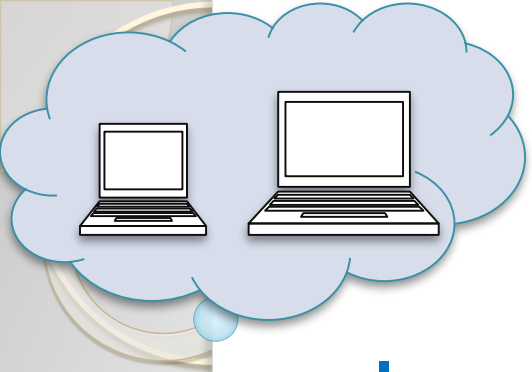


云计算入门

Introduction to Cloud Computing

GESC1001

Philippe Fournier-Viger

Professor

School of Humanities and Social Sciences

philfv8@yahoo.com

Fall 2020

QQ group: 1127433879

Course schedule

Part 1	Introduction and overview
Part 2	Distributed and parallel systems
Part 3	Cloud infrastructure
Part 4	**Cloud application paradigm (two lectures)**
Part 5	Cloud virtualization and resource management
Part 6 & 7	Cloud computing storage systems Cloud computing security
	Final exam

Introduction

Last week:

- **Chapter 3: mutual exclusion (**互斥) + cloud infrastructure (**e.g.** Amazon).

Today:

- Review
- **Chapter 4**: Cloud application paradigm (part 1)

3-CLOUD INFRASTRUCTURE (BRIEF REVIEW)

Introduction

Last week, we discussed the **infrastructure (基础结构)** of a popular **cloud provider (云提供商)** named **Amazon**.

It is one of the pioneers for the "*Infrastructure-as-a-service*" model

Introduction

- **Amazon Web Service (AWS - https://aws.amazon.com/).**
- It offers various services.
- One of the most popular is EC2.
- It allows to pay to use *virtual machines* (虚拟机) in the cloud.

What is a virtual machine (虚拟机)?

It is an application that works like a computer inside a real computer.

Two virtual machines (虚拟机)

Windows 10

回收站

jamesfriend.com.au/pce-js/

PCE.js Mac Plus emulator running Mac OS System 7 — a hack by James Friend

Mac Os 7

James Friend | PC...

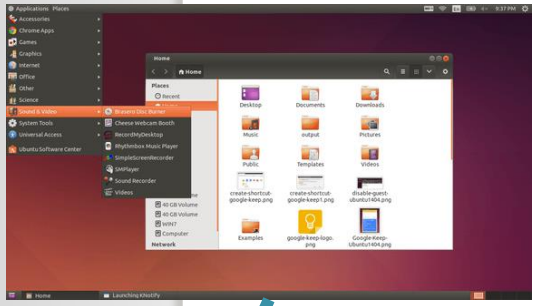
FRA 14:15 2019/10/11

Amazon EC2

- The user must create a **virtual machine image (虚拟机镜像)**
  (called **AMI**, Amazon Machine Image).
- It contains the operating system (操作系统) and the application(s) that the user wants to run.
- The user can start several virtual machines (instances - 实例) using an image.
- The user can stop a virtual machine.

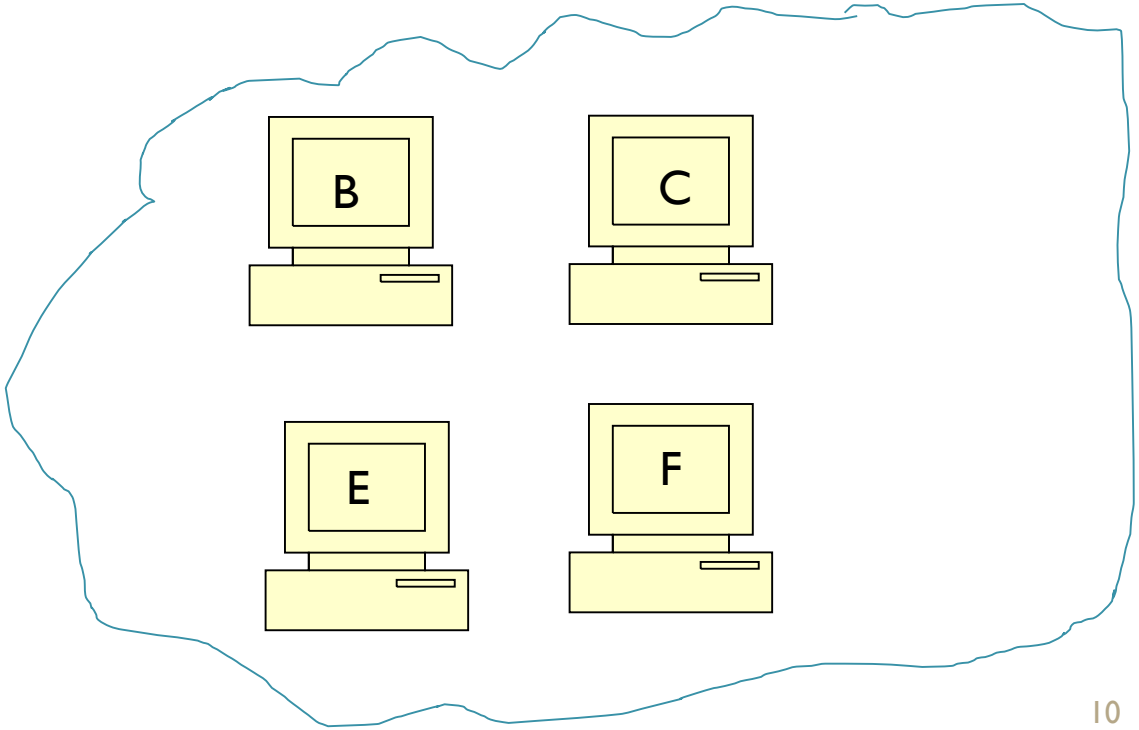
Illustration

An **image (虚拟机镜像)** is the state of a computer that has been saved into a file



User

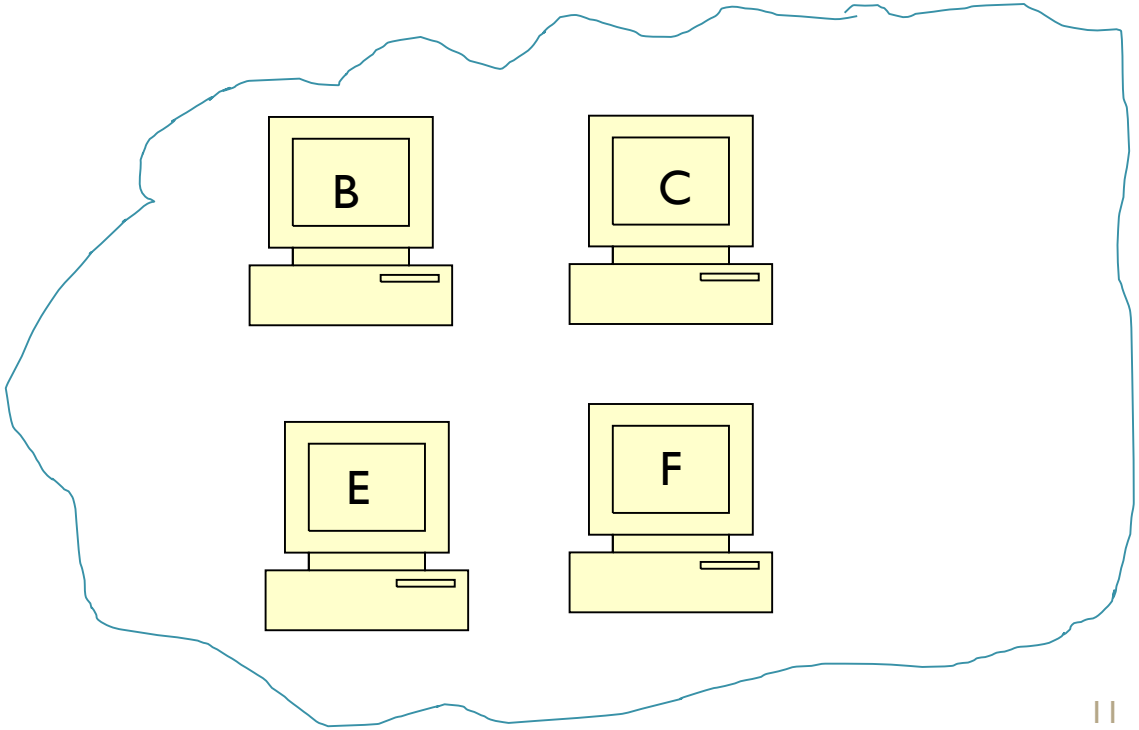
Image
AMI



Illustration

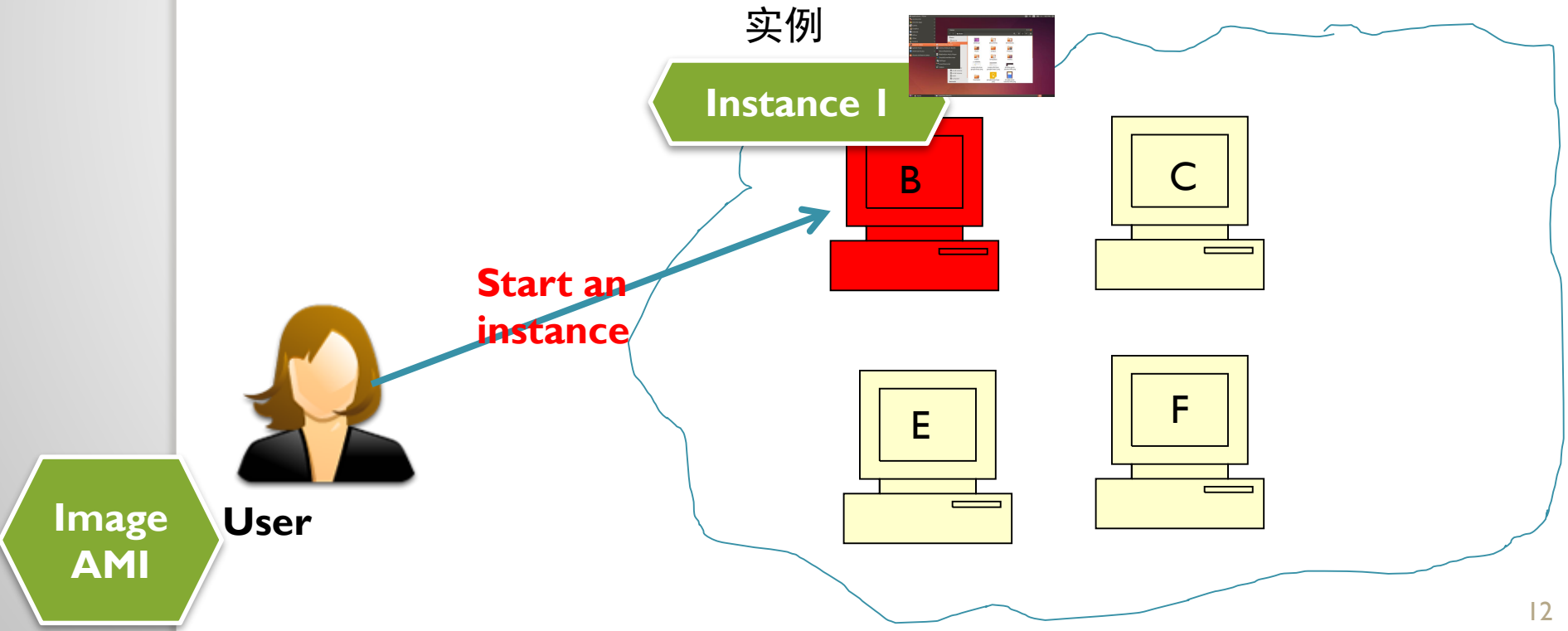
User

Image AMI



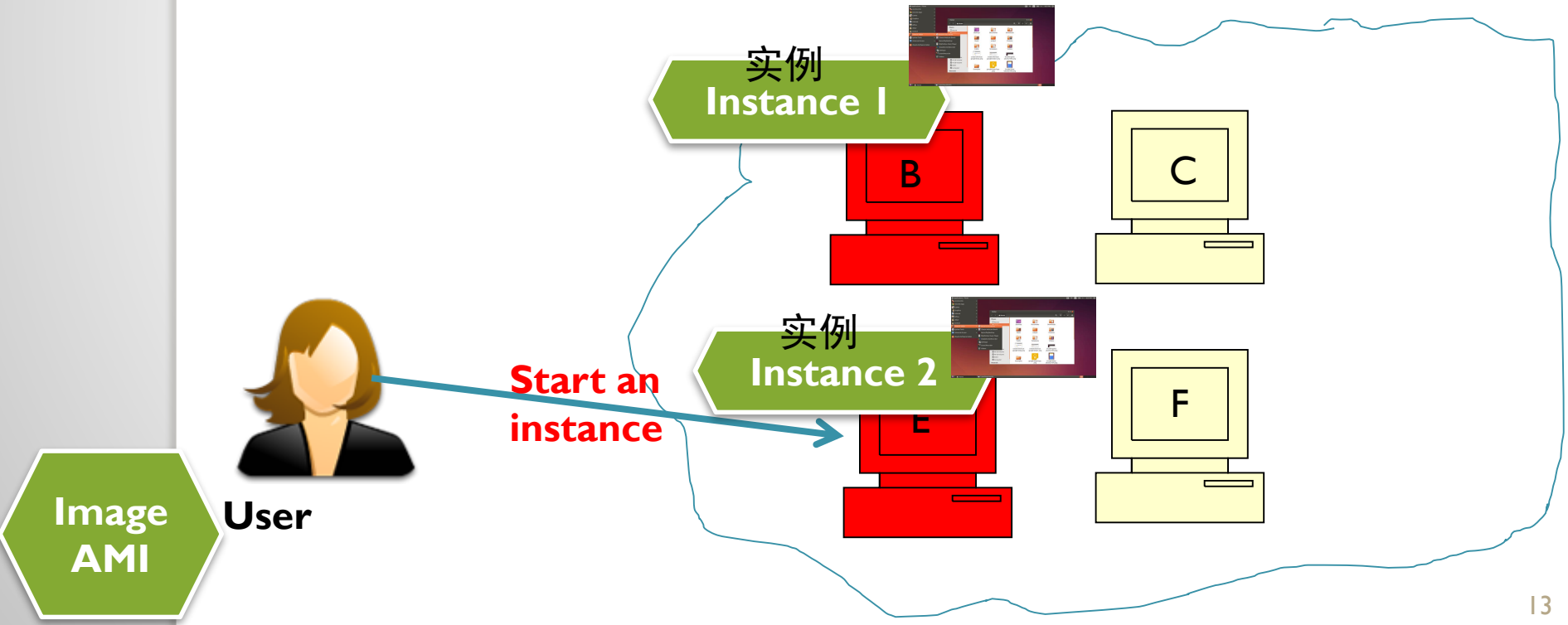
Illustration

An **image (虚拟机镜像)** can be used to start an **instance** in the cloud (a virtual machine虚拟机)



