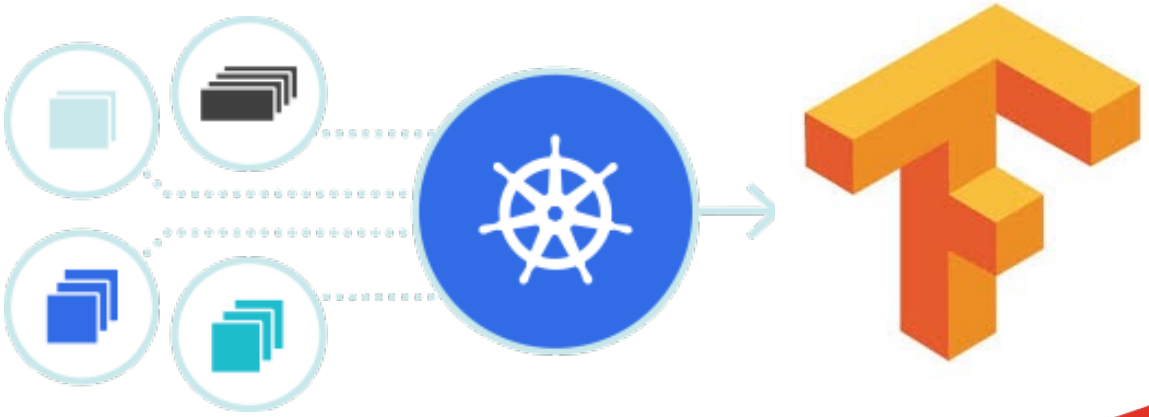
inwinSTACK
G.O.D. Great OpenStack Delivery

openstack. FOUNDATION
GOLD MEMBER

TensorFlow + Kubernetes = Google OP!



About Me

白凱仁
Kyle Bai

- inwinSTACK 菜鳥攻陳屍
- 來自 HTC 2.0 台中北區分公司
- OpenStack 與 Ceph Contributor
- 2015 全國大專院校OpenStack 程式創意設計競賽 - 冠軍

https://goo.gl/GZAQ4E

Agenda

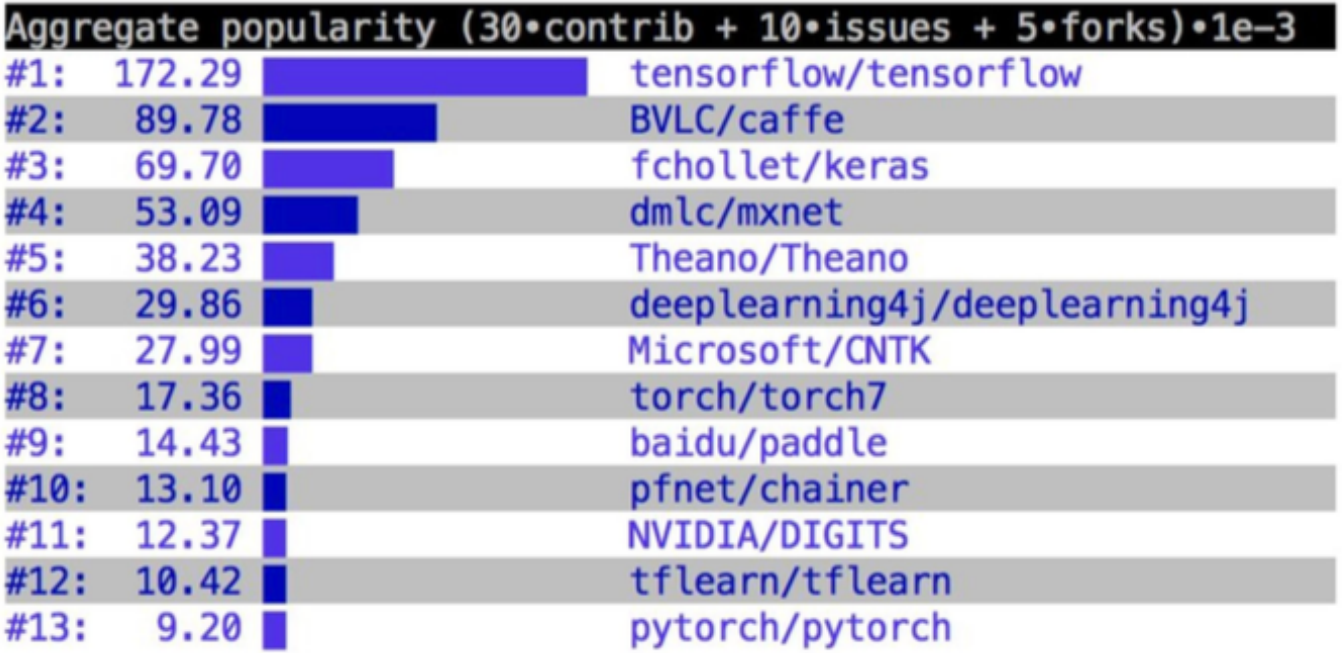
- TensorFlow
- Distributed TensorFlow
- Kubernetes
- Why TensorFlow + Kubernetes?
- TensorFlow on Kubernetes Lab

TensorFlow

TensoFlow 是 Google 開源的機器學習框架，其內建支援許多深度學習，使用途非常廣泛，該框架可用於任何運算流程圖（computational flow graph），並執行於不同異質環境。

