# 雲端運算服務
## Cloud Computing Service

## 單元02 雲端運算核心技術

蘇維宗 (Wei-Tsung Su)
suwt@scu.edu.tw
H307-3

# Revision

| Rev. | Description | Date | Authors |
|---|---|---|---|
| v1.0 | Baseline | 2023/1/10 | 蘇維宗 |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# 單元大綱

- 虛擬化技術
  - 虛擬機
  - 容器化技術
  - 無伺服器架構
- 平行運算技術

# 虛擬化技術

# 雲端運算軟硬體

- 軟體可依據需求存取、依據用量收費
  - anyone with any device, ease to use, on-demand
  - 共享與協同合作
- 硬體可當使用者需要的時候，隨時要求
  - 增減伺服器數量(vertical scaling)
  - 增減伺服器規格(horizontal scaling)
- 關鍵技術
  - **虛擬化技術(virtualization)**

# 虛擬化技術

- Cloud computing is an example of computing in which **dynamically scalable** and often **virtualized resources** are provided as a service over the Internet.

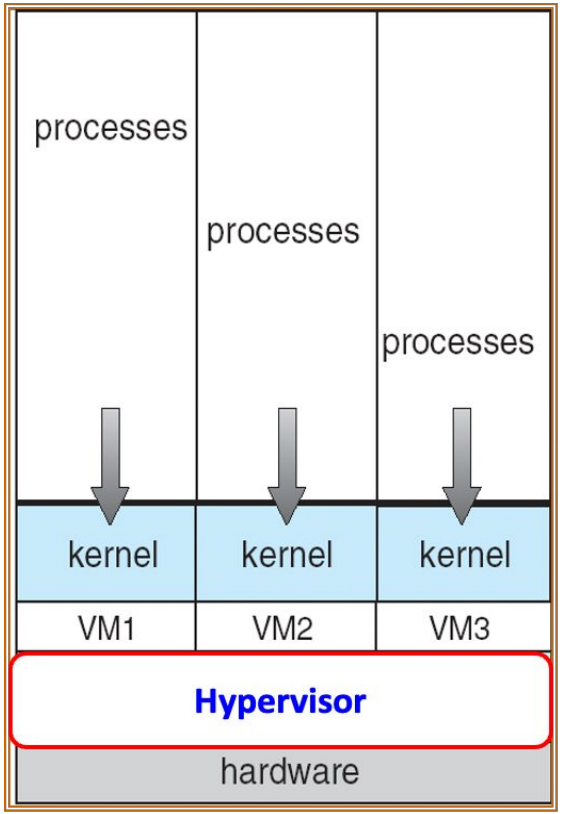
***Source:*** *Wikipedia*

- Everything as a service
  - 軟體、平台、基礎架構、網路、資訊安全、...
  - 用多少算多少
- 虛擬化技術被廣泛運用在雲端運算中

