluções para este tipo de trabalho. Podemos citar a GCP como exemplo. A GCP oferece hoje um dos melhores sets de ferramentas e serviços para Machine Learning. De um outro lado, a cloud a AWS tem uma das melhores relações custo/benefício para computação em geral. E fosse o caso, de desenvolver serviços e reconhecimento facial na GCP mas continuar rodando computação da AWS? Usando um único cloud provider para todas as soluções talvez não seja possível ou talvez comprometa um ciclo rápido de inovação.

## Padrões Cloud-Native Emergentes

### Containers

Containerização permite que aplicações sejam empacotadas e lançadas de uma padrão, ou seja, do mesmo jeito em qualquer plataforma indepen-

dente da infraestrutura que está rodando. Pode ser a máquina do desenvolvedor ou um servidor na Cloud, contêineres são tratados da mesma forma em ambos os casos.

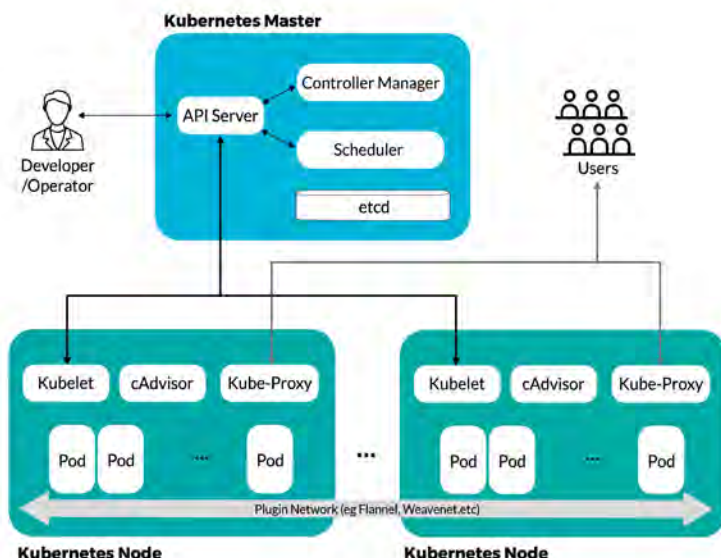


Containerização não é uma técnica nova, ela já existe desde o início no Linux, entretanto se tornou muito mais popular com a difusão de *Docker* que oferece uma boa API em cima de LXC. Usando contêineres, não se faz necessário lidar com certos aspectos de infraestrutura na deploy e runtime da aplicação uma vez que ele suporte containers (exemplo: docker daemon no host onde vai rodar a aplicação).

Apesar do fato de ser uma abordagem com maior portabilidade, rodar contêineres em escala oferece desafios. Novas ferramentas são necessárias para orquestrar a execução dos containers. Existem várias soluções no mercado para orquestração, entretanto a mensagem é: Contêineres por si só são suficientes.

## Kubernetes

Kubernetes é uma solução open-source de orquestração de contêineres desenvolvida pela Google. Se tornou membro da Cloud Native Foundation e desde então tem sido largamente utilizada como uma solução para orquestrar contêineres em escala.