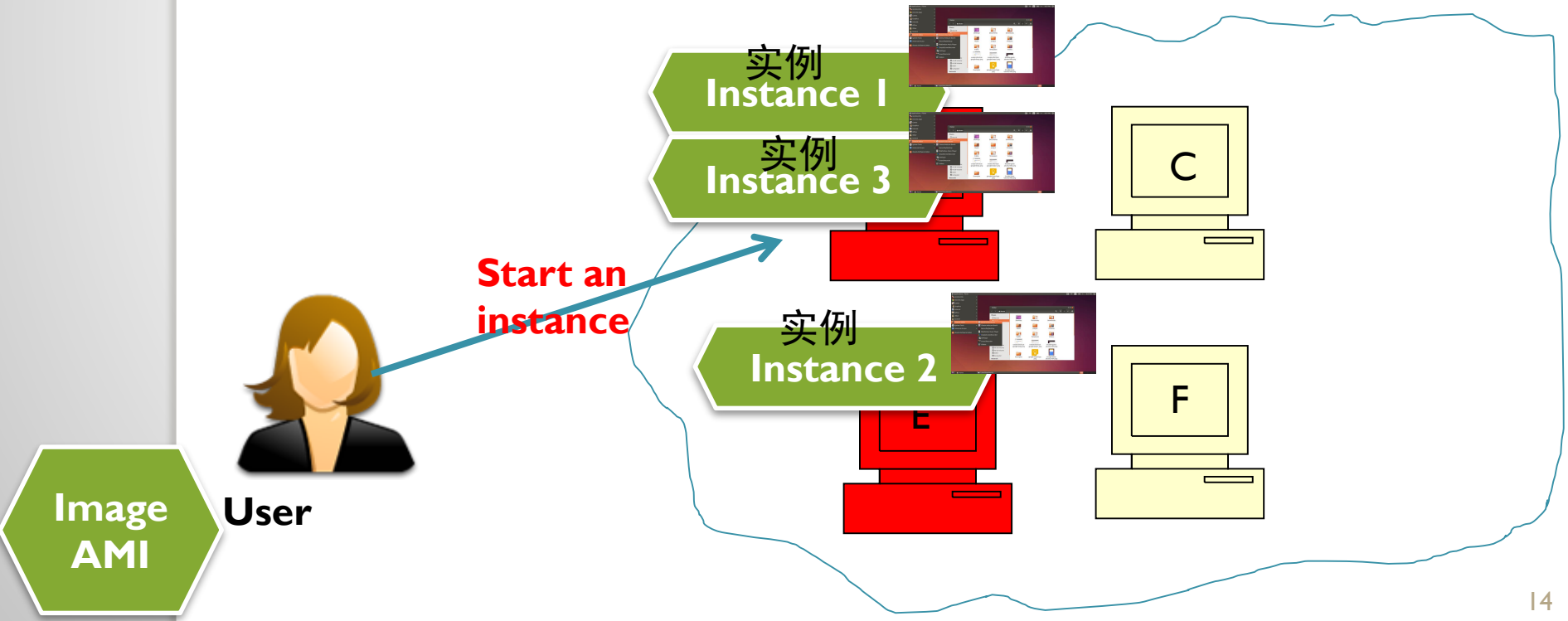
Illustration

The same image can be used to start many instances



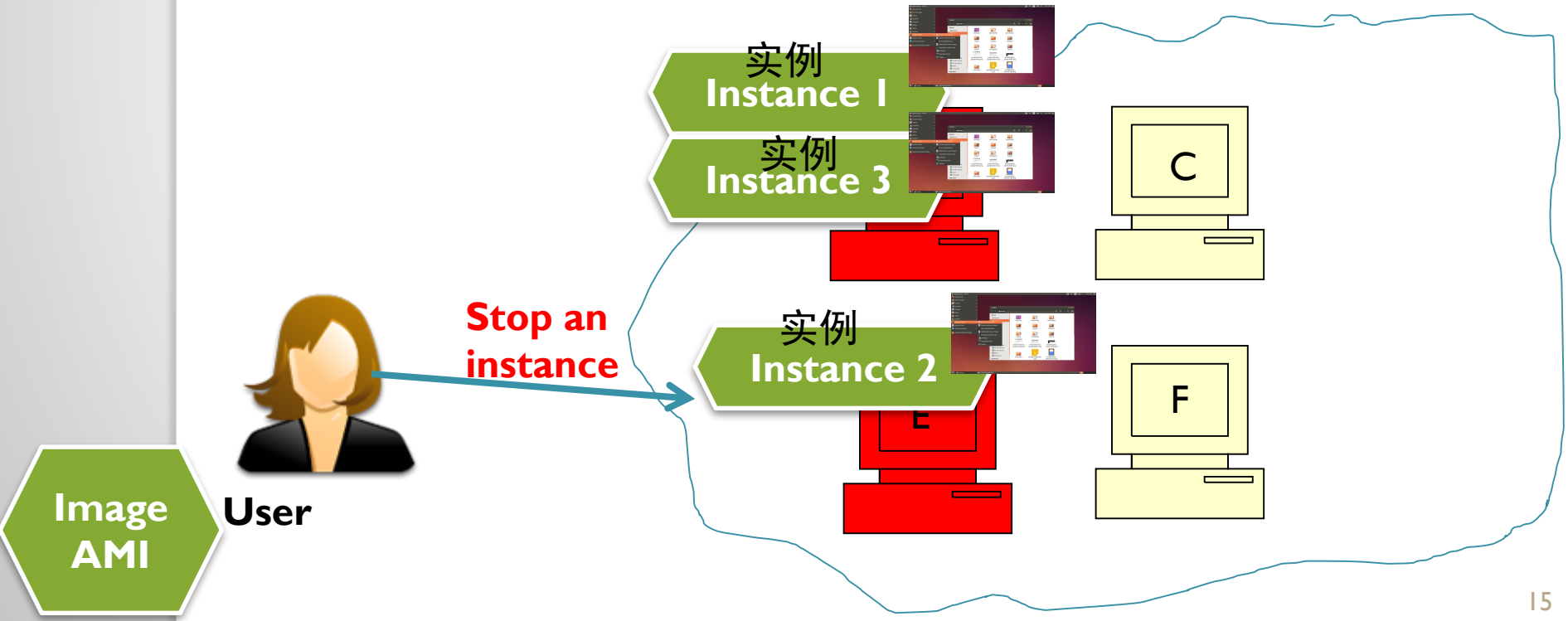
Illustration

Multiple instances can be run on the same computer



Illustration

The user can **stop** an instance.
The user can **restart** an instance.



Illustration

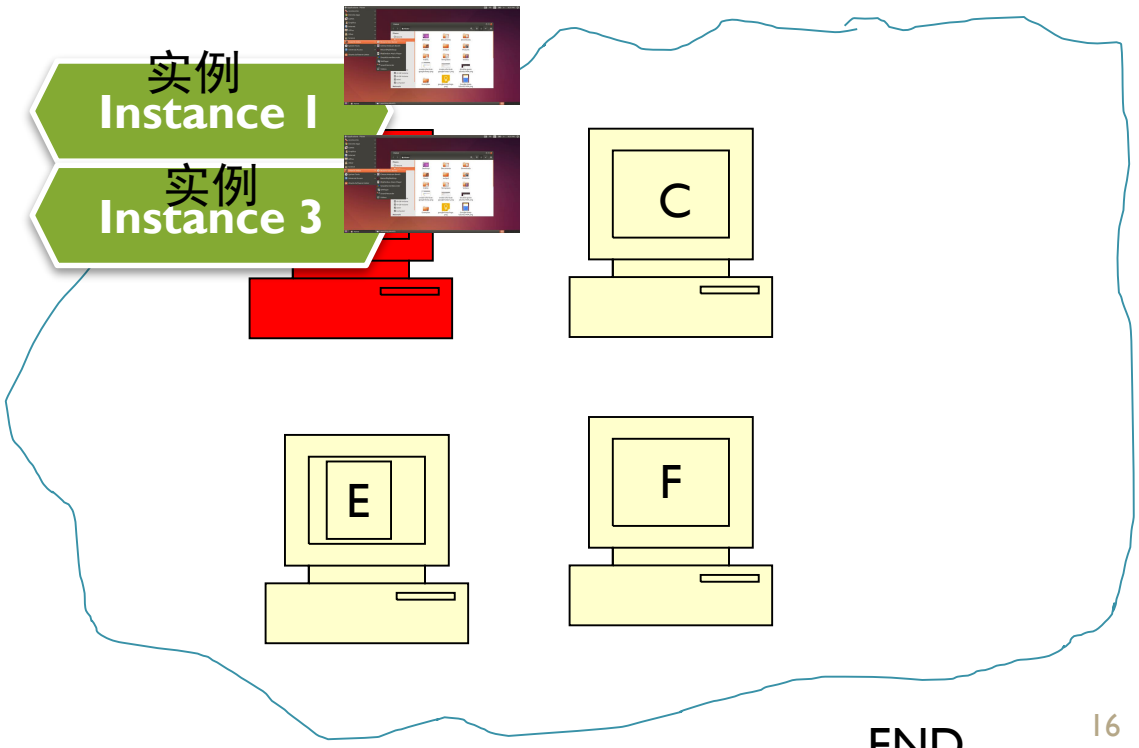


Image AMI

User

END

Important features of the Amazon Cloud

- It can be used for **elastic computing** (弹性计算)
- It can perform **load-balancing** (负载平衡), that is make sure that work is shared between instances
- Various technologies to store data in the cloud (S3, EBS, SimpleDB…).
- Offers a website called CloudWatch to manage your computers in the cloud →

Sensor **CloudWatch**

Overview | Live Data | 2 days | 30 days | 365 days | Historic Data | Log | Settings | Notifications

Last Message:

OK

Last Scan:	Last Up:	Last Down:	Uptime:	Downtime:	Coverage:	Sensor Type:
46 s	**46 s**	**21 d**	**99,9505%**	**0,0495%**	**99%**	**Amazon CloudWatch sensor**

CPU Utilization

8 % 0 % 100 %

DiskRead

300.919 Bytes/sec

DiskRead Ops

11 Ops/sec

DiskWrite

21.026 Bytes/sec

DiskWrite Ops

5 Ops/sec

Network In

8 KByte

Network Out

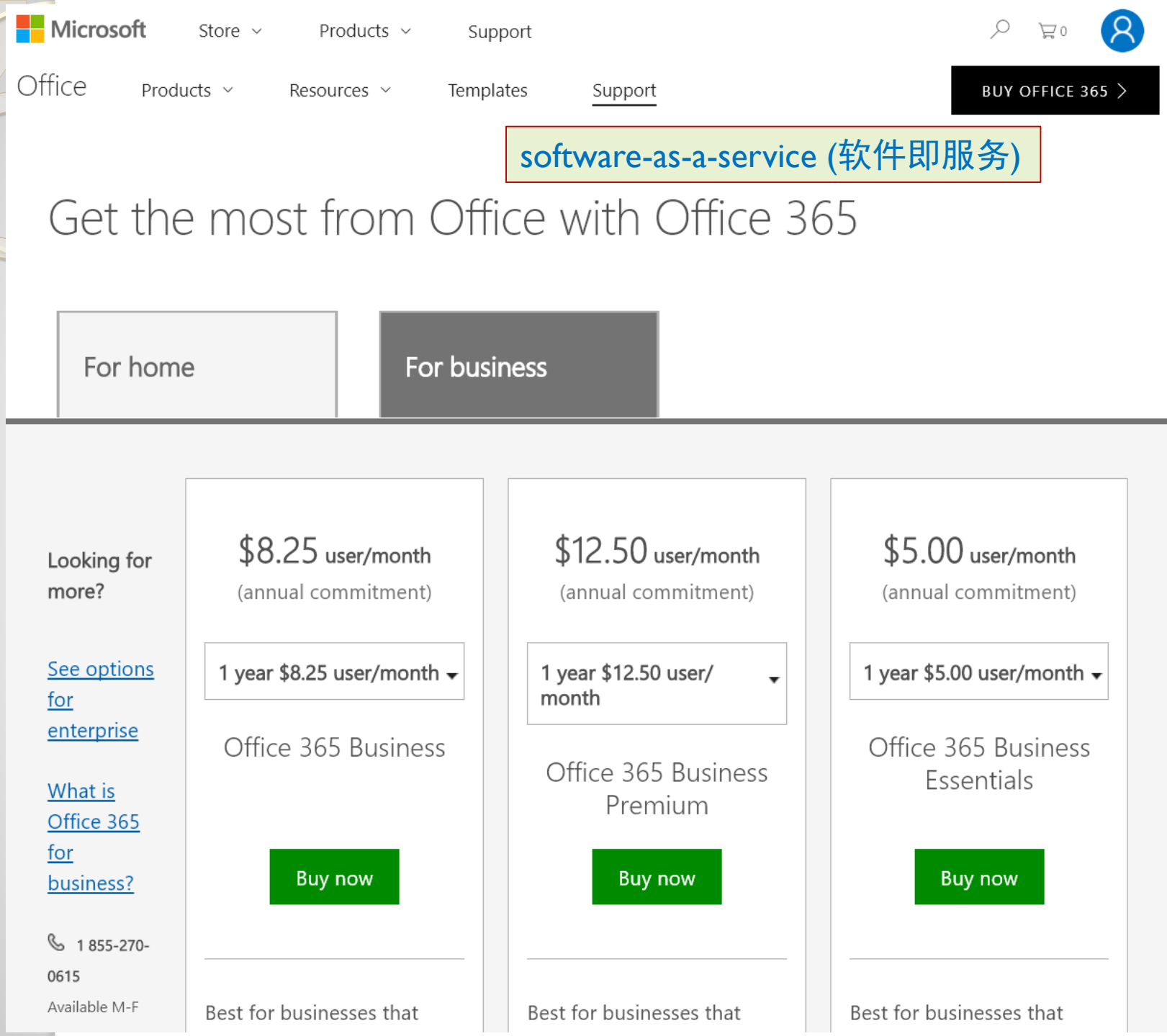
48 KByte

CHANNELS

Channel	ID	Last Value	Minimum	Maximum	Settings
CPU Utilization	0	8 %	<1 %	100 %	
DiskRead	5	300.919 Bytes/sec	7.100 Bytes/sec	3,07445697911762E17 Bytes/sec	
DiskRead Ops	3	11 Ops/sec	2 Ops/sec	1.943 Ops/sec	
DiskWrite	6	21.026 Bytes/sec	9.011 Bytes/sec	10.883.140 Bytes/sec	
DiskWrite Ops	4	5 Ops/sec	2 Ops/sec	1.929 Ops/sec	
Downtime	-4				
Network In	1	8 KByte	0 KByte	7.068 KByte	
Network Out	2	48 KByte	0 KByte	63.149 KByte	

Microsoft

- **Microsoft Online services**:
  software-as-a-service (软件即服务)
  (**e.g.** Hotmail, Office365…)
- **Microsoft Azure**:
  platform-as-a-service (平台即服务)
  - **Windows Azure**: an operating system (操作系统)
  - **SQLAzure**: a database (数据库) for storing data based on SQLServer
  - **AzureAppFabric**: a collection of services for cloud applications


Microsoft
Store
Products
Support
Office
Products
Resources
Templates
Support
BUY OFFICE 365
software-as-a-service (软件即服务)
Get the most from Office with Office 365
For home
For business
Looking for more?
See options for enterprise
What is Office 365 for business?
1 855-270-0615
Available M-F
$8.25 user/month
(annual commitment)
1 year $8.25 user/month
Office 365 Business
Buy now
Best for businesses that
$12.50 user/month
(annual commitment)
1 year $12.50 user/ month
Office 365 Business Premium
Buy now
Best for businesses that
$5.00 user/month
(annual commitment)
1 year $5.00 user/month
Office 365 Business Essentials
Buy now
Best for businesses that

Microsoft Azure

SALES 000-800-100-3928 | MY ACCOUNT | PORTAL | Search

Why Azure | Solutions | Products | Documentation | Pricing | Training | Marketplace | Partners | Blog | Resources | FREE ACCOUNT

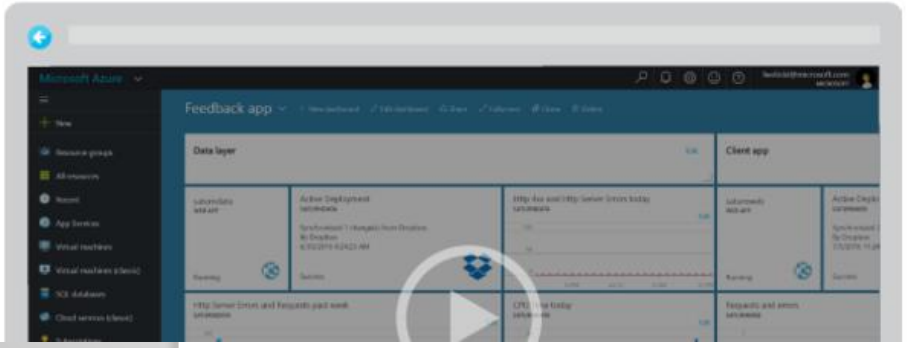
platform-as-a-service (平台即服务)

Global. Trusted. Hybrid.