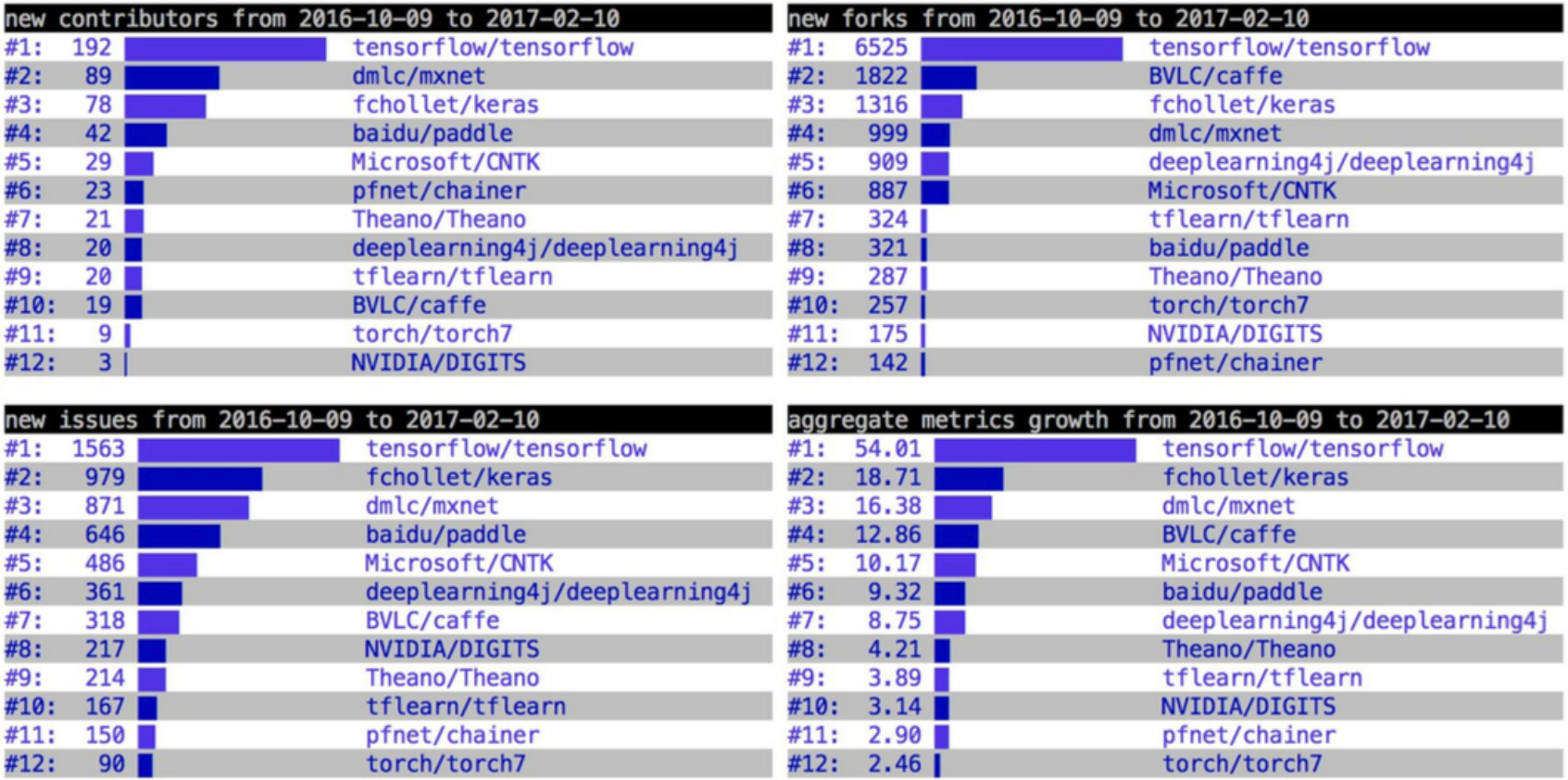
https://hackmd.io/s/HJxsUvOpg

Deep learning libraries: Accumulated GitHub metrics



https://twitter.com/fchollet/status/830499993450450944/

Deep learning libraries: growth over past three months

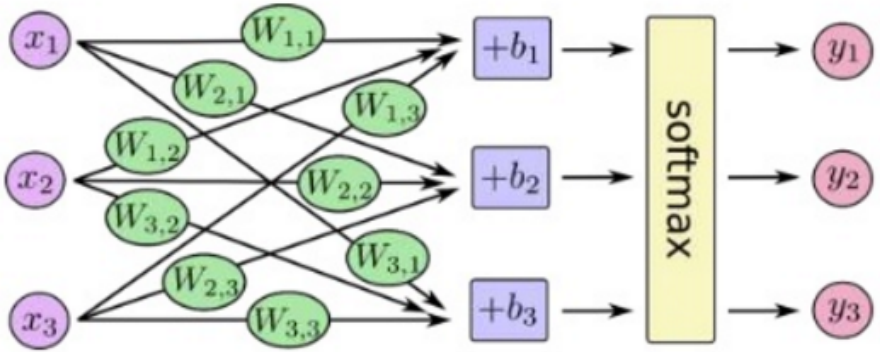


https://twitter.com/fchollet/status/830499993450450944/

Tensor + Flow

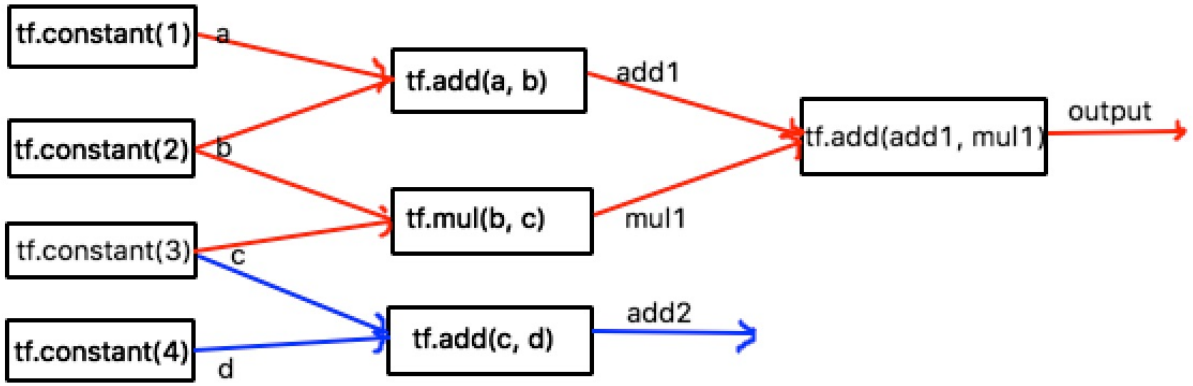
TensorFlow 其實在意思上是要用兩個部分來解釋，Tensor 與 Flow：

- **Tensor**：是中文翻譯是張量，其實就是一個n維度的陣列或列表。如一維 Tensor 就是向量，二維 Tensor 就是矩陣等等.
- **Flow**：是指 Graph 運算過程中的資料流.



Data Flow Graphs

資料流圖(Data Flow Graphs)是一種有向圖的節點(Node)與邊(Edge)來描述計算過程。




```python
# coding=utf-8
import tensorflow as tf

a = tf.constant(1)
b = tf.constant(2)
c = tf.constant(3)
d = tf.constant(4)
add1 = tf.add(a, b)
mul1 = tf.multiply(b, c)
add2 = tf.add(c, d)
output = tf.add(add1, mul1)

with tf.Session() as sess:
    print sess.run(output)
```

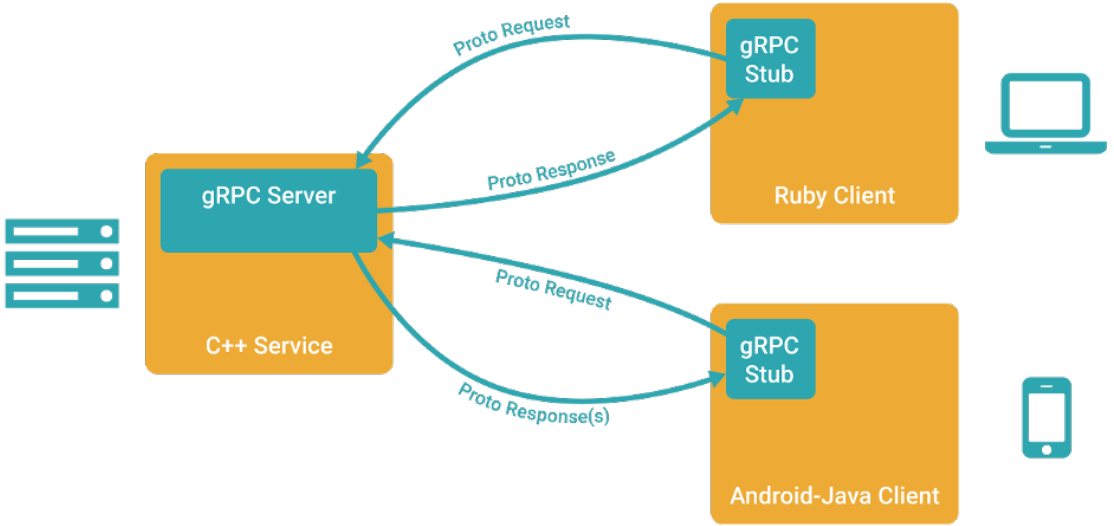

https://github.com/hunkim/DeepLearningZeroToAll/

Distributed TensorFlow

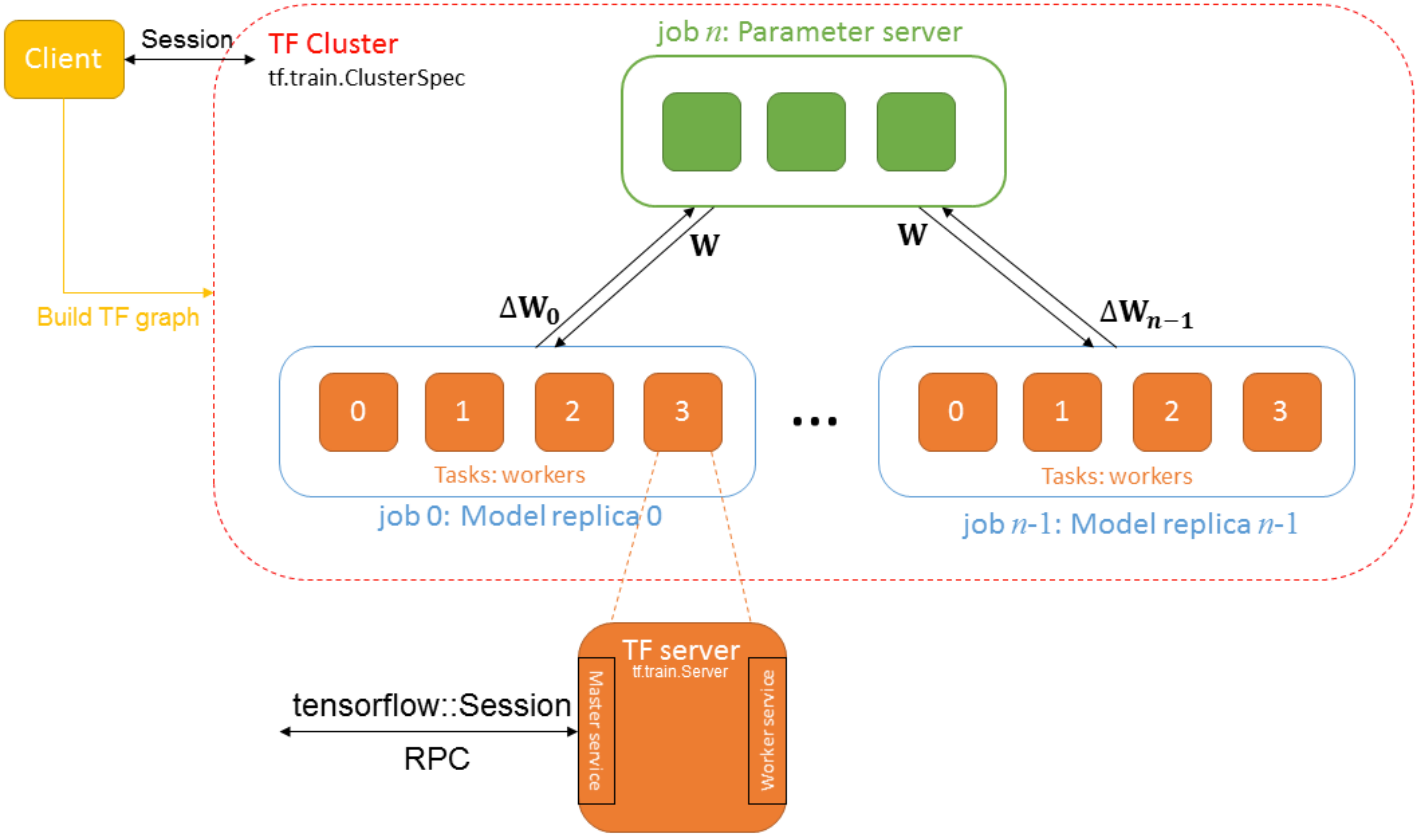
去年四月中旬 Google 釋出 TensorFlow 0.8，新增加分散式運算能力，使 TensorFlow 可在數百台的機器上執行訓練程序，建立各種機器學習模型，將原本要耗費數天或數個星期的模型訓練過程縮短到數小時。