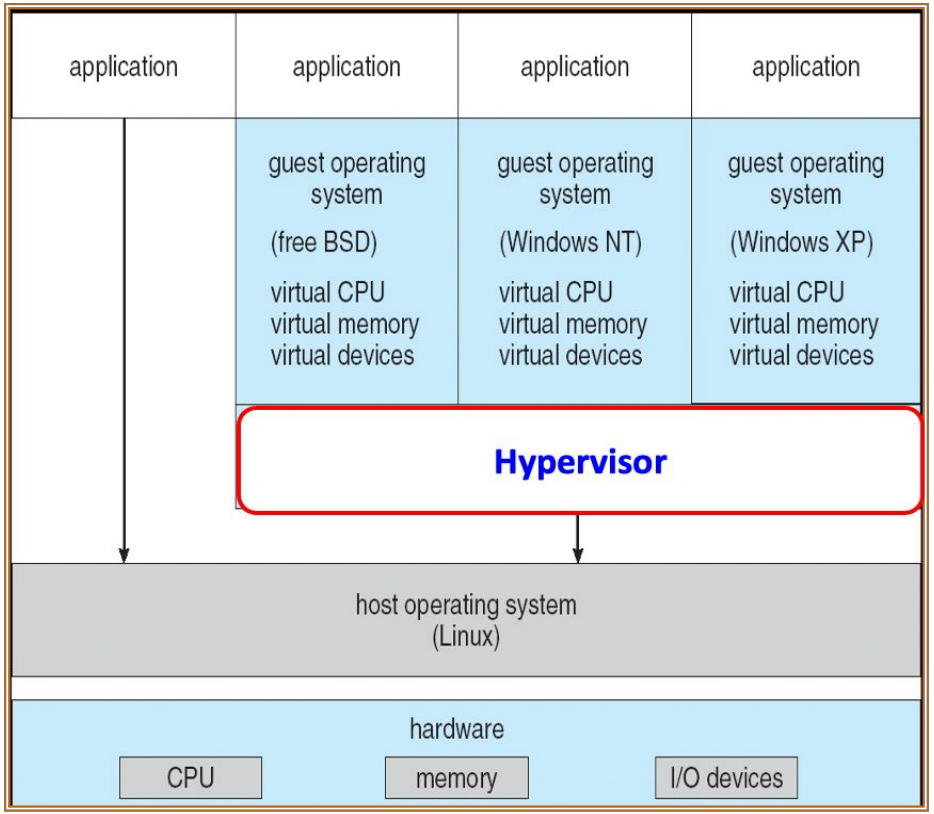
# 虛擬化技術類型

- Hardware Abstraction Level Virtualization
  - 虛擬機(virtual machine)技術
- Operating System Level Virtualization
  - 容器(container)技術
  - 無伺服器運算(serverless computing)技術