This is the cloud on your terms

Start free

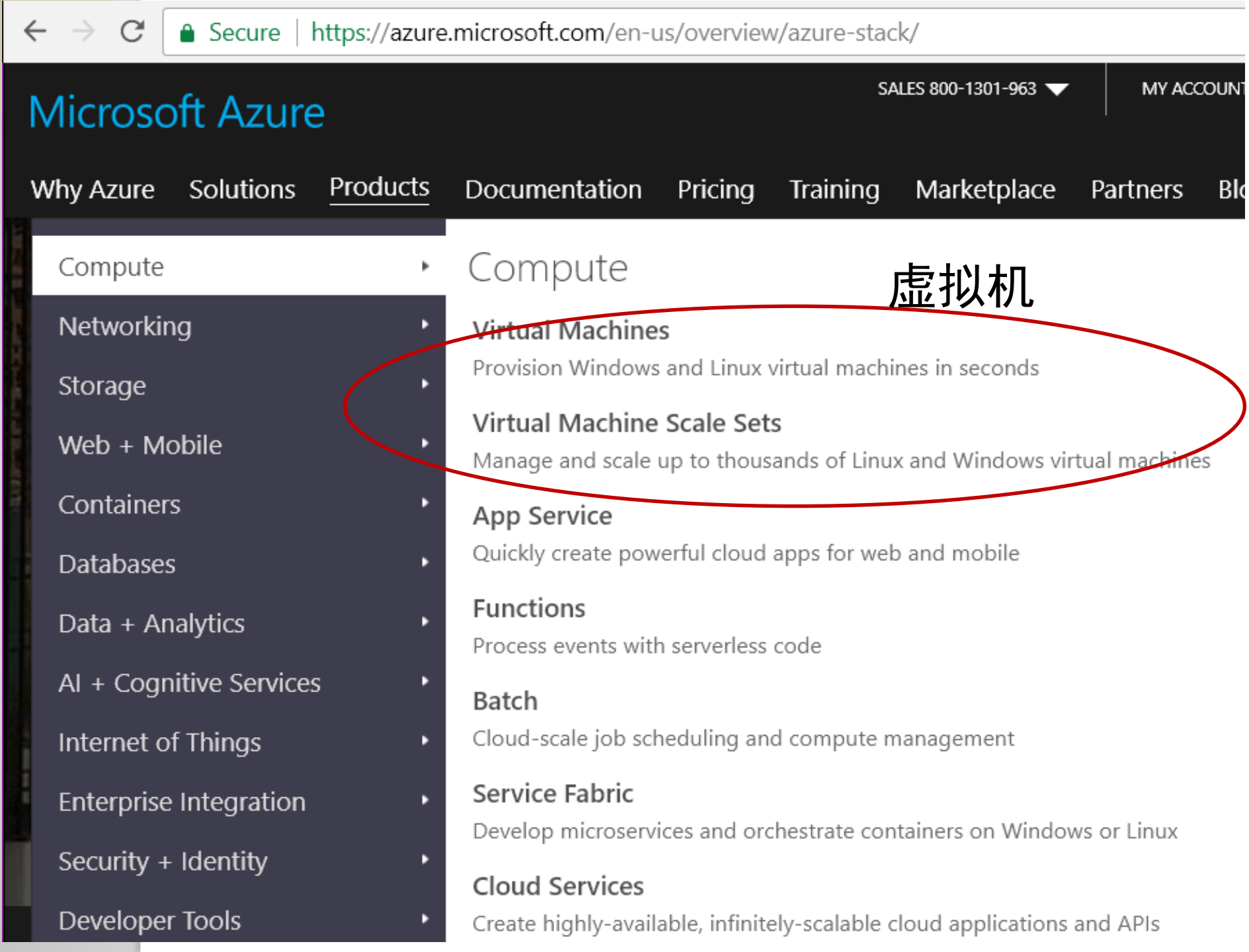
Deploy and scale container-based applications with ease. Explore Web App for Containers