gRPC

gRPC(google Remote Procedure Call) 是 Google 開發基於 HTTP/2 和 Protocol Buffer 3 的 RPC 框架，該框架有各種常見語言的實作，如 C、Java 與 Go 等語言，提供輕鬆跨語言的呼叫。



Distributed TensorFlow 架構



Distributed TensorFlow 架構

如圖上所示，幾個流程說明如下：

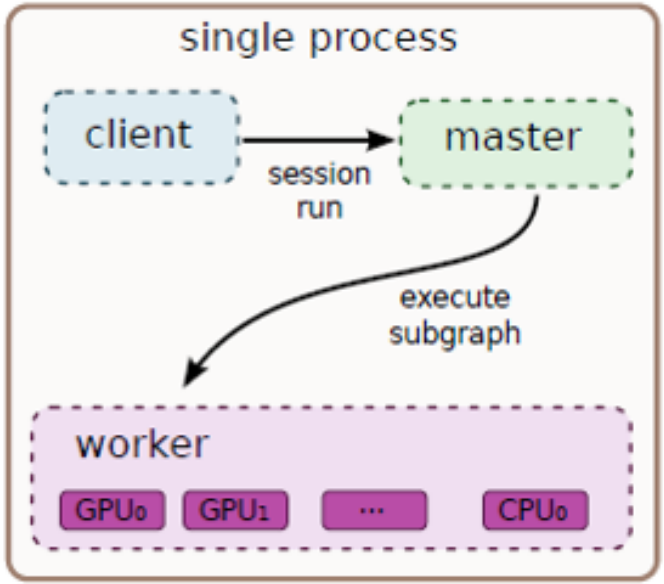
- 整個系统映射到 TensorFlow 叢集.
- 參數伺服器映射到一個 Job.
- 每個模型(Model)副本映射到一個 Job.
- 每台實體運算節點映射到其 Job 中的 Task.
- 每個 Task 都有一個 TF Server，並利用 Master 服務來進行溝通與協調工作，而 Worker 服務則透過本地裝置(CPU 或 GPU)進行 TF graph 運算.

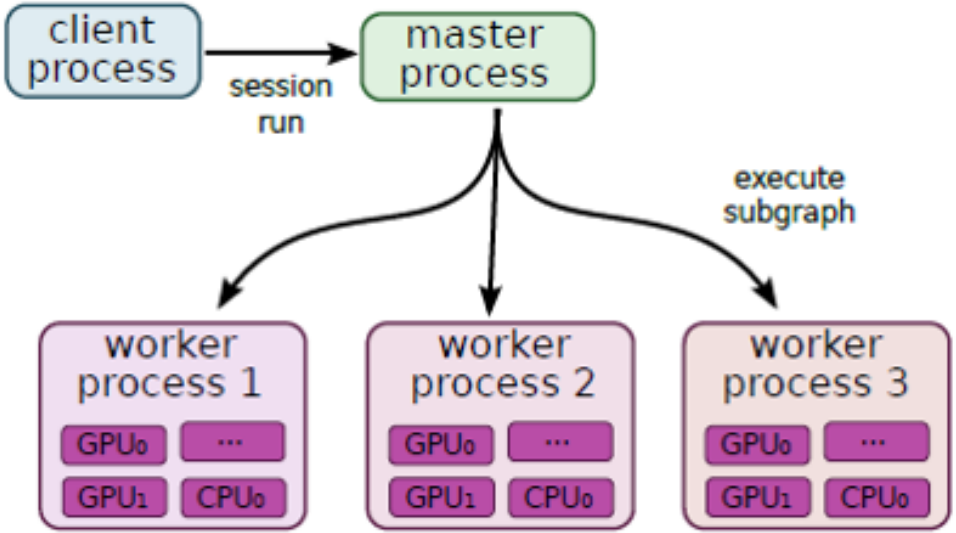
TensorFlow Cluster

TensorFlow 的叢集就是一組工作任務，每個任務是一個服務，服務分成了 **Master** 與 **Worker**，並提供給 **Client** 進行操作。

- **Client**：是用於建立 TensorFlow 計算 Graph，並建立與叢集進行互動的 tensorflow::Session 行程，一般由 Python 或 C++ 實作，單一客戶端可以同時連接多個 TF 伺服器連接，同時也能被多個 TF 伺服器連接.
- **Master Service**：是一個 RPC 服務行程，用來遠端連線一系列分散式裝置，主要提供 tensorflow::Session介面，並負責透過 Worker Service 與工作的任務進行溝通.
- **Worker Service**：是一個可以使用本地裝置(CPU 或 GPU)對部分 Graph 進行運算的 RPC 邏輯，透過 worker_service.proto 介面來實作，所有 TensorFlow 伺服器均包含了 Worker Service 邏輯.

TensorFlow Cluster



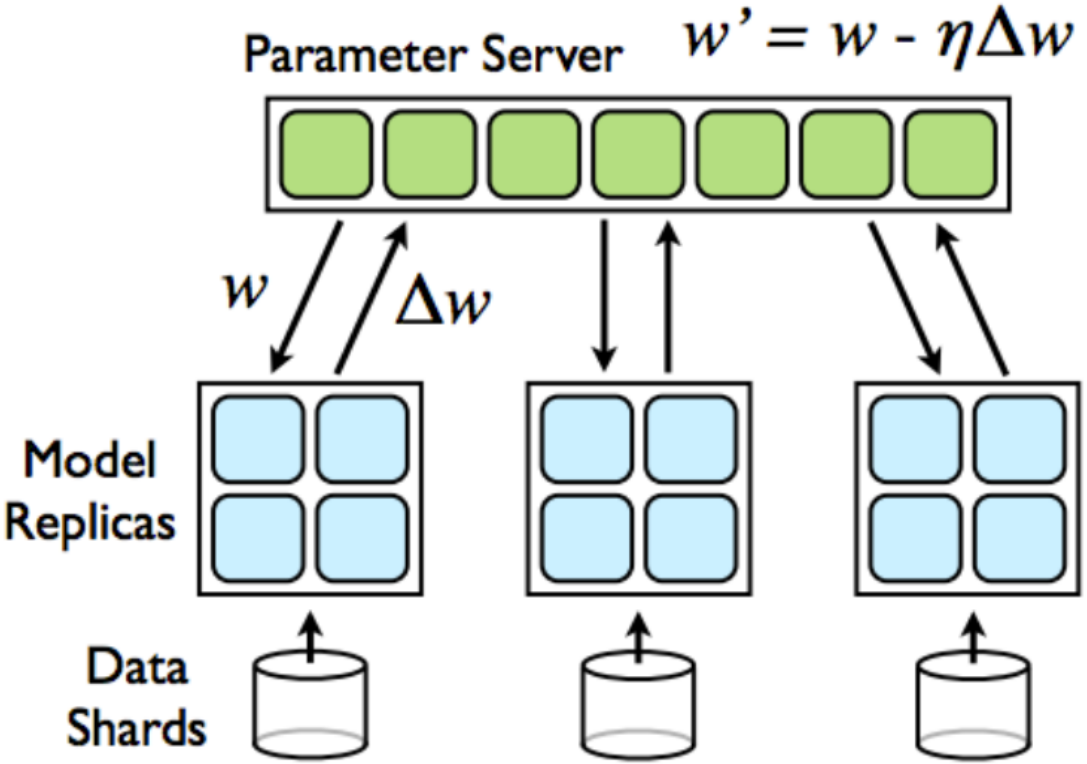


TensorFlow Job

TensorFlow 的工作(Job)可拆成多個相同功能的任務(Task)，這些工作又分成 **Parameter server** 與**Worker**。

- **Parameter server(ps)**:主要根據梯度更新變數，並儲存於 tf.Variable，可理解成只儲存 TF Model 的變數，並存放 Variable 副本。
- **Worker**:通常稱為計算節點，主要執行密集型的 Graph 運算資源，並根據變數運算梯度。存放 Graph 副本。