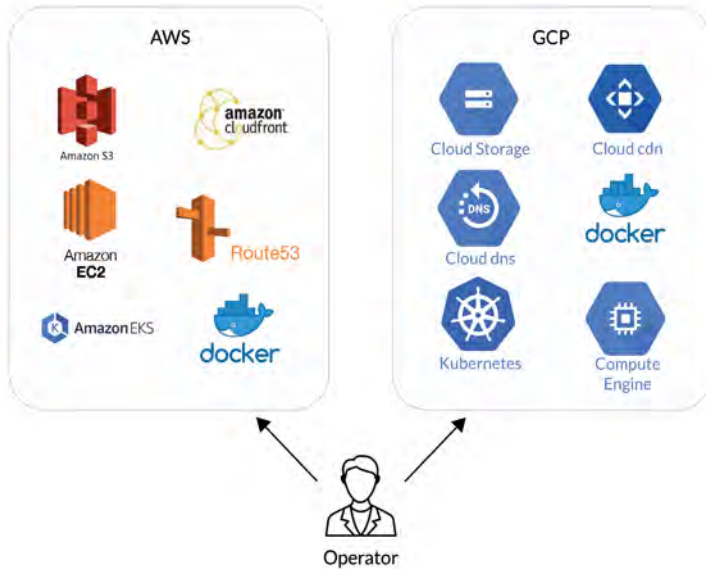
Kubernetes tem se tornado um padrão de mercado, tanto é verdade que os 3 maiores Cloud Provider já oferecem Kubernetes com *Managed Service*. Significa também que é possível fazer deploy, rodar e operações aplicações do mesmo jeito independente do Cloud Provider. Em termos de portabilidade e evitar Vendor Lock in, Kubernetes é a melhor opção no mercado para orquestrar aplicações em contêineres.

Entretanto kubernetes e Containers não representam o Landscape geral. Outros componentes como Object Storage, databases, serverless, caches, etc. são partes que compõe o todo. A adoção de Kubernetes pura e simples não cobre todas estas dimensões.

## Estratégias de Arquitetura Cloud-native

### Multi-cloud

Nesta abordagem, empresas estão usando serviços de mais Cloud Provider, e mais importantes ao mesmo tempo. Abaixo é um exemplo de como esta abordagem se parece.



Neste cenário, uma dada empresa quer ter as suas aplicações portáteis entre mais de um provider, ou seja, a mesma aplicação sem modificações roda em diferentes plataformas de Cloud. Na imagem não aparece, mas é bem provável que exista a necessidade de sincronizar dados entre os diferentes Cloud Provider através de algum pareamento, dependendo do cenário.

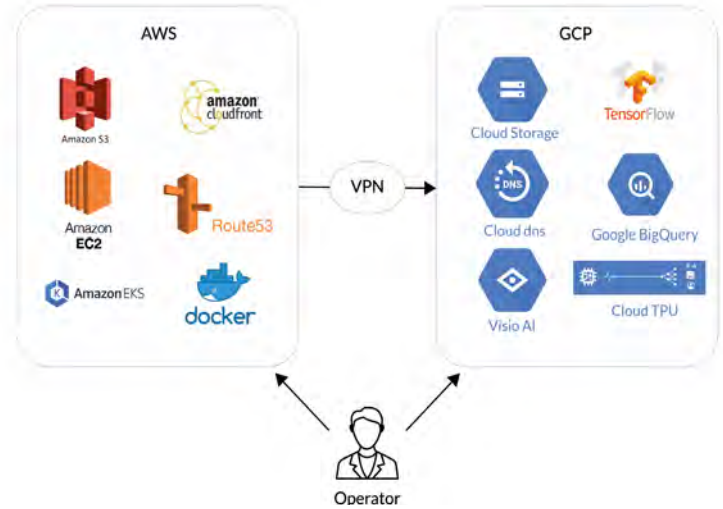
Este é de longe a abordagem mais flexível e portátil para rodar em aplicações na cloud, entretanto é de longe também a mais complexa e cara pelas seguintes razões:

1. Exige um complexo nível automação para ter toda a infraestrutura disponível;
2. Manter sincronização de dados de forma confiável pode ser desafiador e trazer riscos;
3. Requer engenheiros altamente qualificados. Para manter o código da aplicação desacoplado de qualquer funcionalidade específica de um Cloud Provider ou alguma SDK;
4. Uso de Managed Services muito provavelmente não será possível, aumentando os custos de operação.

## Polycloud

Diferentemente de Multi-cloud, Polycloud aceita o fato de mais uma cloud ao mesmo tempo, entretanto se considera o uso dos melhores serviços de uma delas para resolver diferentes problemas. Nesta abordagem se aceita o lock in até um certo nível (exemplo: aceita o uso de Managed Servi-

ces) e diferentemente da abordagem Multi-cloud é bem mais simples.