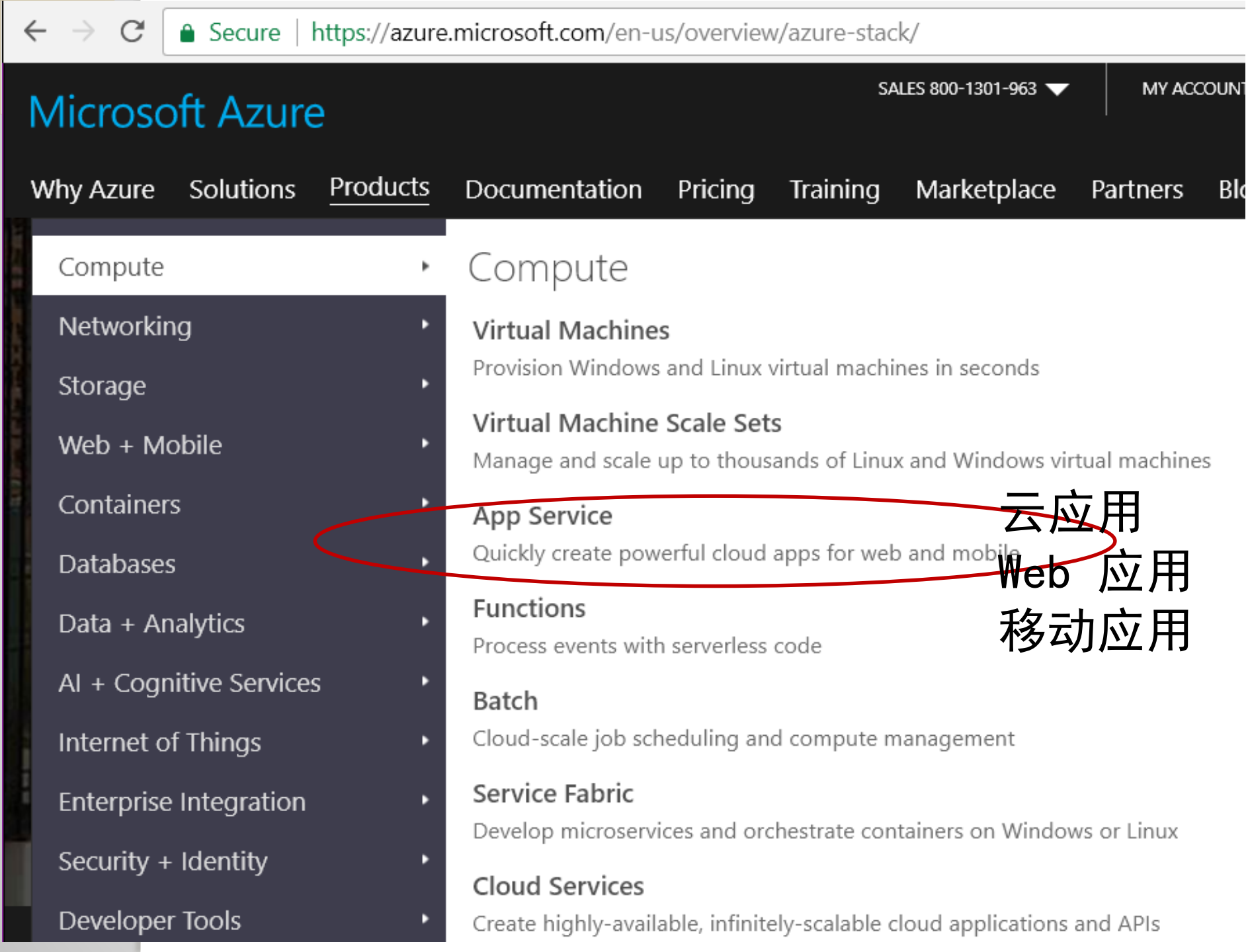


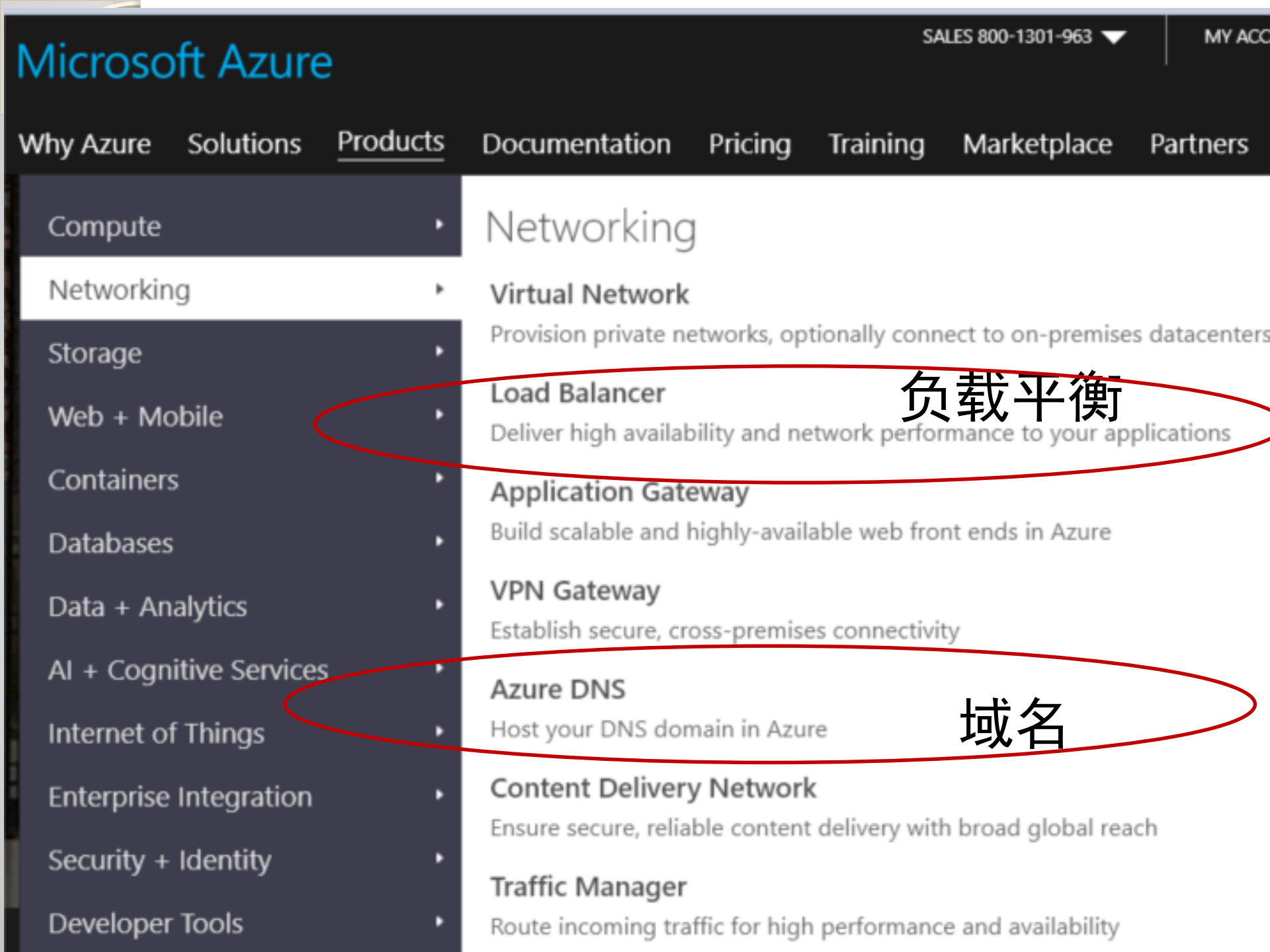
Explore more

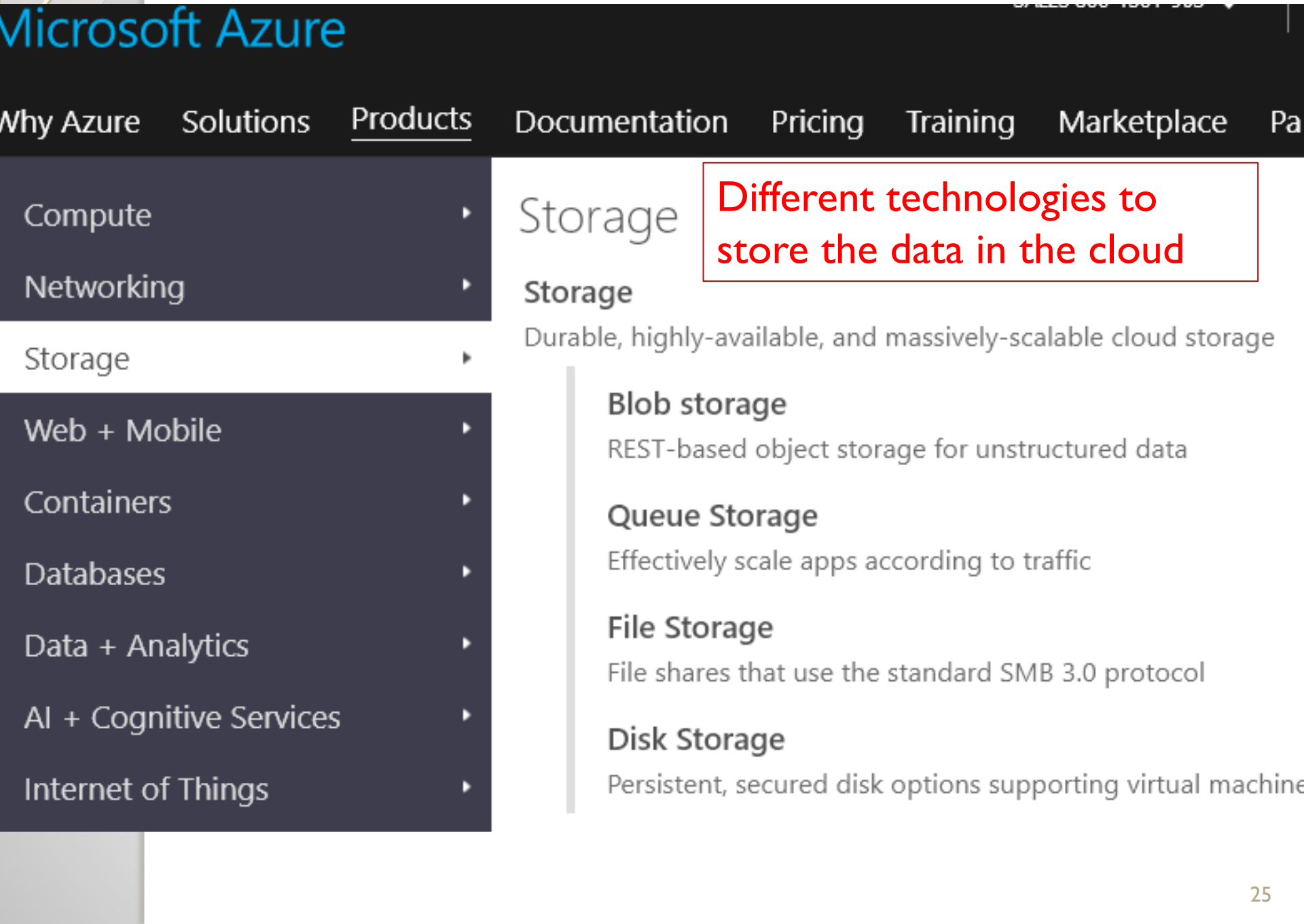
See how you can quickly get started with Microsoft Azure

Explore more


Secure
https://azure.microsoft.com/en-us/overview/azure-stack/
Microsoft Azure
SALES 800-1301-963
MY ACCOUNT
Why Azure
Solutions
Products
Documentation
Pricing
Training
Marketplace
Partners
Compute
Networking
Storage
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Enterprise Integration
Security + Identity
Developer Tools
Compute
虚拟机
Virtual Machines
Provision Windows and Linux virtual machines in seconds
Virtual Machine Scale Sets
Manage and scale up to thousands of Linux and Windows virtual machines
App Service
Quickly create powerful cloud apps for web and mobile
Functions
Process events with serverless code
Batch
Cloud-scale job scheduling and compute management
Service Fabric
Develop microservices and orchestrate containers on Windows or Linux
Cloud Services
Create highly-available, infinitely-scalable cloud applications and APIs


Secure
https://azure.microsoft.com/en-us/overview/azure-stack/
Microsoft Azure
SALES 800-1301-963
MY ACCOUNT
Why Azure
Solutions
Products
Documentation
Pricing
Training
Marketplace
Partners
Compute
Networking
Storage
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Enterprise Integration
Security + Identity
Developer Tools
Compute
Virtual Machines
Provision Windows and Linux virtual machines in seconds
Virtual Machine Scale Sets
Manage and scale up to thousands of Linux and Windows virtual machines
App Service
Quickly create powerful cloud apps for web and mobile
Functions
Process events with serverless code
Batch
Cloud-scale job scheduling and compute management
Service Fabric
Develop microservices and orchestrate containers on Windows or Linux
Cloud Services
Create highly-available, infinitely-scalable cloud applications and APIs
云应用
Web 应用
移动应用


Microsoft Azure
SALES 800-1301-963
MY ACC
Why Azure
Solutions
Products
Documentation
Pricing
Training
Marketplace
Partners
Compute
Networking
Storage
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Enterprise Integration
Security + Identity
Developer Tools
Networking
Virtual Network
Provision private networks, optionally connect to on-premises datacenters
Load Balancer
Deliver high availability and network performance to your applications
负载平衡
Application Gateway
Build scalable and highly-available web front ends in Azure
VPN Gateway
Establish secure, cross-premises connectivity
Azure DNS
Host your DNS domain in Azure
域名
Content Delivery Network
Ensure secure, reliable content delivery with broad global reach
Traffic Manager
Route incoming traffic for high performance and availability


Microsoft Azure
Why Azure
Solutions
Products
Documentation
Pricing
Training
Marketplace
Compute
Networking
Storage
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Storage
Different technologies to store the data in the cloud
Storage
Durable, highly-available, and massively-scalable cloud storage
Blob storage
REST-based object storage for unstructured data
Queue Storage
Effectively scale apps according to traffic
File Storage
File shares that use the standard SMB 3.0 protocol
Disk Storage
Persistent, secured disk options supporting virtual machine

Shanghai Blue Cloud Te...nology Co., Ltd. [CN] | https://en.21vbluecloud.com/

世纪互联
www.21vianet.com
上海蓝云网络科技有限公司

About Us Servic

- 30+ Cities
- 80+ Data Centers
- 600+ Pops
- 30,000+ Cabinets

Microsoft Azure Operated by 21Vianet

- The first international standard cloud computing services used in domestic commercial area.
- More than 80 thousand local corporate customers.
- Provide a financial guarantee monthly 99.9% SLA service level agreement guarantee.
- Provide 7x24 hours of rapid response services for customers with data center support, cloud platform operations, customer support, compliance consulting services.

Microsoft Azure

世纪互联®
www.21vianet.com
上海蓝云网络科技有限公司

Power BI 由世纪互联运营
Microsoft Azure 由世纪互联运营

首 页 | 蓝云的服务 | 资讯中心 | 运营的云产品 | 云应用方案 | 蓝云社区中心 | 关于蓝云

当前位置：首页 > 运营的云产品 > Microsoft Azure > 案例研究

运营的云产品

- Microsoft Azure
  - > 解决方案
  - > 功能
  - > 用户类型
  - > 案例研究
- Office 365
- Power BI

联系蓝云

热线电话：

400 089 0365

咨询电话：

86-10-57835502

案例研究

人们正在使用 Azure 创造精彩的生活

金蝶K/3CloudERP — 查看案例

qTestin自助应用测试平... — 查看案例

北京渲染平台 — 查看案例

经纬 — 查看案例

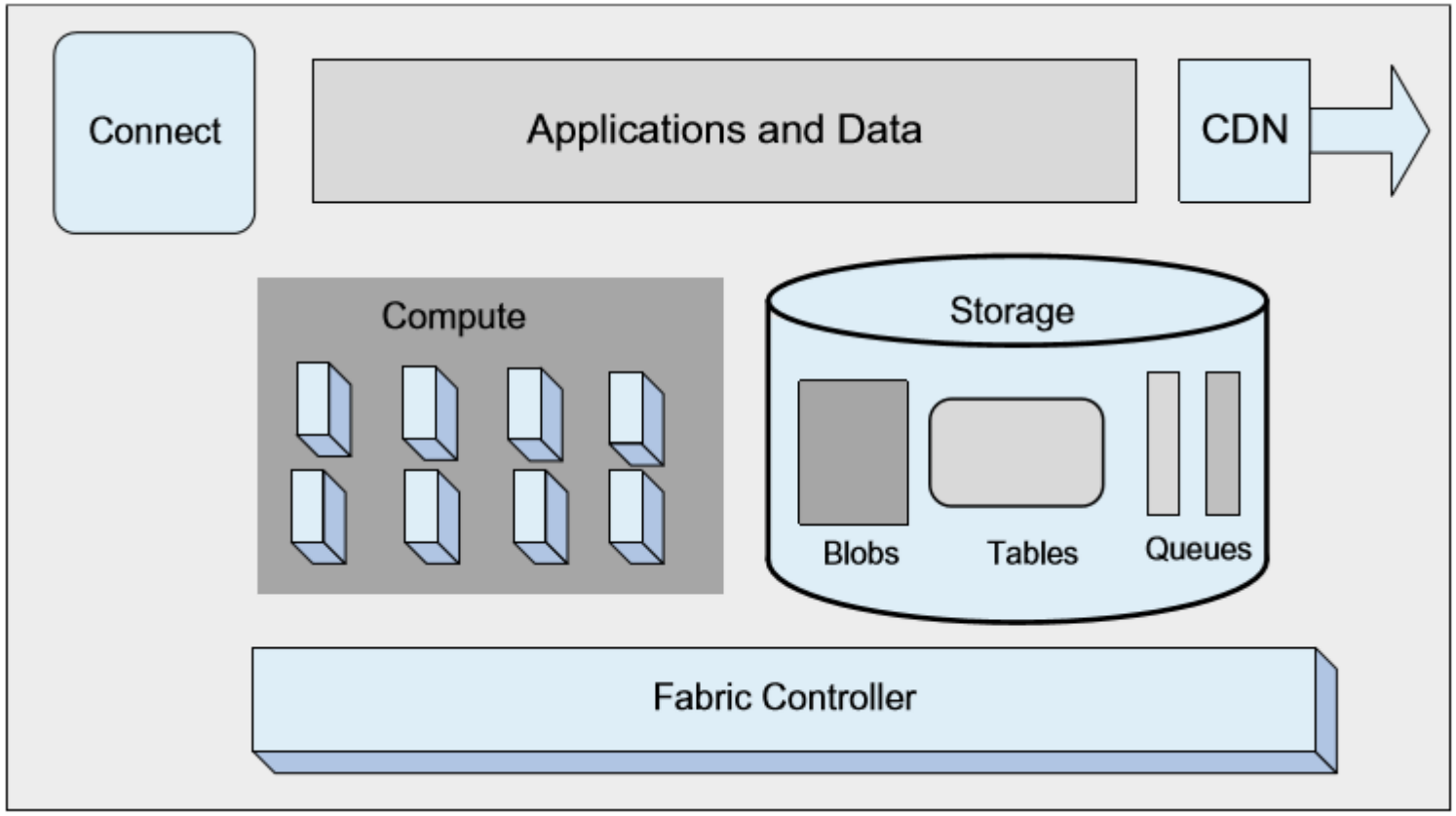
观致QorosQloud车联网 — 查看案例

PPTV亚洲电视网 — 查看案例

蓝汛Webluker — 查看案例

虫洞语音助手 — 查看案例

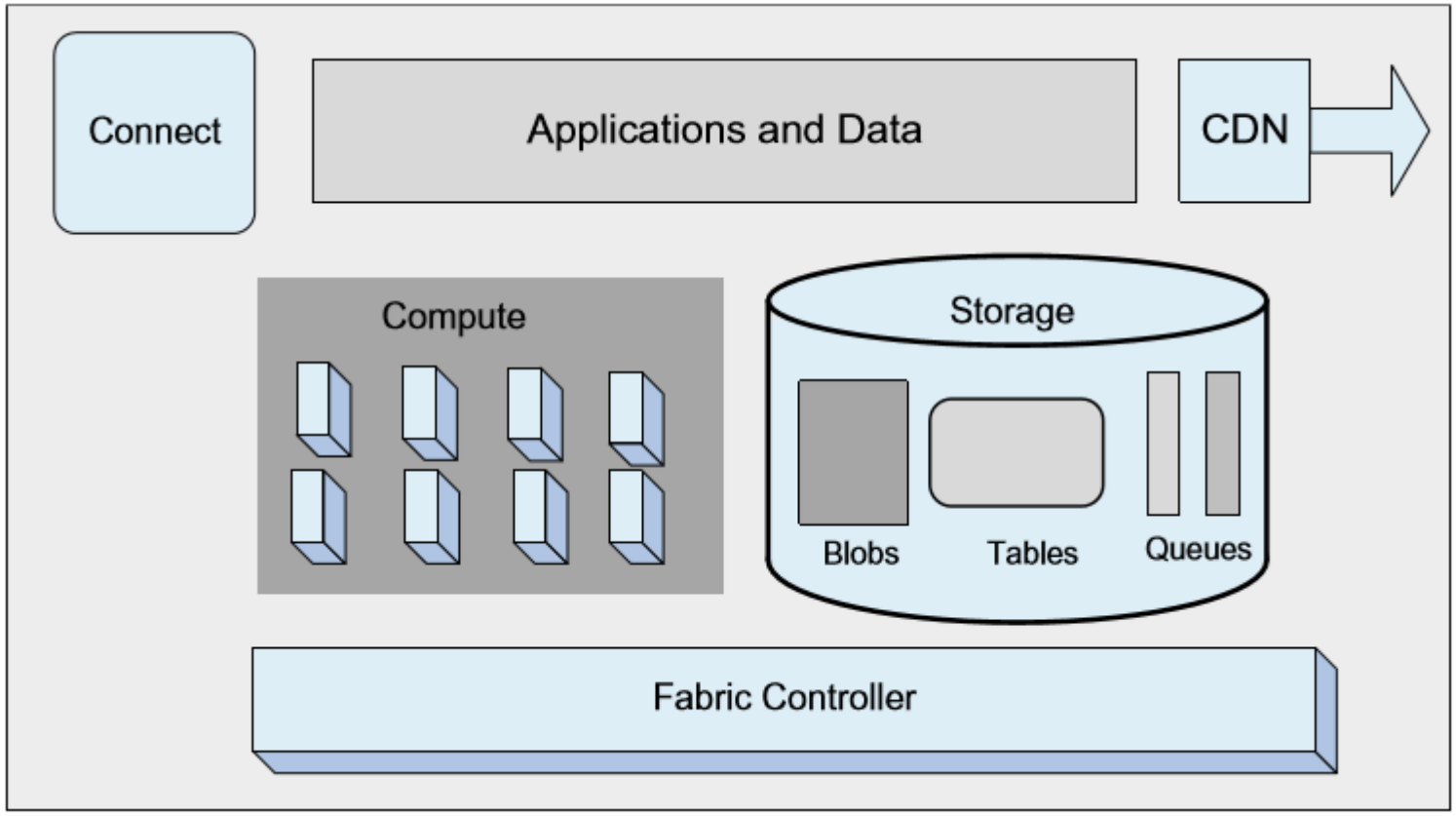
Microsoft Azure



Three main components:

- **Computers**: provides a computation environment
- **Storage**: scalable storage (可扩展的存储)
- **Fabric Controller**: deploys, manages, and monitors applications; it interconnects servers, high-speed connections, and switches (网络交换机)

Microsoft Azure



- **Storage** uses blobs, tables, and queues to store data,
- **Fabric controller**: provides scaling, load balancing, memory management (内存管理), and reliability (可靠性)
- **CDN**: maintains cache (缓存) copies of data, for faster access.