Parameter server

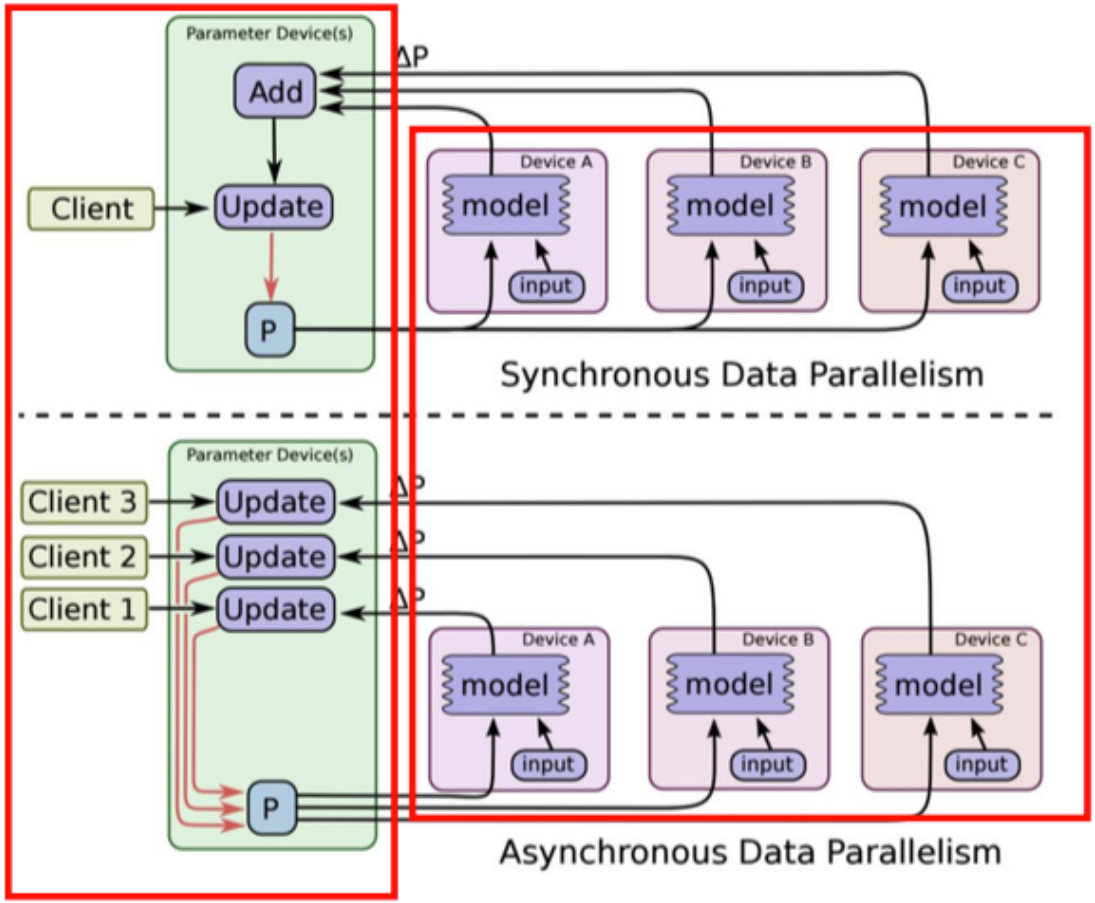


不同規模的分配

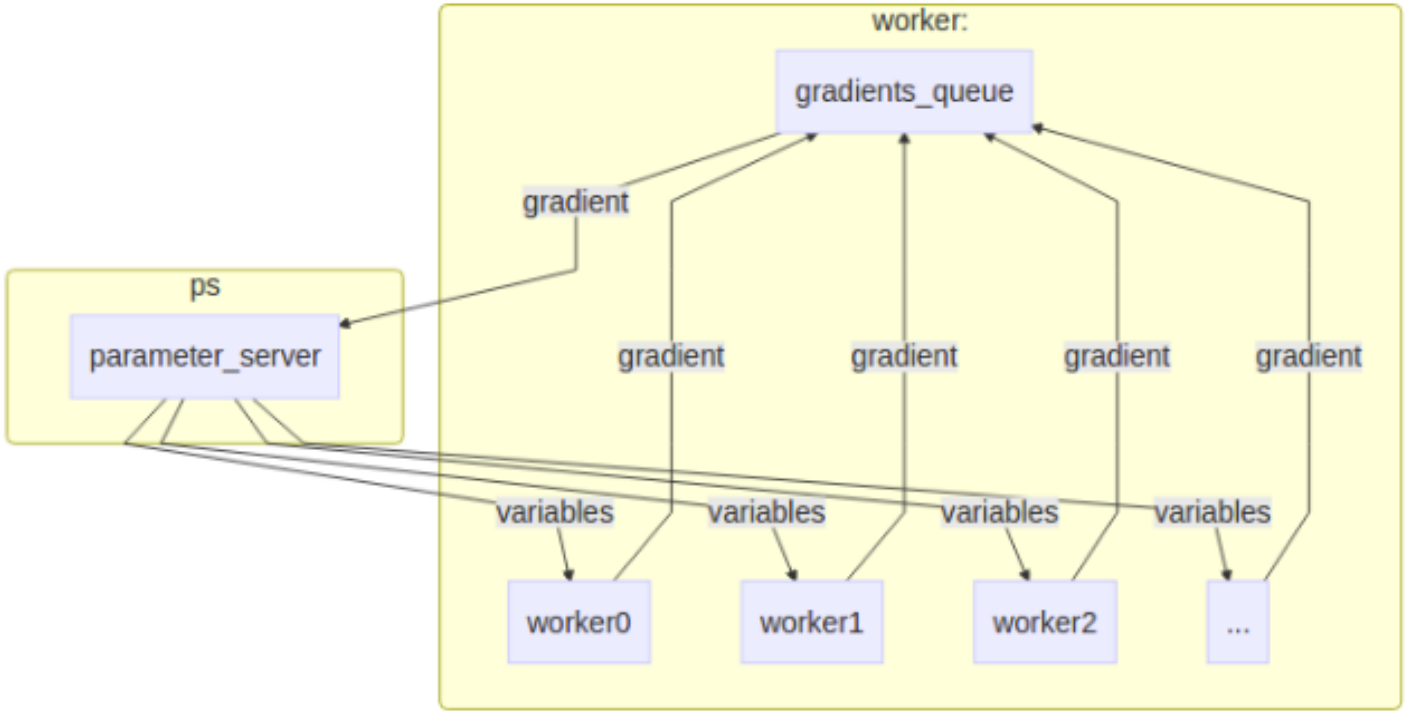
- **小型規模訓練**，這種資料與參數量不多時，可以用一個 CPU 來同時執行兩種任務。
- **中型規模訓練**，資料量較大，但參數量不多時，計算梯度的工作負載較高，而參數更新負載較低，所以計算梯度交給若干個 CPU 或 GPU 去執行，而更新參數則交給一個 CPU 即可。
- **大型規模訓練**，資料與參數量多時，不僅計算梯度需要部署多個 CPU 或 GPU，連更新參數也要不說到多個 CPU 中。