**Source:** Operating System Concepts 8/e, John Wiley, 2009.



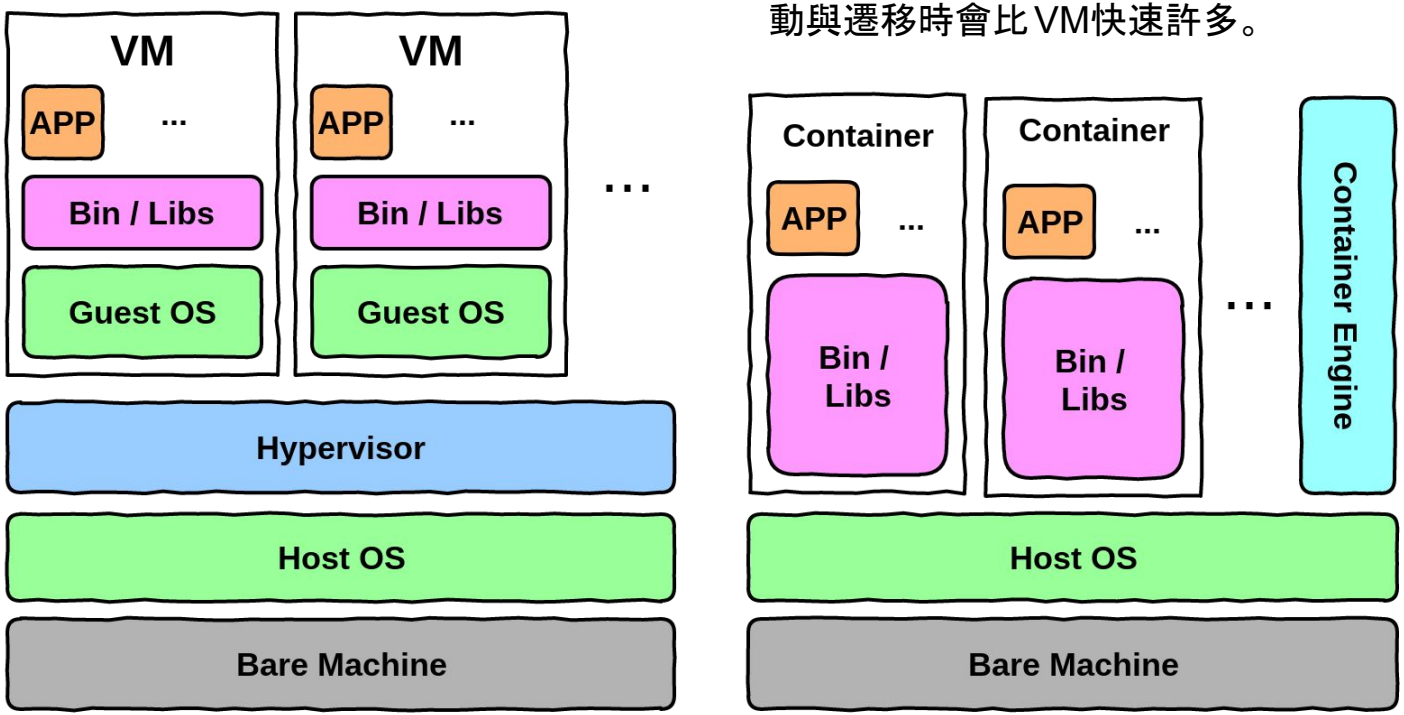

Type 1




Type 2


Hardware Abstraction Level Virtualization

Container有獨立的執行環境與函式庫，但是因為共用Host OS的核心，所以在啟動與遷移時會比VM快速許多。



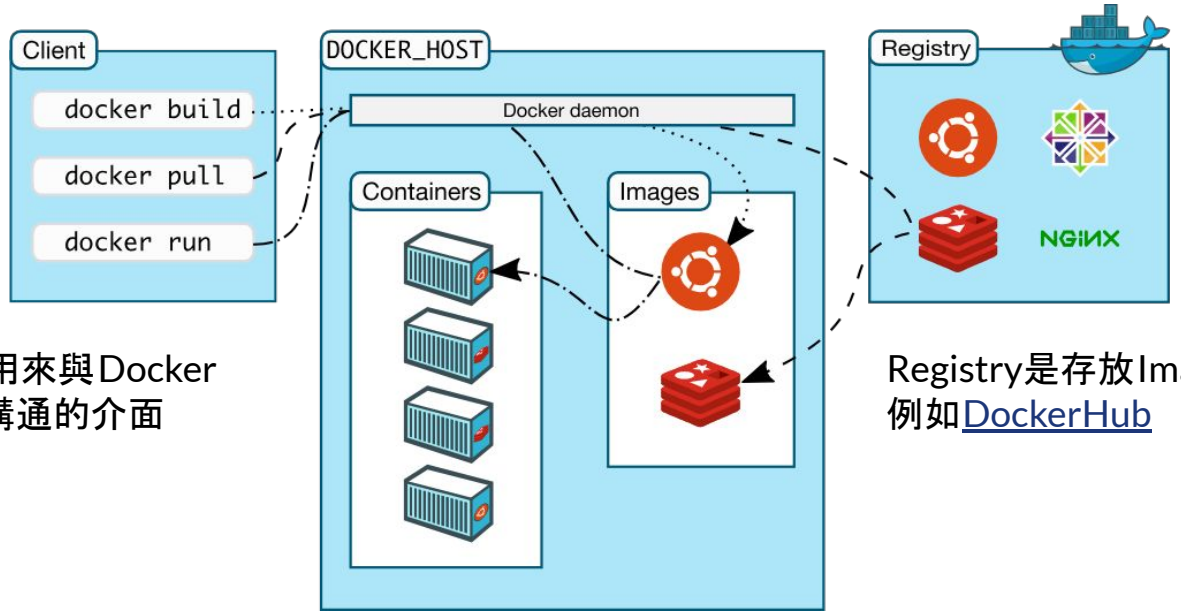

VM (Type II)

Container

Operating System Level Virtualization (Container)