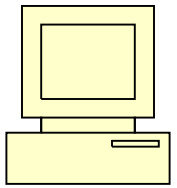
blob = up to 1 terabyte (TB) of data

4 – CLOUD APPLICATIONS (PART I)

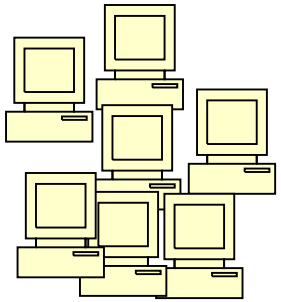
pl18

Introduction

- We will discuss **how applications (应用) are developed** for the cloud.
- **Goal:** understand how programmers develop cloud applications.
  - **Standard applications (应用):** run on a single computer.



  - **Cloud applications (云应用):**
    - run on one or more computers,
    - possibility of sharing the workload (工作负载),
    - a type of distributed application 分布式应用.



Introduction

Developing applications for the cloud is more challenging than developing applications for **a desktop computer**.

Why? →

How to develop a desktop application (桌面应用)?

1. A programmer discusses with **users** to understand their **needs** (用户需求), and know **what type** of application should be developed.
2. The programmer **write the application** using a **programming language** (编程语言) (**e.g**. C++, Java…).
3. The programmer will **test** the application(应用测试)

How to develop a desktop application (桌面应用)?

1. The programmer will **deliver** the application to the user (应用交付).
2. If necessary, the programmer will find and fix problems in the application (**debug** - 调试) or add **new features**.

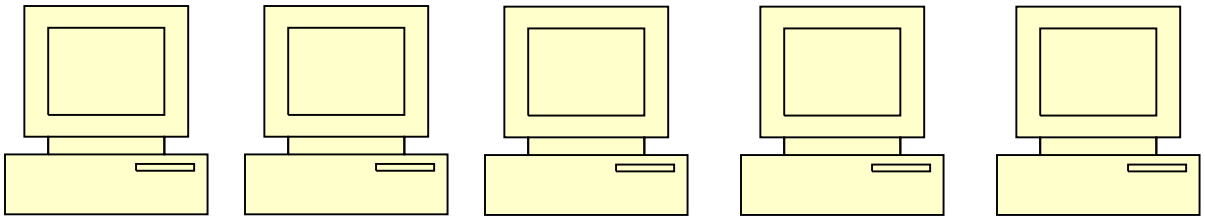
How to develop a cloud application (云应用)?

Key differences

- The cloud is a **parallel** (并联系统) and **distributed system** (分布式系统).
- A **cloud application** must be able to run on multiple computers at the same time to share the workload.

Developing a cloud application

- The **development of a cloud application** follows a **similar process to the development of a desktop application**.
- However, developing a cloud application is done using specific programming languages or **technologies, to be able to use the cloud** and its **benefits** (load balancing, cloud storage, etc.)
- **For example,** an application can be designed in the **Java** programming language using the **Hadoop** *framework* for developing cloud applications, and using cloud storage.

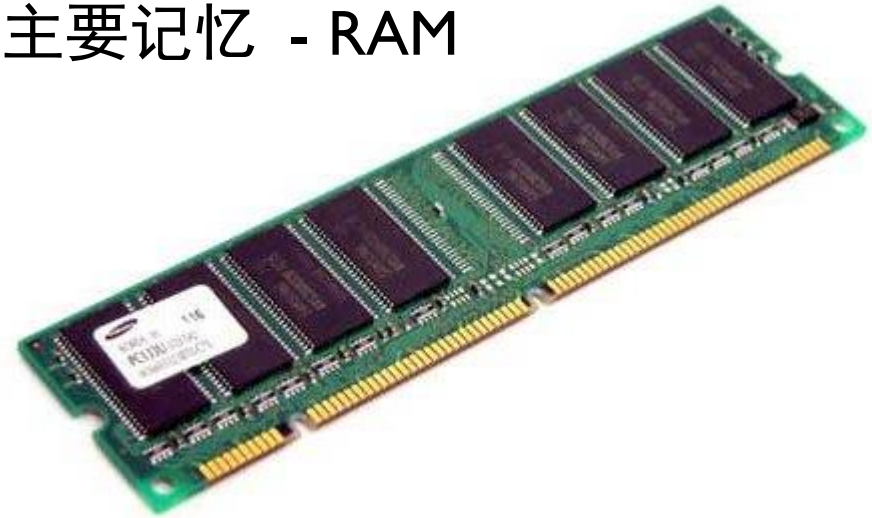


Challenges to develop a cloud application

Challenge 1: speed imbalance (速度不平衡)

- A desktop computer can quickly access data in its memory.
- Accessing data in **main memory (主要记忆 - RAM)** is faster than on a **hard drive (硬盘驱动器 - HD)**, which is faster than on a **computer network (计算机网络)**.
- Thus, accessing some data may be **slower** than accessing some other data. This is called **speed imbalance**.

主要记忆 - RAM



hard drive (硬盘驱动器)

Challenges to develop a cloud application

Why?

- **In the cloud**, speed imbalance is greater.
- **Accessing data can sometimes be very slow** because of various reasons (slow **network speed**, **computers** are busy, etc.)

Challenges to develop a cloud application

Challenge 2: difficult to predict the performance

- The performance of a cloud application can be measured in terms of:
  - computer performance,
  - network performance,
  - other aspects.
- The performance of a cloud application may be unpredictable or fluctuate with time.
  - **WHY? →**

Challenges to develop a cloud application

Why?

- the cloud is often used by many users at the same time.
- users may run applications on the same computer(s),
- users often use the same network as other users.

Besides, a cloud application will **not perform well if** a **huge amount of communication** must be performed between computers.

Challenges to develop a cloud application

Challenge 3 : data storage

- Developers of cloud applications must also think carefully about how data will be stored in the cloud.
- Three important questions:
  - **how** the data will be stored
    (in what kind of database 数据库? or files?…)
  - **where** the data will be stored
    (how many computers?, where? …) ?
  - will the data be **replicated**
    (multiple copies of the data? – 数据复制)?

Challenges to develop a cloud application

Challenge 4: complex errors may occur

- **Complex problems** may occur when several computers are working together:
  (**e.g.** concurrent accesses, livelocks, unfairness…)

Challenges to develop a cloud application

- It is **difficult to ensure** that a **parallel application** (并行应用) **will work correctly** (that it has no *bugs* 缺陷)
- When problem occurs in a parallel system, it is **difficult to find the reasons** because
  - many events occur at the same time.
  - the interaction between computers may be complex.
- **A solution**: record **logs** (日志)

Other challenges

- Cloud **technologies are rapidly changing**.
- Hence, cloud applications developers may need to learn new technologies.

  Hadoop, Spark, etc.
- Some **technologies may be proprietary** to some companies
  (**e.g.** Microsoft, Amazon).

Why developing cloud applications?

We should develop a cloud application instead of a desktop application:

- for **large applications where we want to use as many servers as required** to provide a reliable service which respect time constraints.
- when we do not want to buy many computers and infrastructure,
- when the application needs to process **large amount of data (big data - 大数据)**

Why developing cloud applications?

- when **the workload can be partitioned** in **segments of arbitrary size** that can be processed in parallel by servers in the cloud.
- …

Note: It is **desirable** that the workload can be divided in an **arbitrary number of segments**, so that as many computers as needed may be used

Example of cloud applications

- Cloud applications are often **compute-intensive** (计算密集型) and **data-intensive** (数据密集型)
- **Examples:**
  - a search engine like **Baidu** and **Bing** must
    - **collect data** about all websites on the internet and store this data in a large database,
    - quickly **answer requests** made by millions of users at the same time,

→

Example of cloud applications

- websites for storing and sharing videos like **Youku** and **Tudou** have to:
  - perform many tasks related to **video processing** (视频处理)(compressing videos, analyzing the content of videos, etc.),
  - quickly **deliver videos** to users from different geographical areas.
- some companies will perform "data mining" (数据挖掘), that is extract interesting knowledge hidden in large databases. For example: analyze the behavior of customers in a retail store.

Example of cloud applications

Other examples:

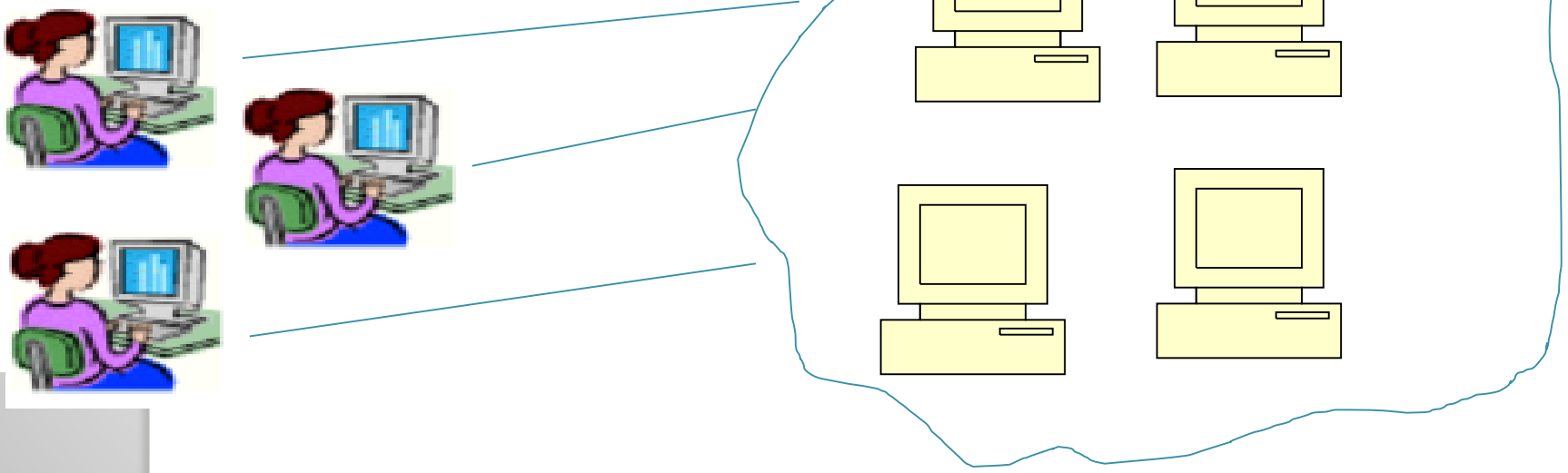
- generate daily, weekly, monthly, annual activity reports related to manufacturing, economy, etc.,
- inventory management (库存管理) for large companies,
- billing, payroll
- **Websites**
- **web applications** (Yahoo Mail, 163 e-mail, etc.)
- **Provide services to mobile applications** (Baidu Maps…)
- …

Architectural styles for cloud applications

Cloud applications adopt a **client-server architecture** (客户机–服务器体系结构).

client (客户机) **computers** can use the services provided by the cloud.

In the **CLOUD:** the **server** (服务器) provides some service to the clients



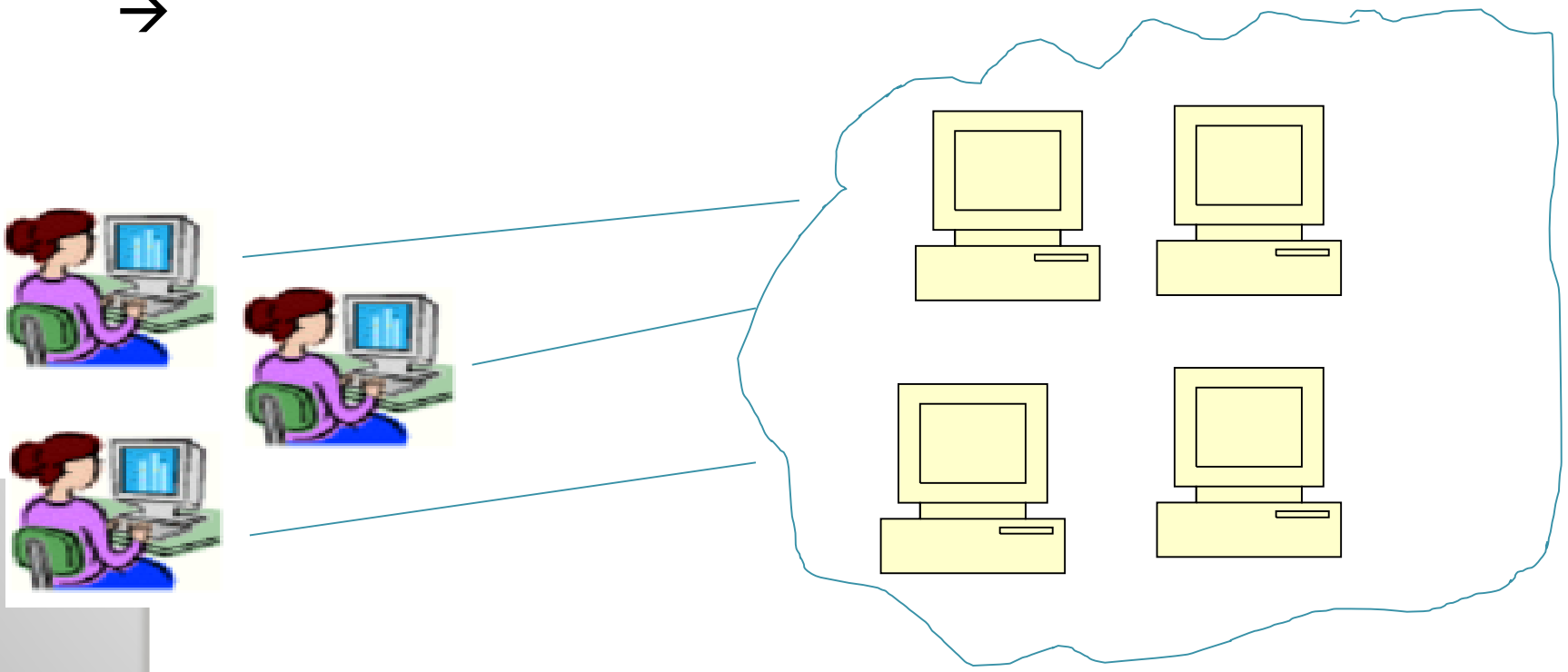
Stateless servers

Stateless server (无状态服务器)

- A **server** (服务器) does not "**remember**" the communication with clients. It processes each message as a new and independent message .

Why?