Na imagem acima, podemos ver as diferenças. De um lado, maiores do serviços usados na AWS são de computação e do outro serviços de Machine Learning na GCP.

Conectividade entre os Cloud Providers não é mandatória, vai depender o caso.

Os desafios nesta abordagem são:

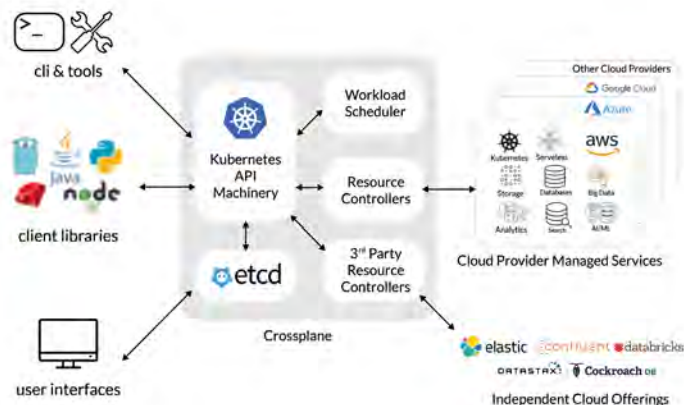
1. Preocupação com segurança nos dois providers;
2. Menos uso de serviços em cada um dos providers reduzem a possibilidades de descontos.

## Crossplane - Um painel de controle multi-cloud

Crossplane é um framework a nível de infraestrutura que permite o deploy de workloads portáteis em diferentes Cloud Providers, e por causa disso cai na categoria de solução **Multi-cloud**.

Quando se diz que é portátil, não está se referindo apenas a contêineres, mas também a bases de dados, object storages e serverless. Crossplane por debaixo dos panos manipula diferentes SDKs de diferentes Cloud Providers, desta forma ele pode **provisionar qualquer Managed Service** via configuração. Ele cobre um grande GAP em arquiteturas Multi-cloud usando Kubernetes, ou

seja, provisiona mais do que apenas aplicações rodando em contêineres mas ainda assim com a mesma portabilidade. A figura abaixo mostra uma visão alto nível da arquitetura:



Crossplane roda em cima do Kubernetes. Usando a API do kubernetes ele recebe requisição tanto de operadores quanto desenvolvedores para provisionar recursos.

## Separação de Papéis

Crossplane enforça alguns papéis na sua operação. São eles:

- **Operador:** Este vai ser responsável por configurar a Cloud (todas as contas). É a pessoa com skills de operação que vai criar a configuração base que toda a plataforma precisa para criar recursos. Este papel pode ficar responsável por:
  - *Instalar o crossplane;*
  - *Manter o cluster onde Crossplane está instalado;*
  - *Definir templates de configuração que serão reusados por toda a plataforma;*
  - *Vai definir o Como os recursos vão ser provisionados.*
- **Desenvolvedor:** papel com skill de engenharia. Vai se valer de todo o trabalho feito pelo Administrador, ou seja, não entra em detalhes de provisionamento ou configuração de cloud. Ele é responsável por:
  - *Definir uma stack de aplicação (serviços, bancos de dados, storage) via configuração;*
  - *Vai definir o que precisa para a sua solução.*

Esta separação de papéis tem como objetivo reduzir a fricção entre dev e ops, fazendo cada um atuar mais próximo de sua área de expertise.