# Docker Architecture

**Step 1.** 先從Registry下載Image到Host中
**Step 2.** 從Image建立Container



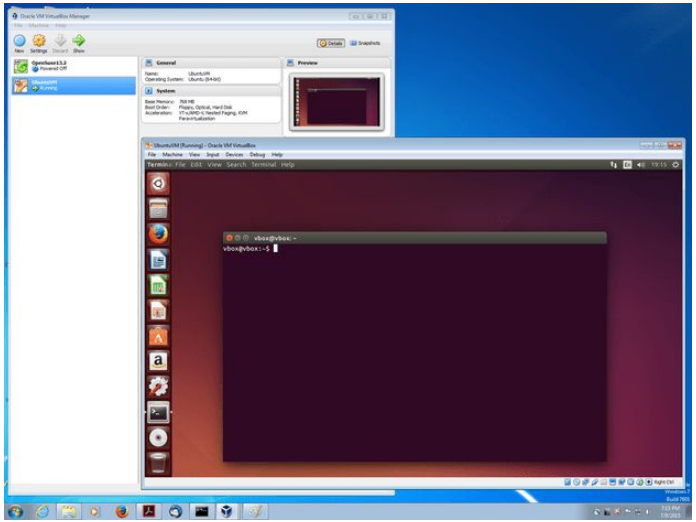

Client是用來與Docker daemon溝通的介面

Registry是存放Image的地方，例如DockerHub

**圖片來源:**https://docs.docker.com/get-started/overview/

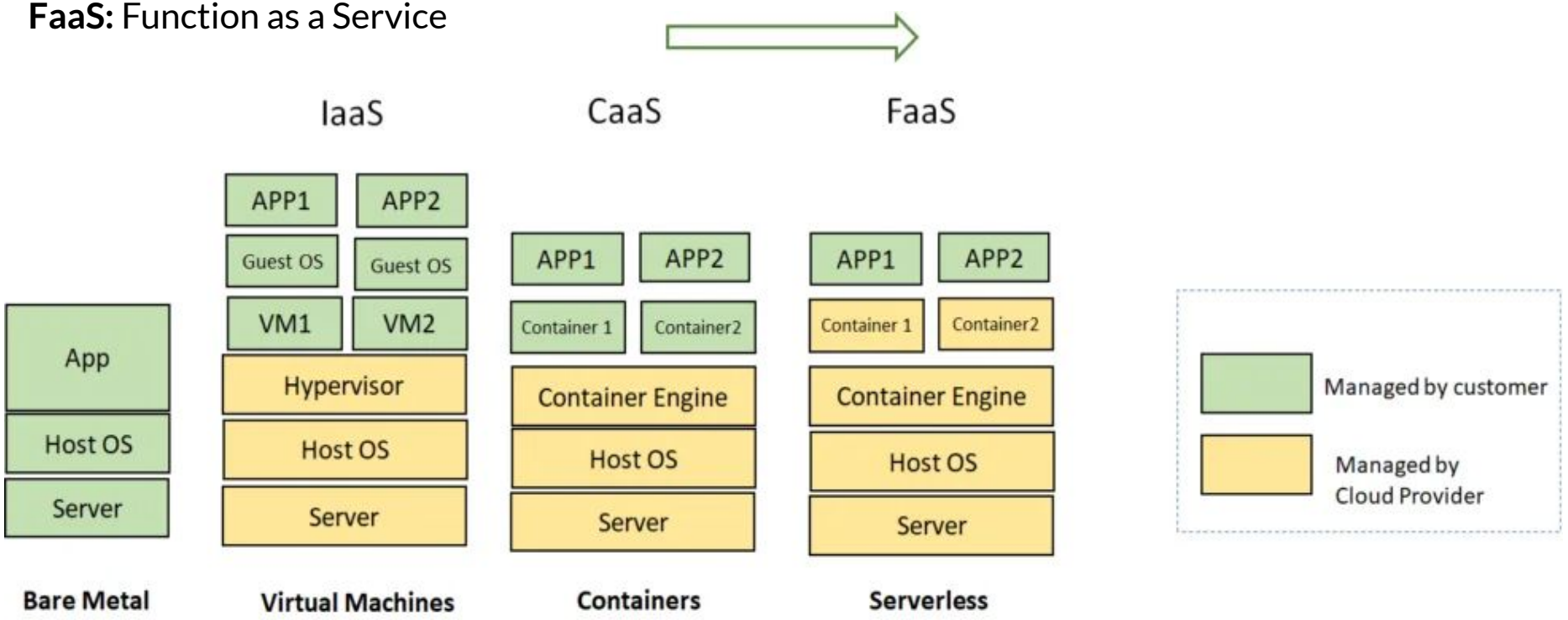
# 虛擬化技術產品

- Hardware Abstraction Level Virtualization
  - **Type 1:** VMware ESXi, Xen, Hyper-V
  - **Type 2:** VirtualBox, VMware WorkStation
- Operating System Level Virtualization
  - Docker
  - Linux-VServer
  - Oracle Solaris Containers
  - FreeBSD Jails



**Ubuntu Linux on Windows VirtualBox**
**圖片來源:**
https://www.virtualbox.org/attachment/wiki/Screenshots/Ubuntu_14.04_on_Windows_7.png