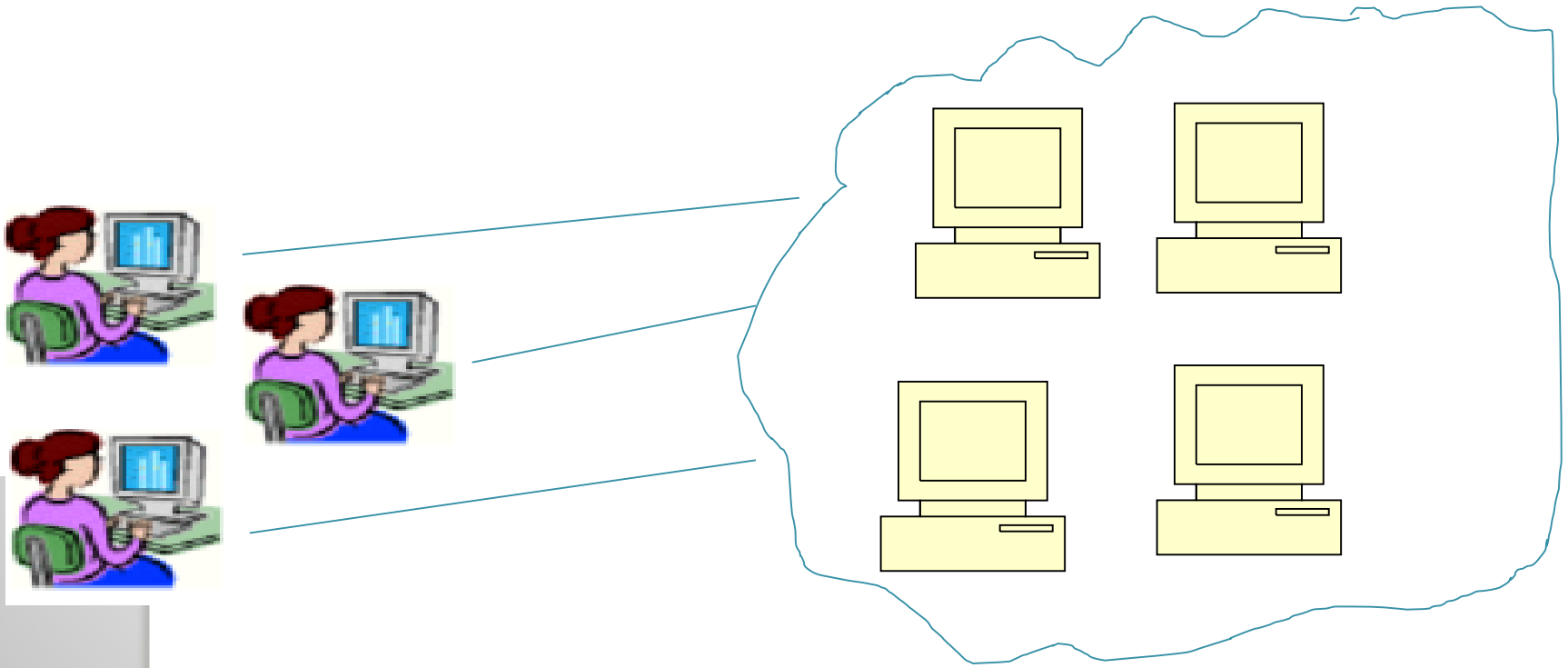
→



Stateless servers

Why?

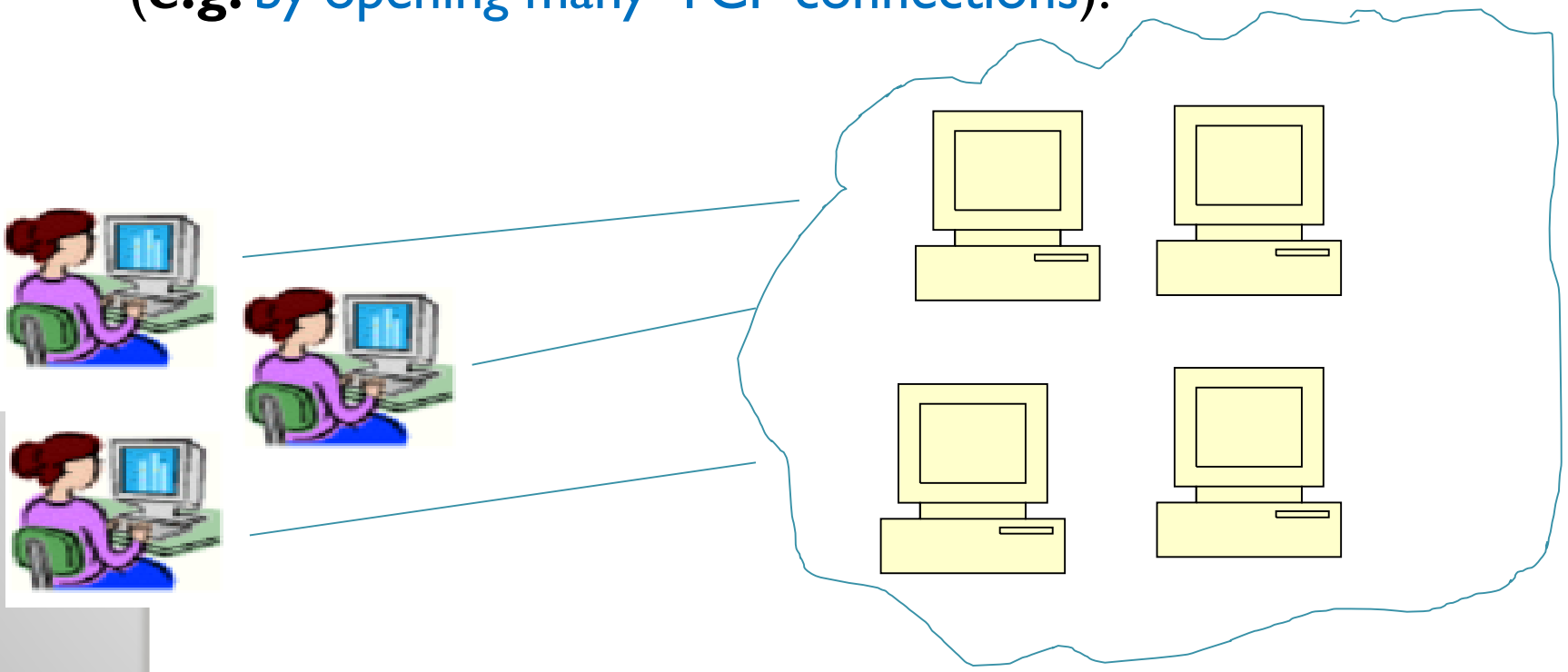
- It is more scalable (可扩展的).
- We can add more servers to answer request by clients. Any servers can answer any requests by any clients.
- Easier to recover after a server failure



Stateless servers

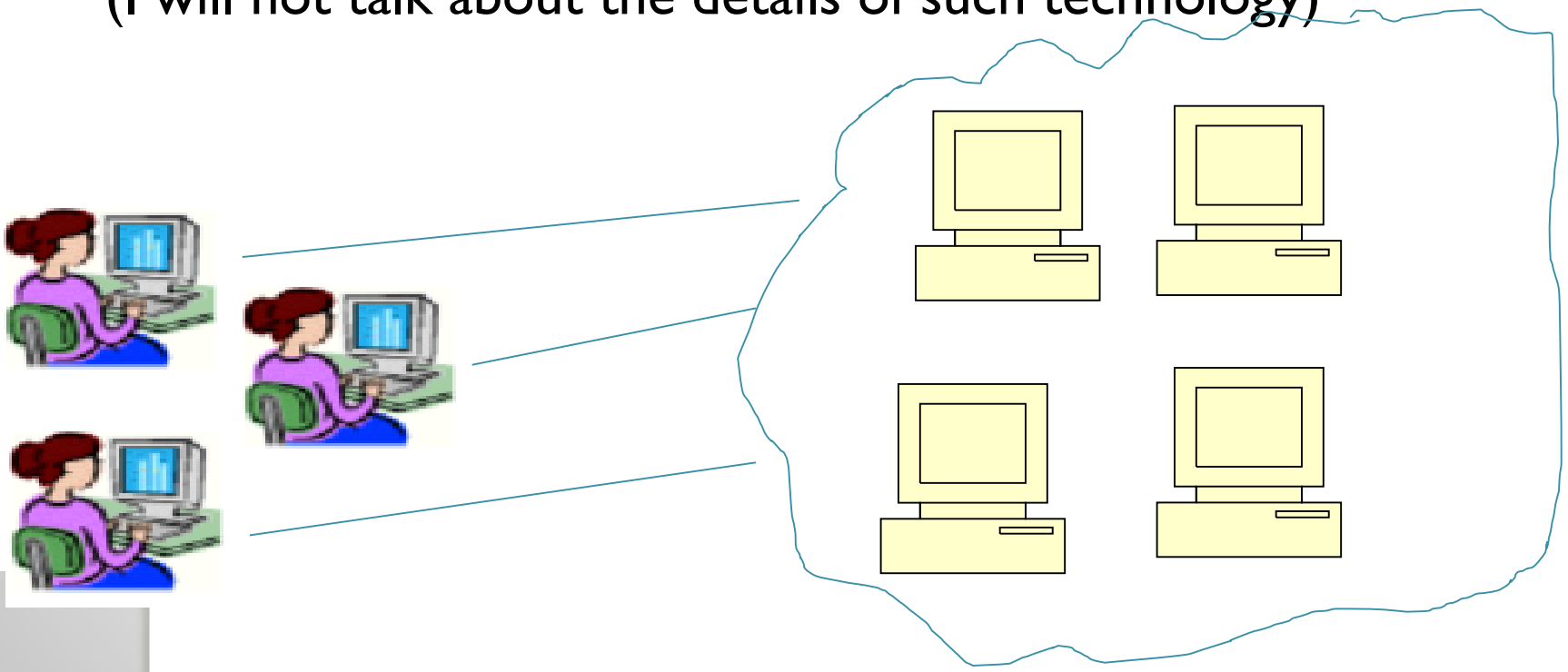
Why?

- A **client does not need to consider the state of the server.** If a server does not answer, the client can send the request again and it can be processed by another server.
- To **avoid some security problems** (if a server would reserve space for each client, a hacker could try to overload a server (**e.g.** by opening many TCP connections).



How to communicate?

- It is desirable that cloud applications can communicate with other computers using different technologies (**e.g**. SOAP messages can be sent using TCP, UDP, SMTP, JMS.. Protocols for communications)
- REST is a popular technology for communicating in the cloud
  (I will not talk about the details of such technology)



Coordination of multiple activities

- Cloud applications often require the completion of multiple interdependent (相互依存) **tasks**.
- The description of a complex activity involving such an **ensemble of tasks** is called a **workflow** (工作流).

- **Coordination** (协调) is thus necessary.

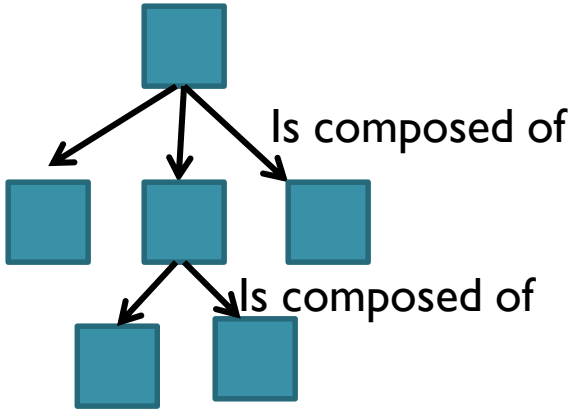
Example - cooking

How can a task be represented?

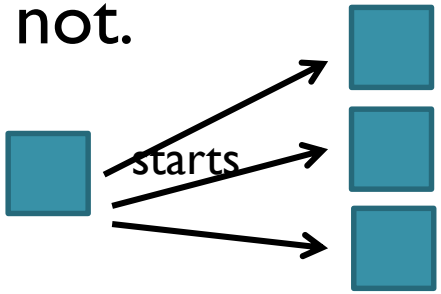
- **Name**: the name of the task
- **Description**: a description of the task in natural language (**e.g.** Chinese, English)
- **Preconditions** (先决条件): some conditions that must be true before the task can be performed
- **Postconditions** (后置条件): some conditions that become true after the task has been performed.
- **Attributes** (属性): resources necessary to perform the task
- **Exceptions** (异常): how abnormal events should be handled?

Some types of tasks

- **Composite task**: a task that is composed of several tasks that must be accomplished according to some order.

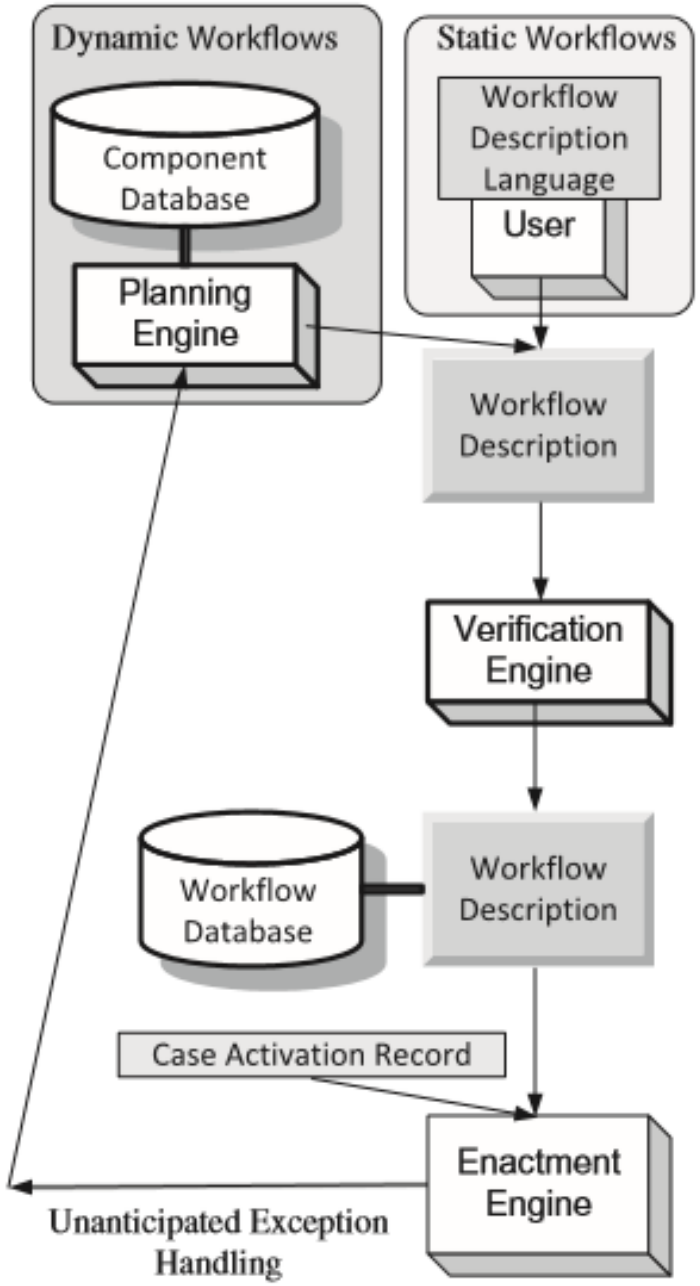


- **Routing task**: a task that has for purpose of triggering the start of multiple other tasks simultaneously or not.



Organizing tasks as workflows

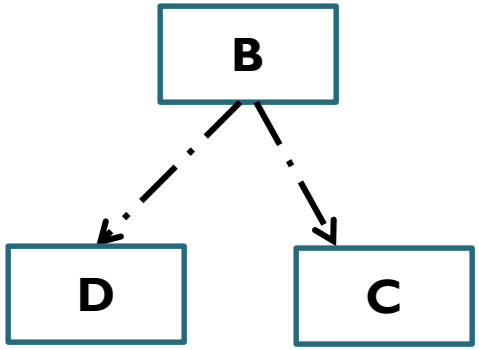
- All tasks can be organized in a **workflow**.
- Some **languages** like **WFDL** (Workflow Definition Language) allows to define **workflows**.