副本與訓練模式

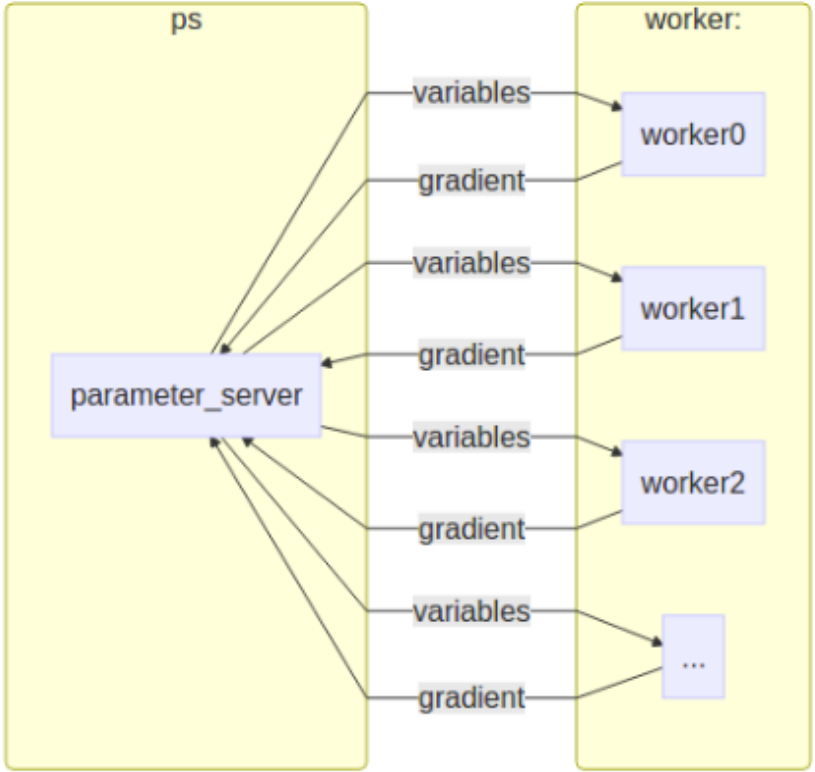
- Replicated
  - In-graph
  - Between-graph
- Training
  - Synchronous
  - Asynchronous



Between-graph Synchronous

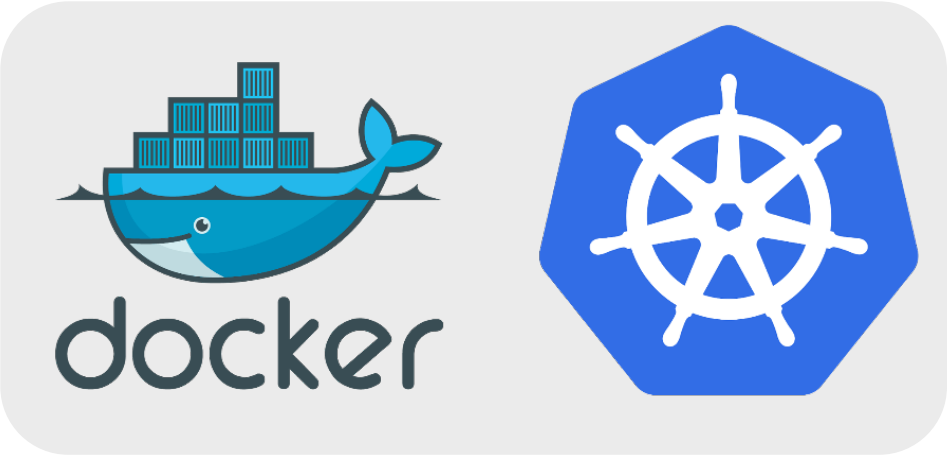


Between-graph Asynchronous



Kubernetes

Kubernetes 是 Google 開源的容器(Container)分散式管理系統，是建於 Docker(或 OCI 標準容器 Runtime 引擎)之上的容器叢集排程系統。

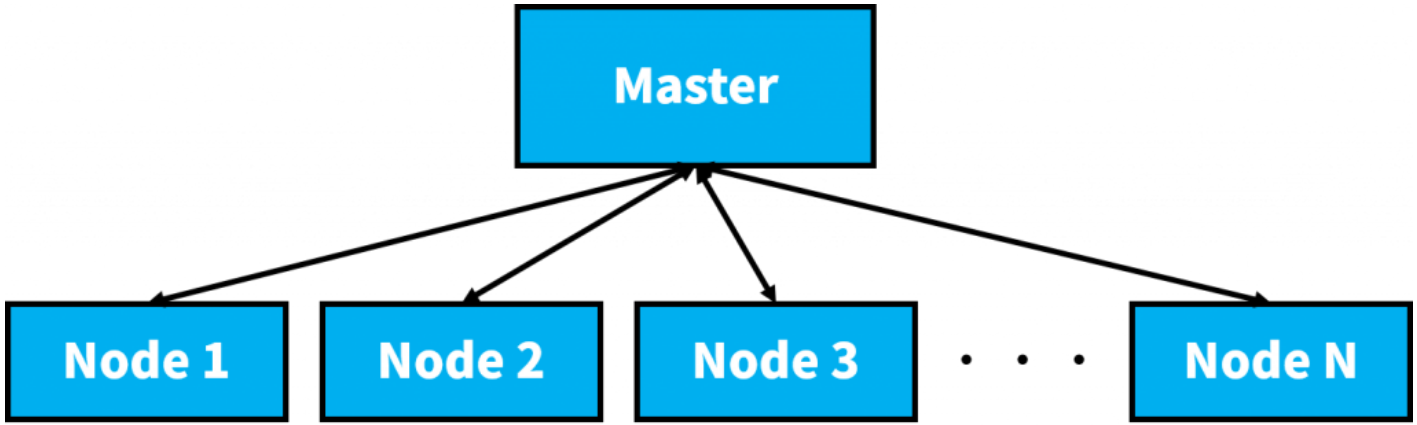


Kubernetes 是 Google **開源的容器(Container)分散式管理系統**，是Google十幾年以來大規模應用容器技術的經驗累積和昇華的一個重要成果，是建於 Docker(OCI標準容器)之上的容器叢集排程服務，簡稱為k8s(‘k’ + 8 letters + 's')。

Kubernetes 基本架構

Kubernetes 屬於分散式架構系統，主要由兩種節點角色組成：

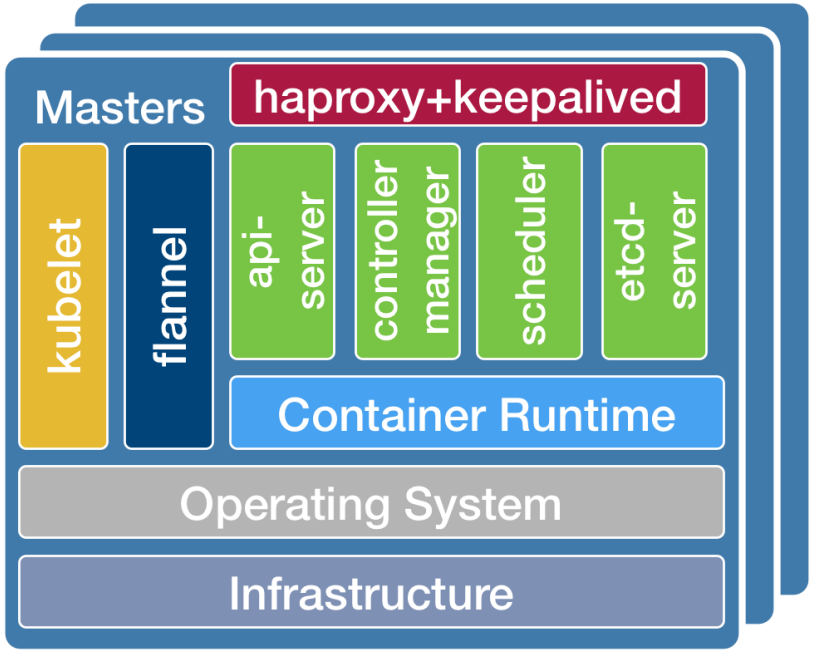
- **Master** – 主要工作為提供 API 與管理工作節點，可視為**主節點**。
- **Node(Minion)** – 主要**執行應用程式的節點**，上面會執行許多容器。



Kubernetes Master

Kubernetes Master 包含了四個基本組件:

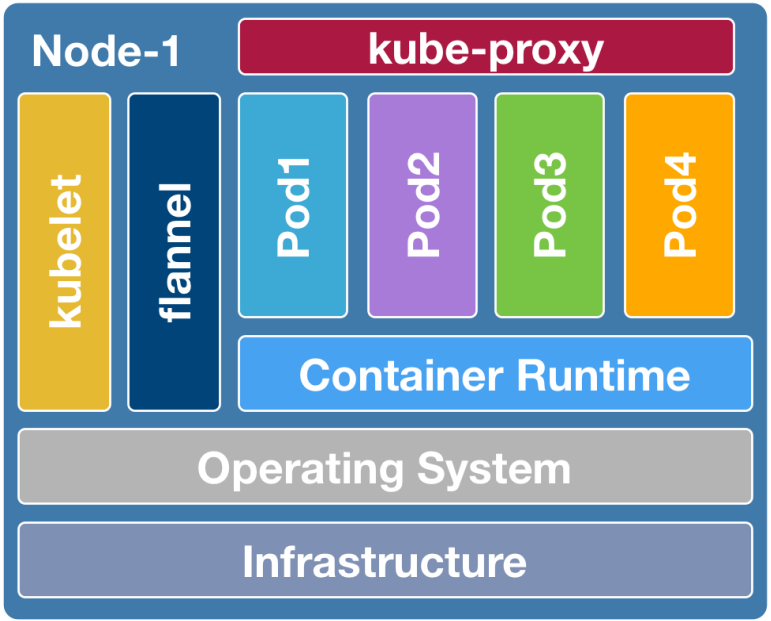
- **Etcd**: 是 CoreOS 團隊發起的一個管理設定資訊和服務發現(service discovery)的專案。
- **API Server**:以 REST APIs 介面方式提供所有業務邏輯CURD操作。
- **Controller Manager Server**:所有其他叢集級功能都是透過控制管理器(Controller Manager)來操作。
- **Scheduler**:負責整個分散式系統的資源排程。



Kubernetes Node(Minion)

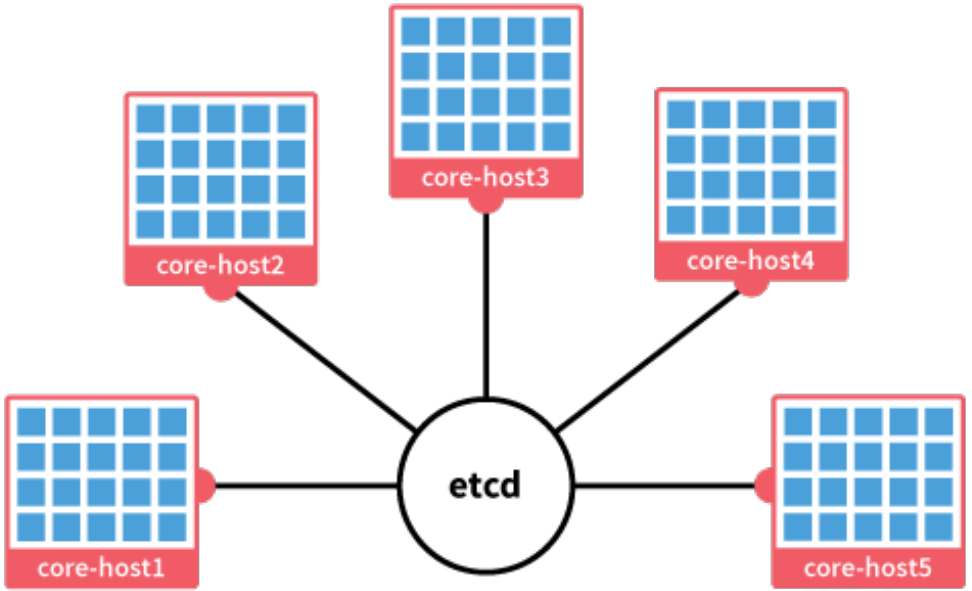
Kubernetes Node 包含了四個基本組件:

- **Kubelet**:負責管理的映像檔、容器與資料 Volume等操作。也是連接 Master 的橋樑。
- **Kube-proxy**: 為了解決外部網路能夠群曲跨機器叢集中容器提供的應用服務而設計。支援 TCP,UDP stream forwarding 或 round robin TCP,UDP forwarding。
- **Container**:基於 Docker engine 來執行應用程式容器實例。



Etcd - Distributed reliable key-value store

- 類似目錄樹狀結構
- JSON/REST API
- 使用 Discovery URL
- 使用 Raft 一致性演算法提供容錯與高可靠



Kubernetes Storage

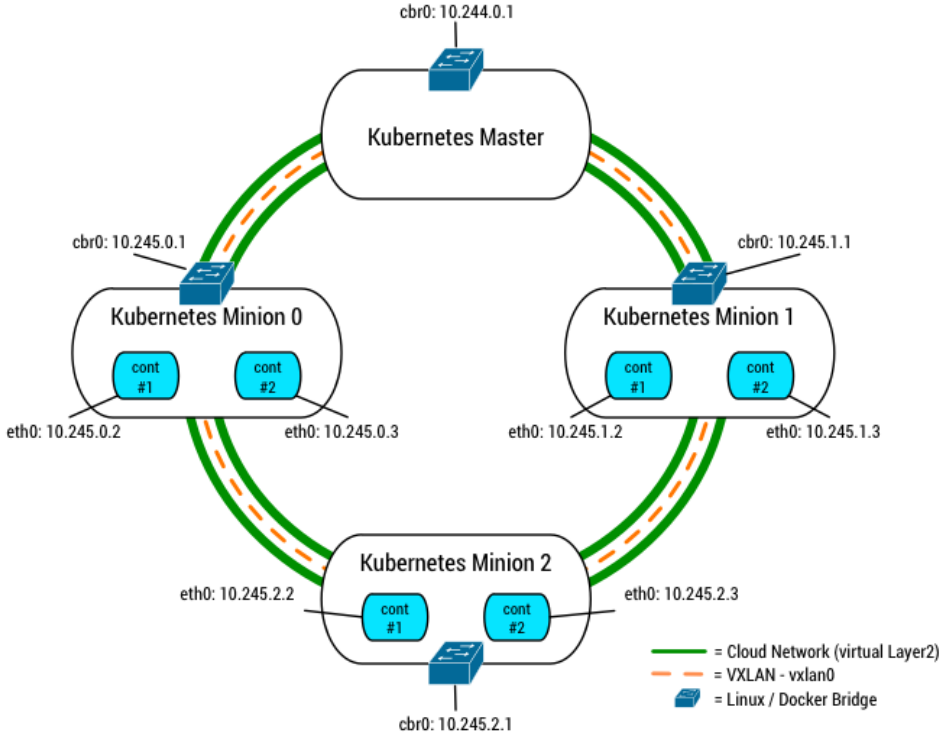
- Volumes
- Persistent Volumes
- 支援多種儲存 plugin

Volume Plugin	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
AWSElasticBlockStore	✓	-	-
AzureFile	✓	✓	✓
AzureDisk	✓	-	-
CephFS	✓	✓	✓
Cinder	✓	-	-
FC	✓	✓	-
FlexVolume	✓	✓	-
Flocker	✓	-	-
GCEPersistentDisk	✓	✓	-
Glusterfs	✓	✓	✓
HostPath	✓	-	-
iSCSI	✓	✓	-
PhotonPersistentDisk	✓	-	-
Quobyte	✓	✓	✓
NFS	✓	✓	✓
RBD	✓	✓	-
VsphereVolume	✓	-	-

Kubernetes Network

Overlay / Underlay Network:

- Flannel
- Weave
- Calico



Kubernetes 有什麼好處？

Kubernetes 管理跨區域與主機的容器節點，提供基本部署、維運、管理，以及執行各項應用程式。Kubernetes 為IT人員帶來以下幾項好處：

- 管理與操作**簡單**，不需要太多複雜元件設定，且支援 Heapster/Grafana/Influx 監控服務。
- 提供Controller與Scheduler管理叢集以及應用程式**資源**與**調度**。
- 簡單地擴展、**遷移**與**升級**Kubernetes元件。
- 提供負載平衡、容錯、Namespace、Auto scale與本地讀寫等功能。
- 支援各種雲端平台部署與作業系統，**No vendor lock-in**。
- 支援 Federation 與 Hybrid。
- Community and Enterprise Support。