**IaaS:** Infrastructure as a Service
**CaaS:** Container as a Service
**FaaS:** Function as a Service




**Source:** https://www.whiteboxsolution.com/blog/virtual-machine-vs-container-vs-serverless/

# 平行運算技術

# Google Cloud (1997)

# Google Cloud (2000)

# Google Cloud (2001)

# Google Cloud (201x)

# WHAT HAPPENS ON INTERNET IN 1 MINUTE


byteant

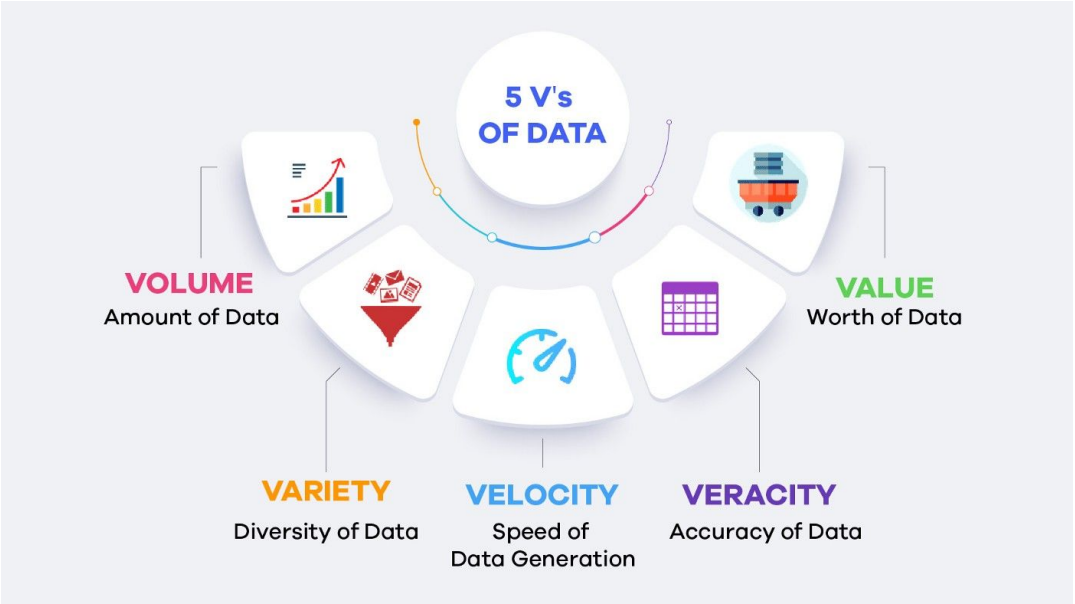
By @byte_ant
Source: https://www.statista.com/chart/17518/internet-use-one-minute/

# 大數據(Big Data)


**圖片來源**:https://1on1.today/blog/big-data-applications/

大數據有哪些特性?

# 大數據特性




**圖片來源**:https://medium.com/@get_excelsior/big-data-explained-the-5v-s-of-data-ae80cbe8ded1

# Big Data Processing in the Cloud

- "The data center is the computer"

*D.A. Patterson – EECS Professor in UCB (CACM 2008)*

- 挑戰
  - 如何有效運用資源?
  - 平行與分散式運算
- 如何設計雲端運算應用程式?
  - 新的軟體架構

# MapReduce

A useful programming model for cloud applications

- (G1) Mappers and Reducers, easy to understand and program.
- (G2) Suitable for large-scale data analysis
  - Common tasks in the cloud
  - Also has been found useful in machine learning
  - Active research area