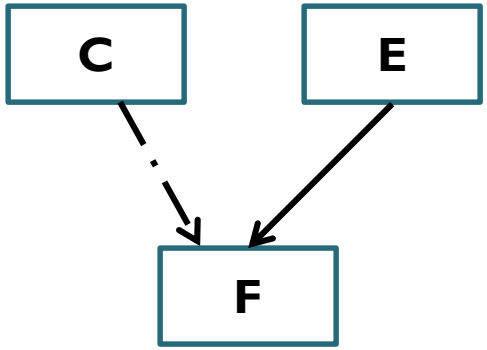
A system based on workflows may work as follows:

1) The user specify a **workflow**.
2) Or a **planning engine** can be used to generate a workflow
3) A **verification engine** may check that the workflow is syntactically correct.
4) Then the **enactment engine** will execute (do) the workflow
5) If errors occur, the plan may be changed.

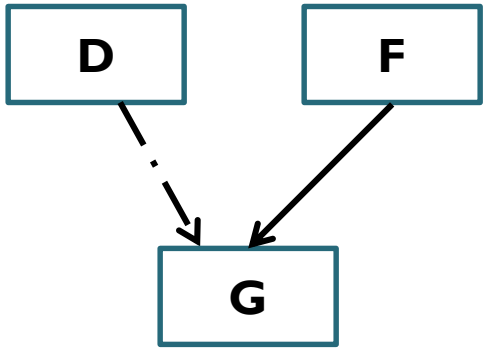
Another workflow representation



After B, we can choose either D or C

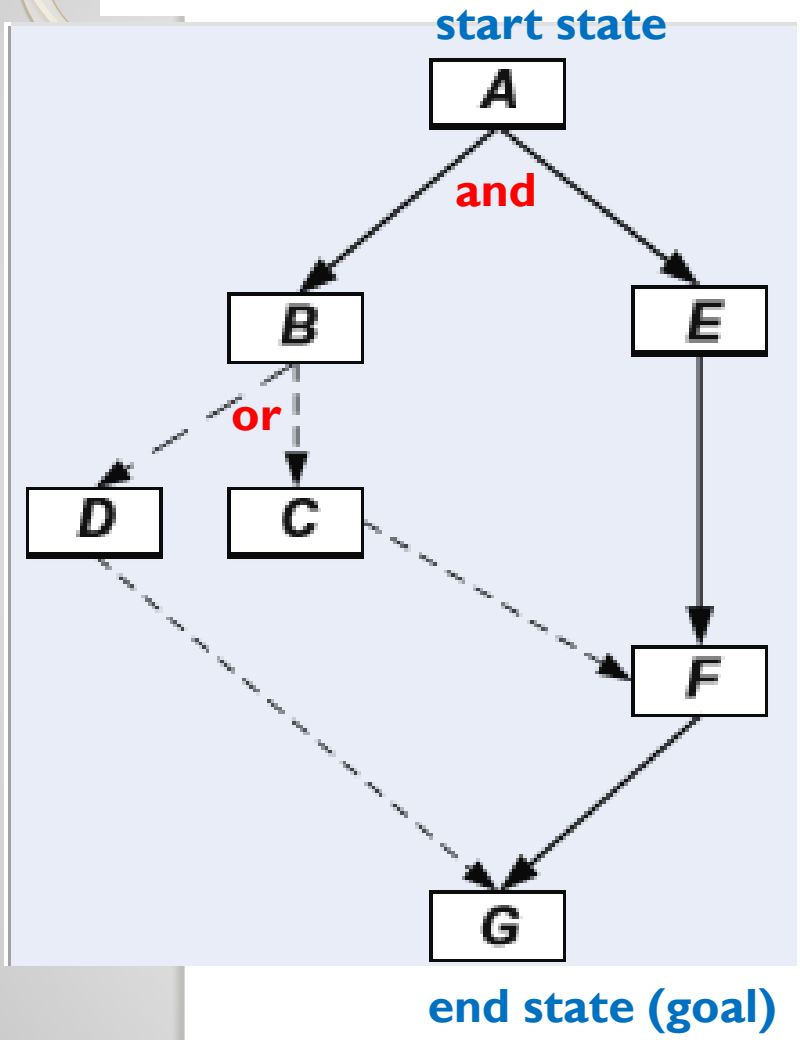


C and E must be performed before F



D and F must be performed before G

Another workflow representation



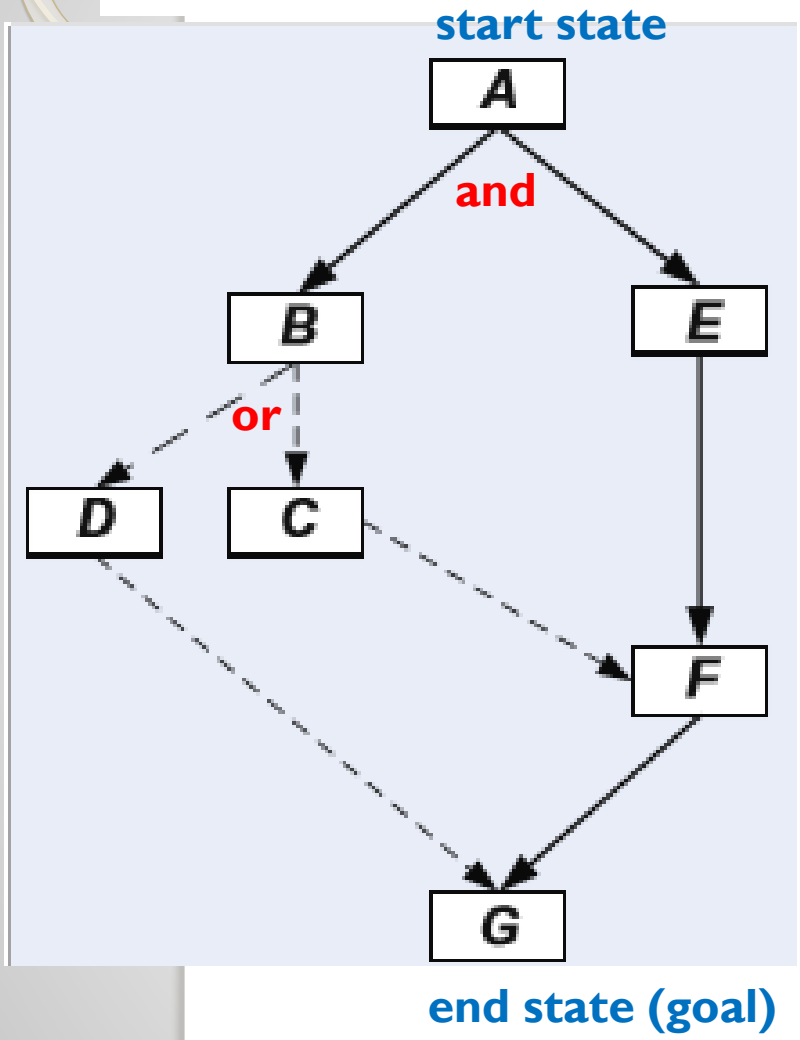
A workflow can also be represented as a **state-transition diagram** (状态转换图):

- each **node** is a **state,**
- each **arrow** between two nodes is a **transition,**
- there is a **start state** and an **end state**

← **Is this a good workflow?**

NOTE: G is a goal (it is not a task!)

Another workflow representation

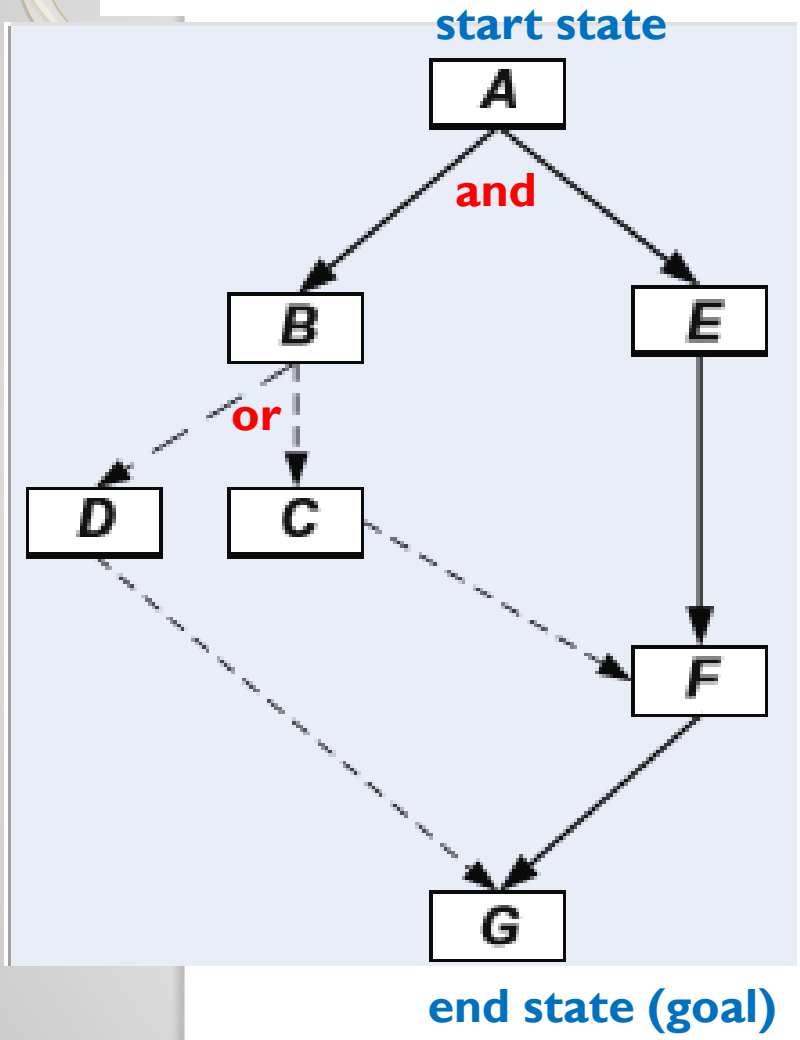


No.

This problem can occur:

- If task **D** is chosen after the completion of **B**, then **F** will never be performed.
- The reason is that **C** and **E** must be performed before **F**.
- Thus the workflow will never be completed.

Another workflow representation

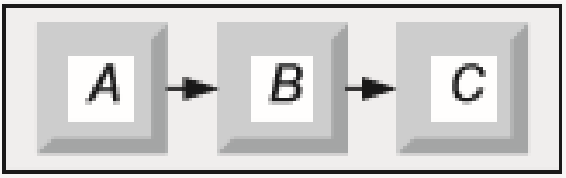


But if task **C** is chosen after the completion of **B**, then there is no problem.

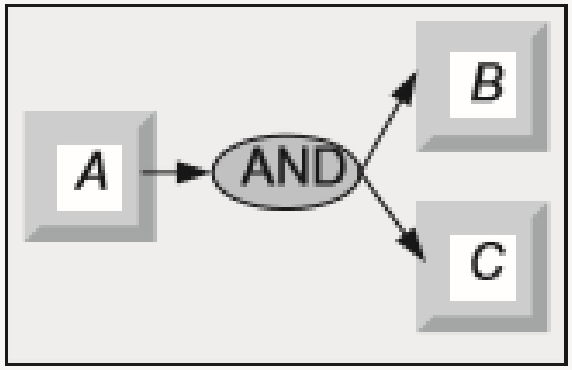
Note that other problems may perhaps also occur because of shared resources (not shown in this example).

Patterns to define a workflow

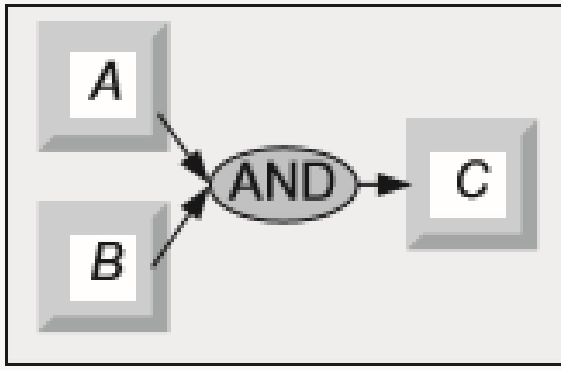
Several types of ***patterns*** can be used to define a workflow. For example:



A, B and C must be performed sequentially

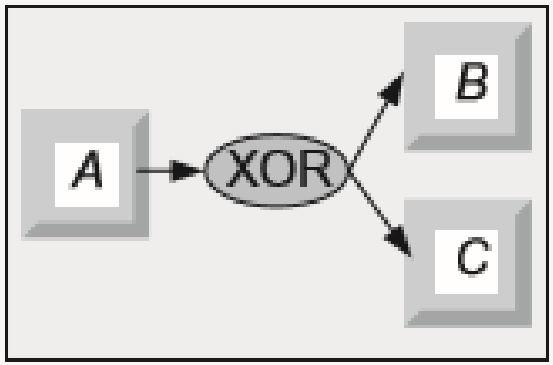


A must be performed and then both B and C

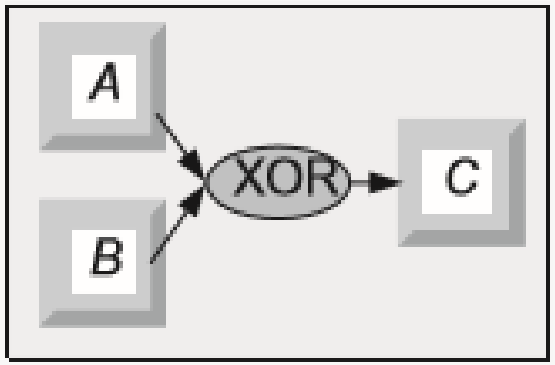


A and B must be performed before C

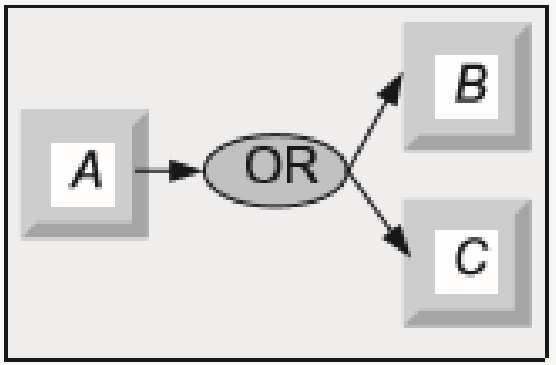
Patterns to define a workflow



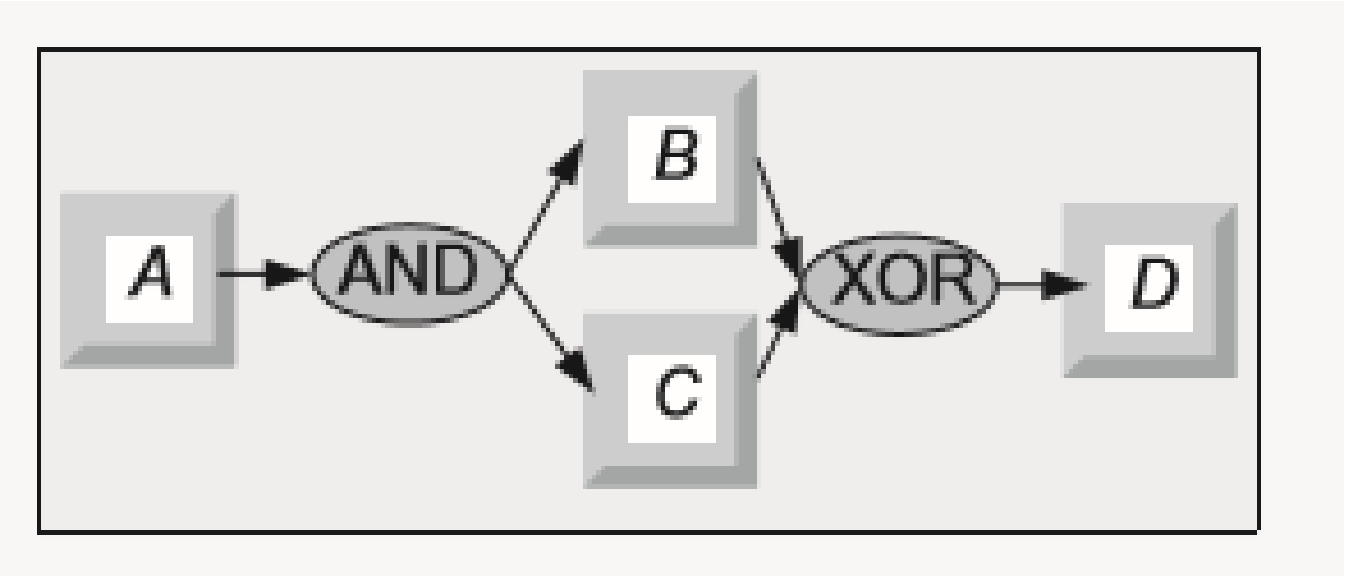
A must be performed and then either B or C



The task C is performed when either A or B are completed.