部署測試叢集

- 安裝 Vagrant 與 VirtualBox，然後透過下載 ha-kube-ansible 執行指令：

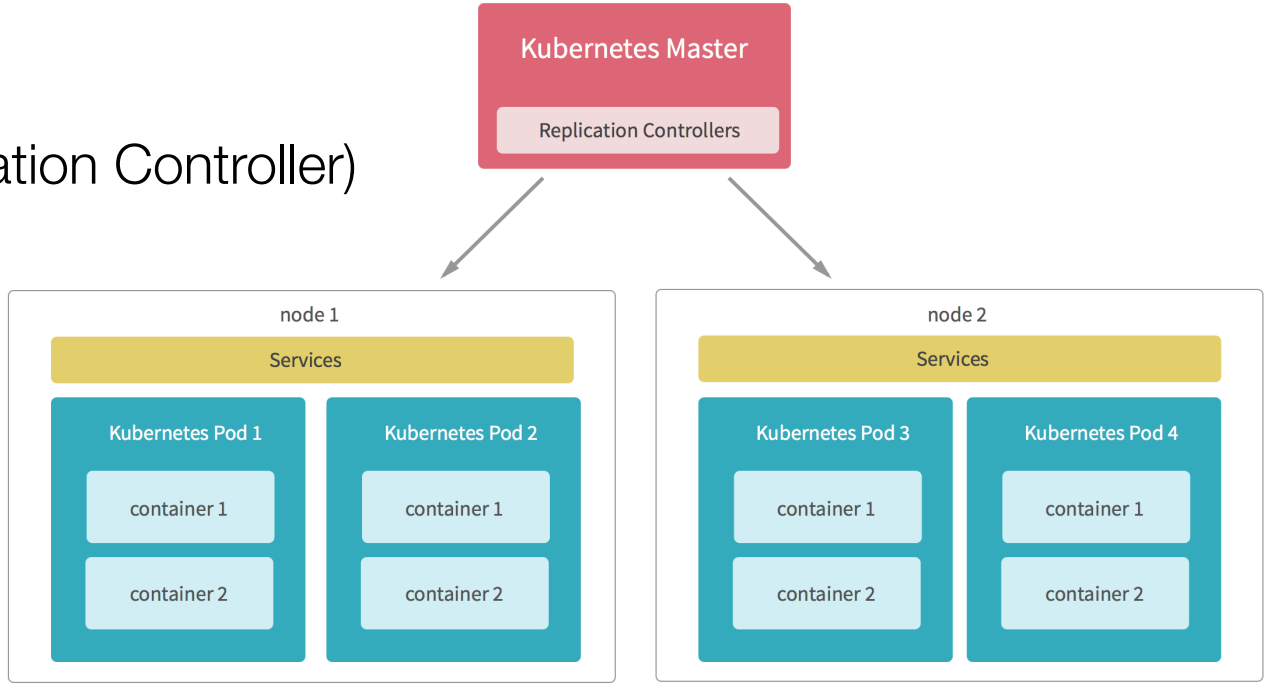
```
$ git clone https://github.com/kairen/ha-kube-ansible.git -b dev

$ cd ha-kube-ansible

$ ./setup-vagrant -b 1 -n 3 -c 1 -m 4096
Cluster Size: 1 master, 3 node.
    VM Size: 1 vCPU, 4096 MB
Start deploying?(y): y
....
$ vagrant ssh master1
```

其他部署工具

- kubeadm
- kargo
- Kops
- Kube-aws
- Bootkube
- Kismatic
- Terraform
- Supergiant
- kubebox
- LazyKube

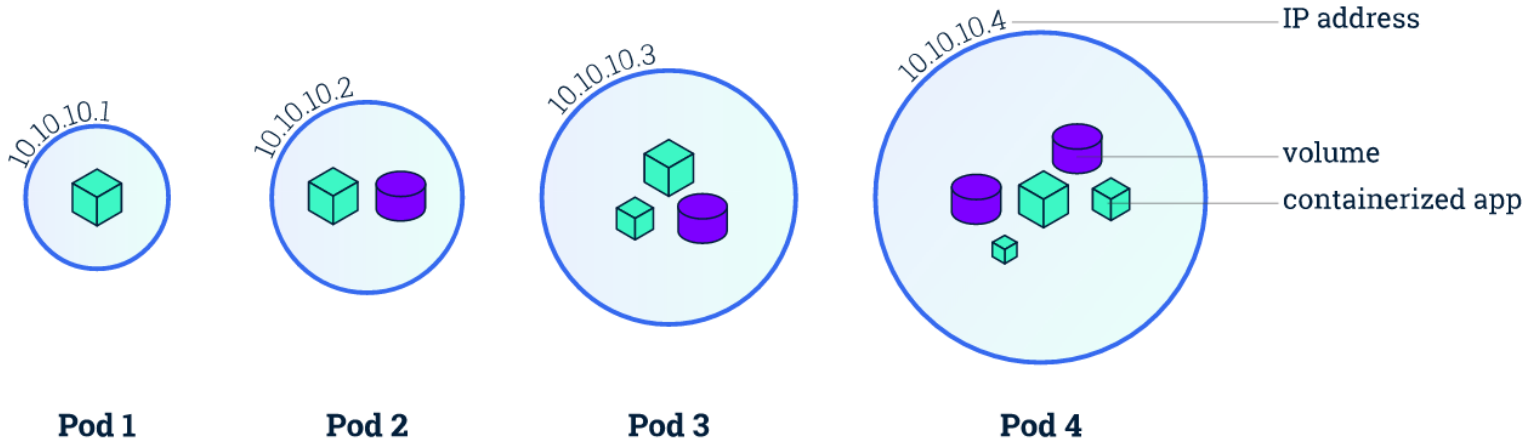
Kubernetes 重要的三個概念

- Pod
- Service
- Deployments (Replication Controller)



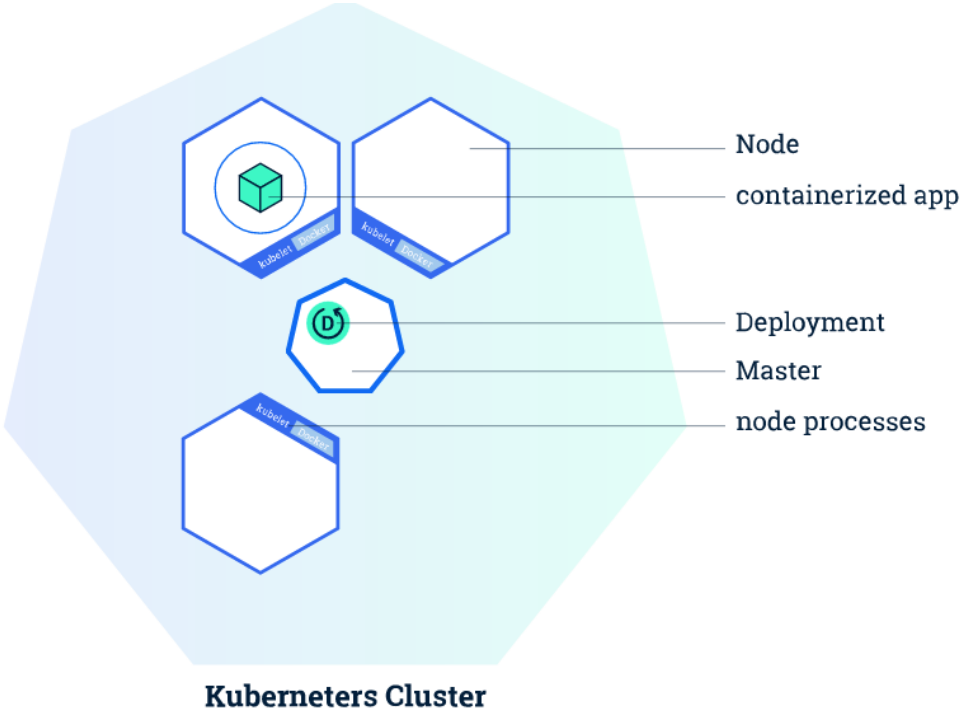
Kubernetes - Pod

Pod 是 K8s 中最小的部署單位，可以將**一個或多個容器**組成一個 Pod，Pod 所包含的容器只會運行在同一個 Host 上，並共享 network namespaces、IP port，每個 Pod的 network mapping 都是由 container pause 所封裝處理的。