## Como Crossplane funciona?

Crossplane permite definir recursos (credenciais e configurações de gerais de uma conta na cloud) bem como *Workloads* (recursos de aplicação tais como: contêineres, bases de dados e storage) usando a sintaxe declarativa do Kubernetes. Seguindo as instruções disponíveis [aqui](#), podemos perceber que Crossplane é uma aplicação rodando dentro do Kubernetes como qualquer outra. Crossplane se vale dos seguintes pontos de extensão na Arquitetura do Kubernetes:

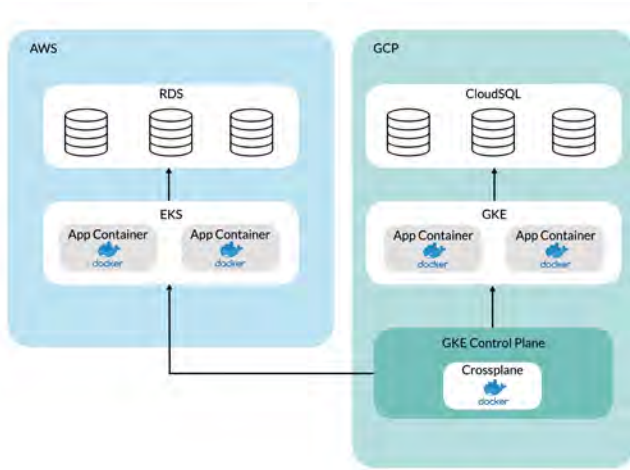
- **CRDs:** Permite criar recursos customizados usando a sintaxe declarativa de definição de recursos do Kubernetes e interagir com a suas APIs. Abaixo um exemplo de como definir uma stack do Crossplane.:

```
apiVersion: storage.crossplane.io/v1alpha1
kind: MySQLInstance
metadata:
  name: demo-mysql
  namespace: default
spec:
  classReference:
    name: standard-aws-mysql
    namespace: crossplane-system
  engineVersion: "5.7"
```

- **Operators:** Lógicas customizadas que são capazes de processar uma request enviada pela KubeAPI e aplicar no ambiente. Este é o componente responsável por garantir o estado da infraestrutura em mais de uma cloud de acordo com o que foi definido nas SPECS.

## Um exemplo de aplicação Multi-cloud

Usando as abstrações do Crossplane é possível lidar com aplicações rodando em mais de uma cloud ao mesmo tempo. A imagem abaixo mostra como poderia funcionar:



Na imagem acima, Crossplane está desempenhando o papel de **Painel de Controle**. Desta forma, ele pode provisionar recursos e controlar o ciclo de vida dos seguintes recursos:

- Cluster GKE;
- Cluster Cloud SQL;
- Cluster RDS;
- Cluster EKS;
- Deploy de uma mesma aplicação em ambos GKE e EKS cluster.

Uma vez configurado, neste modelo, é possível fazer o deploy multi-cloud de quantas outras aplicações forem necessárias. Suporte a outros cloud providers podem ser adicionados através de um novos *ResourceClass* e *Provider*. Um exemplo funcionando pode ser encontrado [aqui](#).

## Conclusão

Apesar dos benefícios de Arquiteturas Multi-cloud na maioria das vezes implementá-las tem um alto custo envolvido. Crossplane provê uma série de abstrações que reduzem a complexidade de lidar com ambientes multi-cloud usando a sintaxe declarativa do kubernetes. Fazer uso das SPEcs do Kubernetes parece um opção lógica uma vez que se tornou padrão de mercado para aplicações Cloud Native.

Crossplane é uma solução muito promissora, mas ainda não é uma solução pronta para produção. Apenas alguns Managed Services são suportados no momento em que este artigo é escrito. Sua portabilidade vem do fato que múltiplos Cloud Providers suportam Managed Services similares, fato que pode limitar o uso desta solução para casos muito específicos.



## Oxigenamos negócios para um mundo inquieto.

Somos uma empresa global de design, inovação e software.  
Desenvolvemos estratégias e experiências de cliente que transformam negócios.



Porto Alegre +55 51 3212-8444 | São Paulo +55 11 3171-0555

[comercial@ilegra.com](mailto:comercial@ilegra.com)

[www.ilegra.com](http://www.ilegra.com)