# Mappers and Reducers

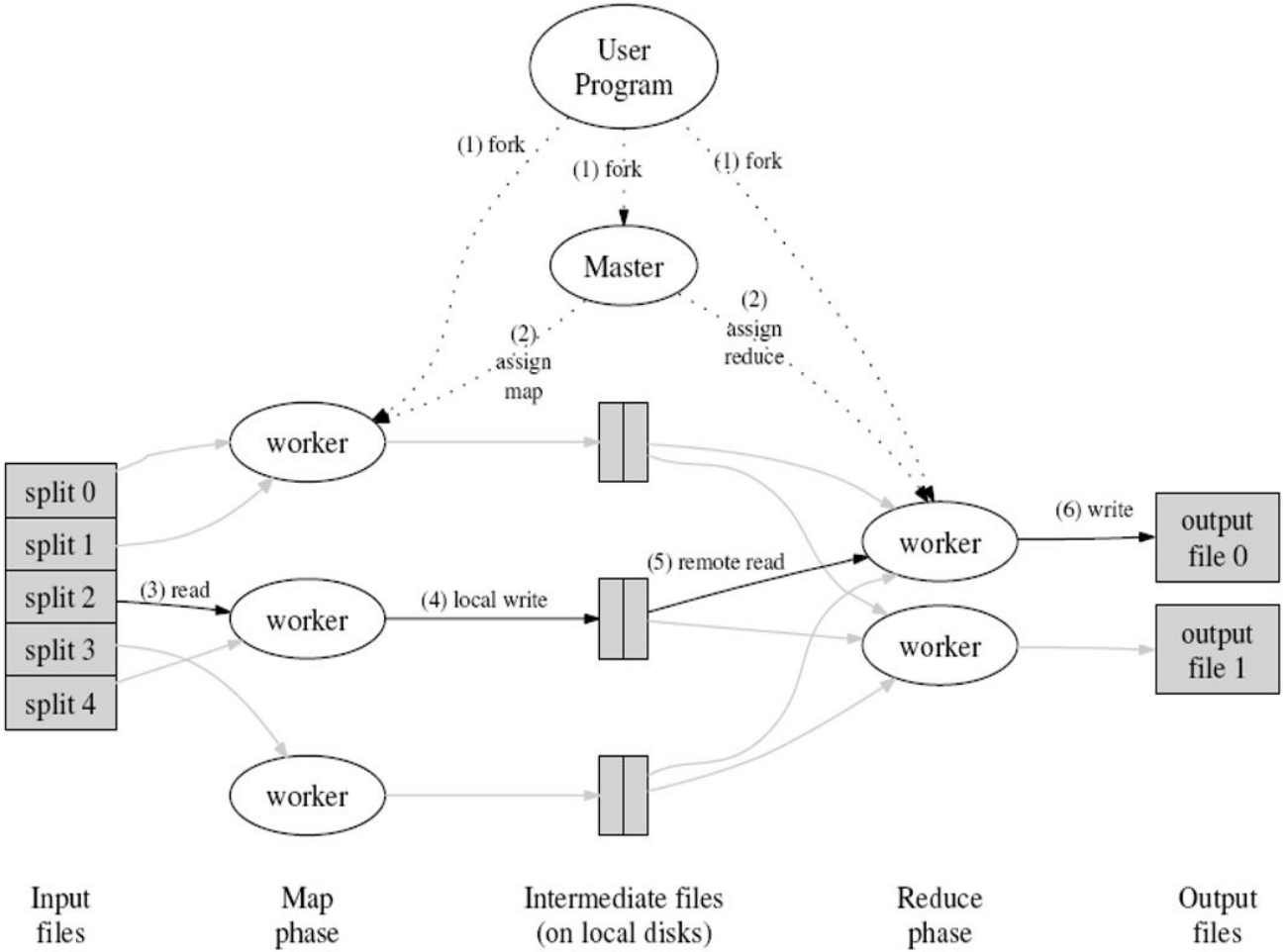
- MapReduce: many operations can be decomposed into a two-stage procedure: **Mappers** and **Reducers**.
- Mapper
  - Computation on individual input data record.
- Reducer
  - Summarize the mappers' output

# Mapper, Shuffler, and Reducer

- Mapper (user-program)
  - **Input:** one data record at a time
  - **Process:** extract/compute keys and values from the input record.
  - **Output:** (key, value) pairs
- Shuffler (build-in)
  - **Input:** (key, value) pairs from the mappers.
  - **Process:** group together values with the same key
  - **Output:** (key, [value1, value2, …]) pairs
- Reducer (user-program)
  - **Input:** (key, [value1, value2, …]) pairs
  - **Process:** compute aggregates of the list [value1, value2, …]
  - **Output:** (key, aggregation value) pairs.

# The Power of Shuffling

- Shuffling output:
  - (key, [value1, value2, …])
- Functionality
  - Group the values output by their keys
  - Sort the values when grouping
  - Allow computation of aggregated result, remove duplicates, etc.
- Standard process in the MR framework
  - Users get these functionalities with no cost