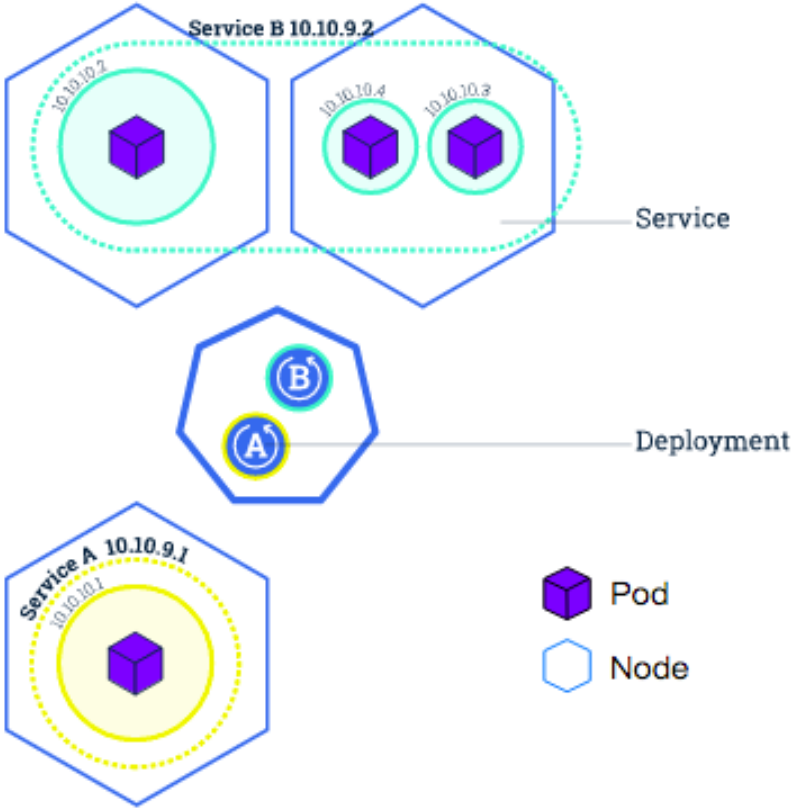
Kubernetes - Deployments

Deployments(Replication Controllers)是部署群組 Pods，其會**確保 K8s 系統中指定數量副本的 Pod 存活**， 也可以利用 Label 及 Rolling update 來進行副本的擴展或更新。



Kubernetes - Service

Service 是 pod 服務的抽象層，俱有**負載平衡**的功能，由於 pod 的數量與IP不會固定，可將一群 pod 指派成 service 並給予一個固定IP (cluster IP)，service 會將請求至 cluster IP 的流量導入其對應 pod。



Why TensorFlow + Kubernetes?

TensorFlow on Kubernetes

- **叢集管理**
  - 透過 Kubernetes DNS 機制來解析伺服器位址
  - 透過 RC 來管理故障重啟問題
  - Kubernetes 提供 Monitoring 與 Logging 等功能
  - 可以使用 CPU 與 GPU 排程來指定節點

TensorFlow on Kubernetes

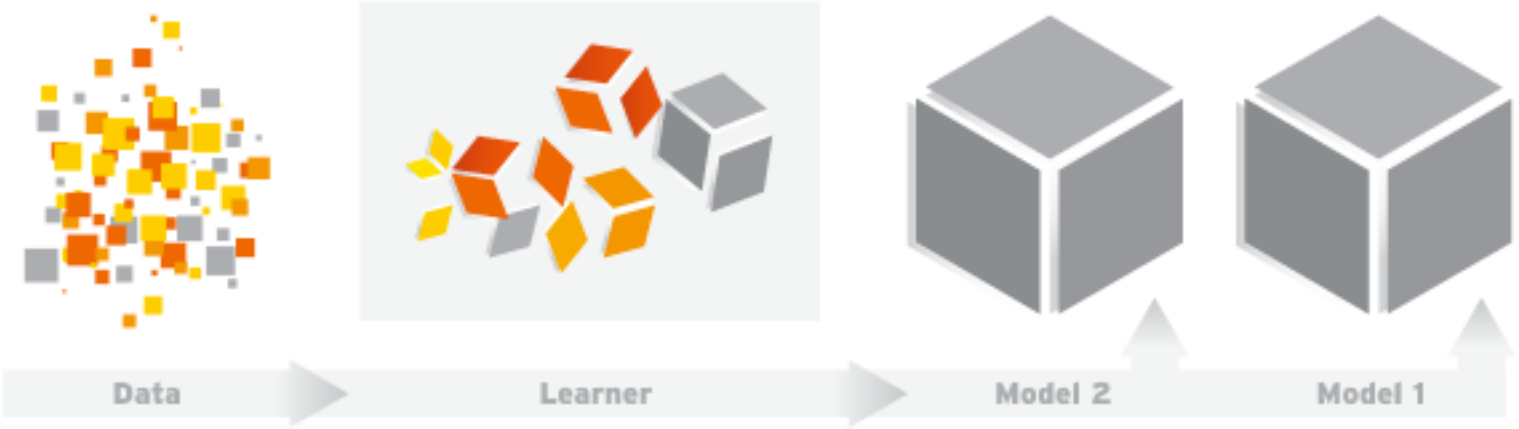
- **行程生命週期管理**
  - 解決目前 TensorFlow 不會自動結束問題
  - 解決無法區分正常完成還是故障退出等問題
  - 解決需要手動管理行程
- **共享儲存解決方案**
  - 可使用多種儲存服務，如 NFS、Ceph 等分散式儲存系統

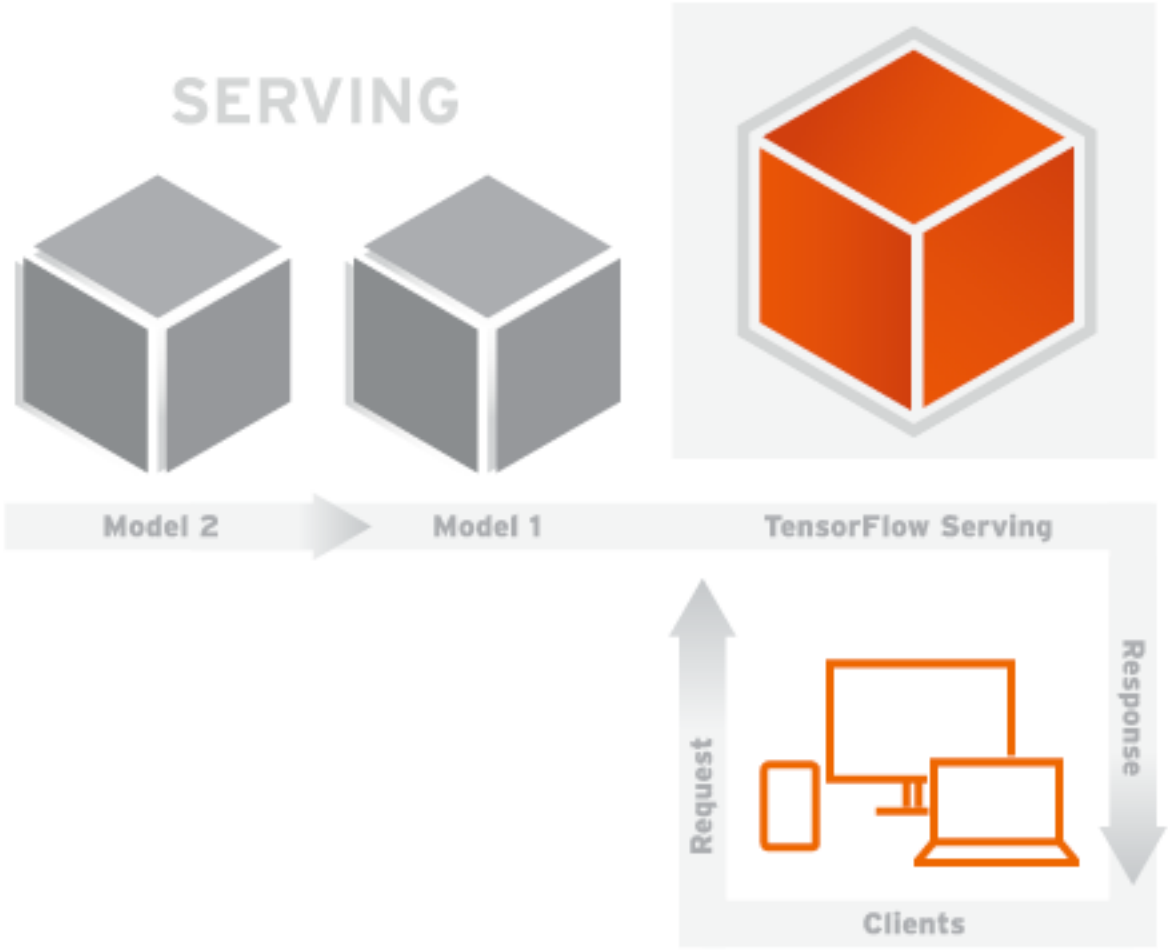
TensorFlow on Kubernetes Lab

Go 來跑幾個 Lab 吧!!!

git clone https://github.com/kairen/workshop413.git

CONTINUOUS TRAINING PIPELINE




SERVING
Model 2
Model 1
TensorFlow Serving
Request
Response
Clients

Thank You!

迎棧科技股份有限公司

inwinSTACK
G.O.D. Great OpenStack Delivery

www.inwinstack.com