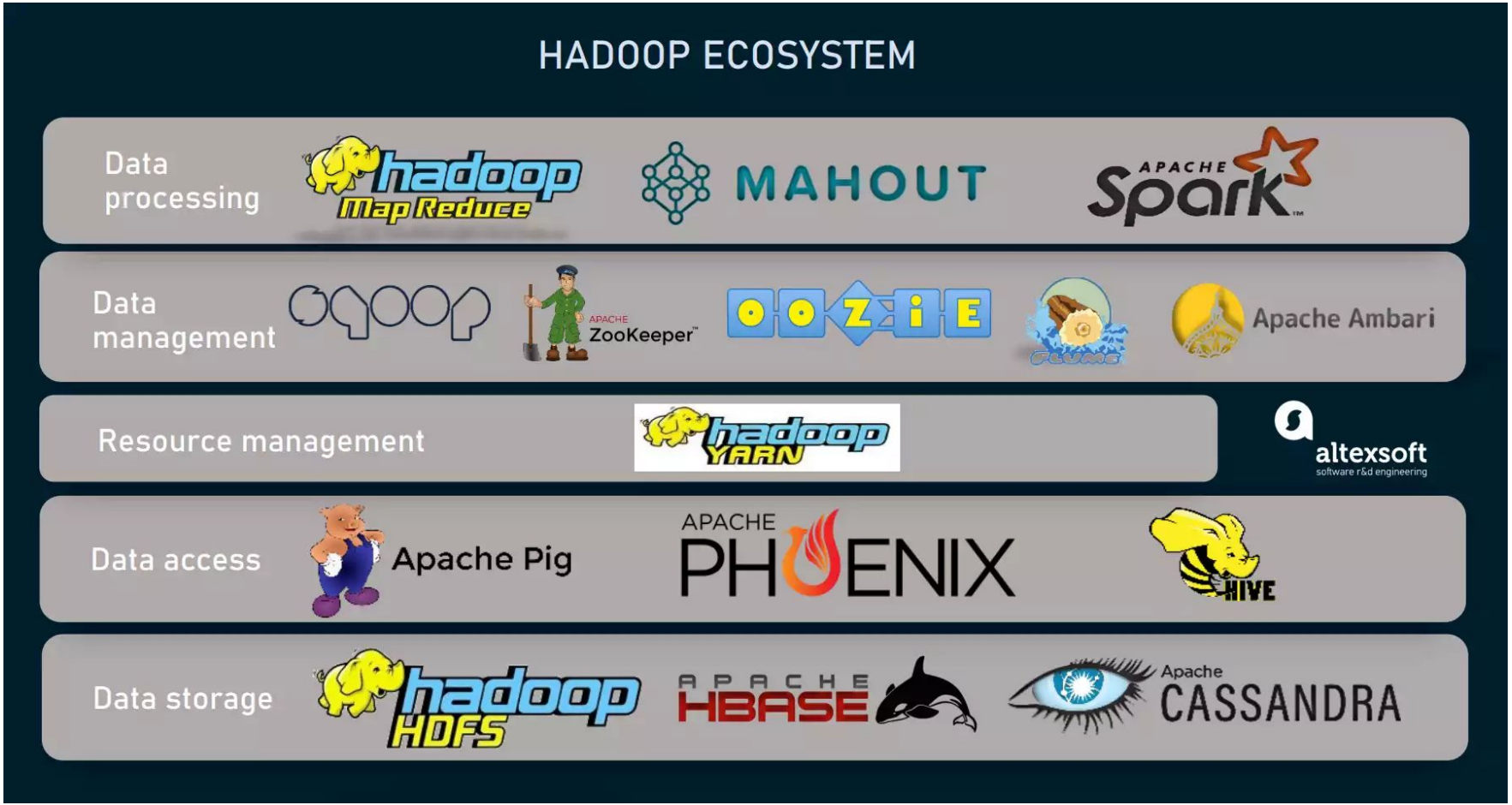

User Program
(1) fork
(1) fork
(1) fork
Master
(2) assign map
(2) assign reduce
worker
split 0
split 1
split 2
split 3
split 4
(3) read
worker
(4) local write
worker
(5) remote read
worker
worker
(6) write
output file 0
output file 1
Input files
Map phase
Intermediate files (on local disks)
Reduce phase
Output files

# Case Study

- Data set
  - A table of (student, class name) records.
- Application
  - Find students that are enrolled in classes where the class size is more than 30.
- Planning
  - (1) Find classes whose size is more than 30.
  - (2) Find students enrolled in these courses.

# Compute the Class Size

- Setup: 2 mappers, 1 reducer.
- Data
  - Partition 1: `(Ann, "OS"), (Bill, "PL")`
  - Partition 2: `(Cat, "OS"), (Dave, "OS")`
- Mapper output
  - Mapper 1: `("OS", 1), ("PL", 1)`
  - Mapper 2: `("OS", 1), ("OS", 1)`
- Shuffling output
  - `("OS", [1 1 1]), ("PL", [1])`
- Reducer output
  - `("OS", 3), ("PL", 1)`




Hadoop Ecosystem (**Source:** https://www.altexsoft.com/blog/hadoop-vs-spark/)

# Q & A


Computer History Museum, Mt. View, CA