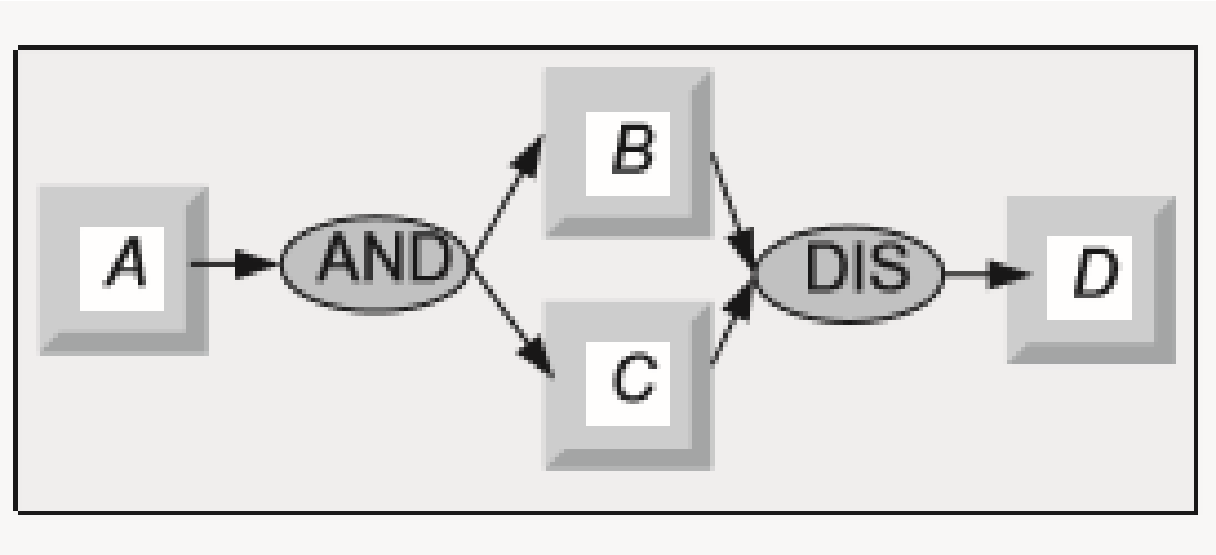


A must be performed before B or C, or both B and C

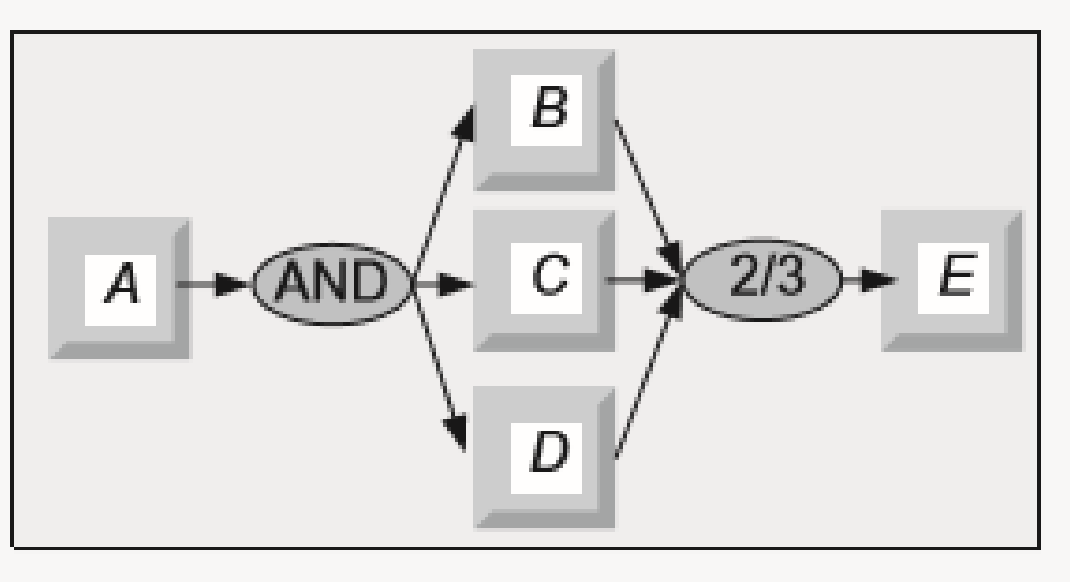


A will be executed and then **B** and **C.**

D will be activated after the completion of **B**, and it will be also activated after the completion of **C.**



A will be executed and then **B** and **C.**
When at least **B** or **C** is completed, **D** will be activated.
Then, when **D** is completed, it is necessary to wait that **B** and **C** are both completed if they were not already.



A will be executed and then **B**, **C**, and **D.**

When two of the three tasks **B**, **C** or **D** are completed, the task **E** will be activated

Workflow management (工作流管理)

- An **enactment engine (**执行引擎) can be used to execute a workflow.
- Given **a set of computer instances** in the cloud, the enactment engine will assign tasks to the computer instances.
- Resources will be assigned to the computer instances.
- **Workflow can be static (**静态)**or** they can be **dynamic**（动态） (they can be modified when needed)

Two workflow management models

Strong coordination:

- There is a computer instance that acts as a **supervisor (the boss!)** of all the other instances.
- It ensures that the workflow is respected.
- **Hierarchical coordination**: several levels of supervisors (as in a company)
- **Benefits**:
  - the supervisor can dynamically modify the workflow by stopping some tasks when needed
    (**e.g** when it receives a request).
  - This model does not require much communication

Two workflow management models

Weak coordination:

- There is **no supervisor.**
- Computer instances communicates with each other to share information about the tasks that have been completed.
- Different ways to do that.
- Each computer may have a copy of the workflow.
- It is more difficult to adapt the workflow dynamically.

The Zoo Keeper model

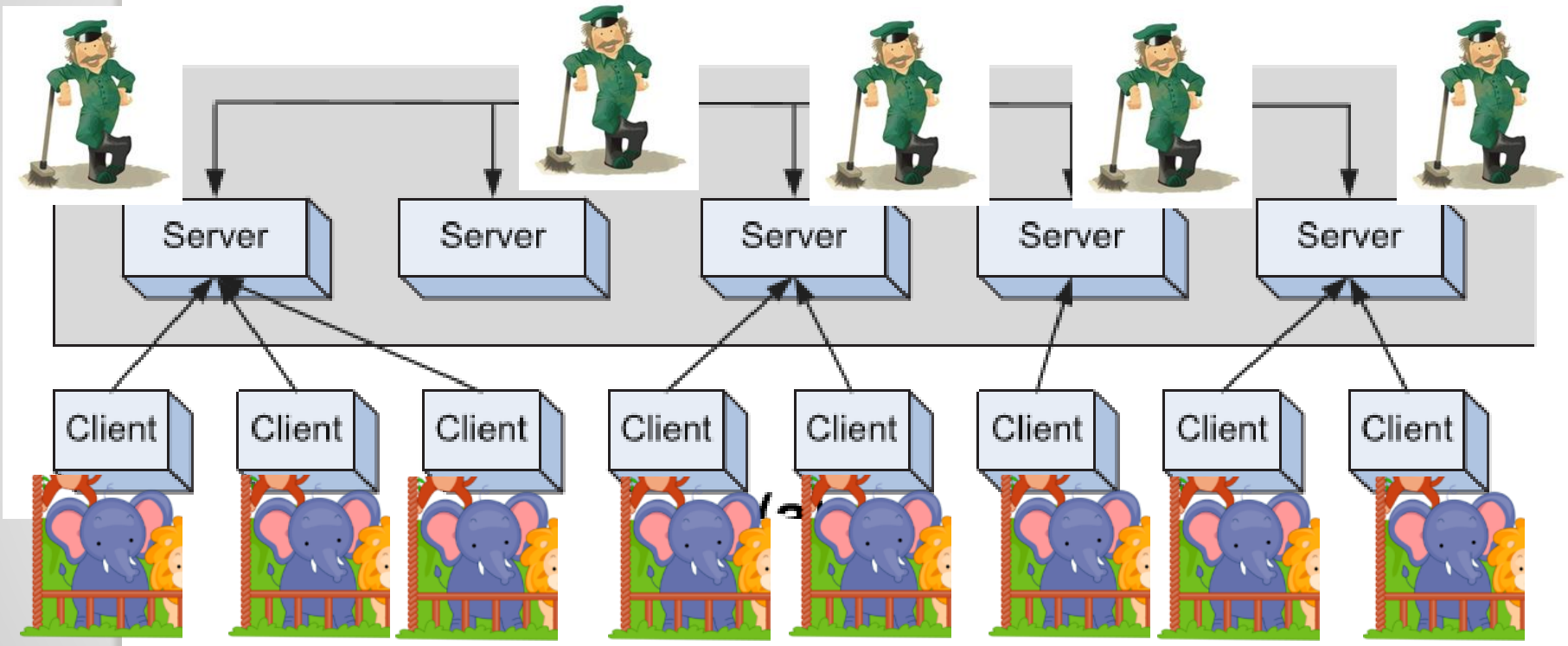
- **Coordination** is crucial for cloud applications.
- There exist **many coordination models (协调模型)**.
- They depend on the task, the type of data storage, how recovery is performed after an error, etc.

- We will discuss the popular **Zoo Keeper model** →

The Zoo Keeper model

- **Zookeper** is a service for coordinating large-scale distributed system.
- Written with the **Java** programming language
- http://zookeeper.apache.org/
- To use **ZooKeeper**, it must be downloaded and installed on **multiple servers**.

- Then **clients** can connect to any **ZooKepper** server to access the **coordination service**.


Server
Server
Server
Server
Server
Client
Client
Client
Client
Client
Client
Client
Client

The Zoo Keeper model

- The **servers** communicate with one another to **elect a leader**.

- A **database** (数据库) is replicated on all servers to keep **multiple copies of the data** used to coordinate the tasks.

The Zoo Keeper model

- Each client:
  - always communicate with the same server
  - synchronize its clock with the server.
  - can **read** or **write** data to servers

The Zoo Keeper model

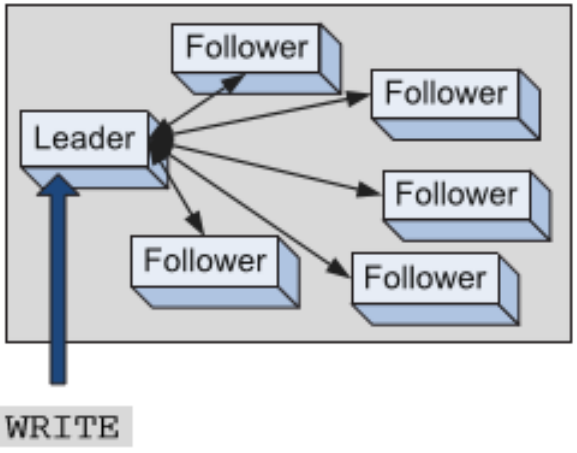
Reading data

- **Reading data** from any server returns the same data, since each server has a copy of the database.
- Because there are multiple servers, reading data can be fast.
- A client typically connect to the closest server.

The Zoo Keeper model

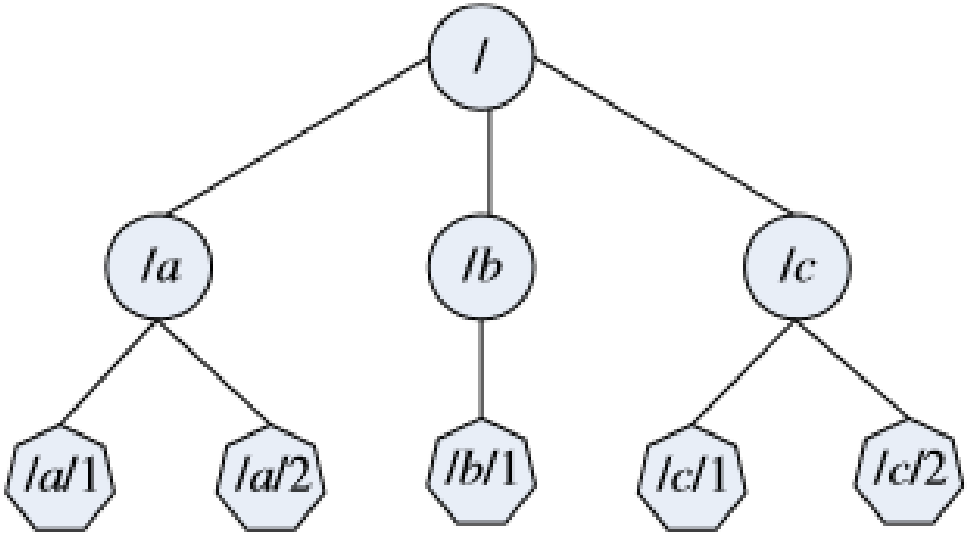
Writing data is more complicated.

- When a client ask to write data, the **leader** will ask all other servers to write the same data.
- Each server will update its local copy of the database.
- This ensures that all servers always have the same database



The Zoo Keeper model

- **The database stores data in a tree.**
- Each node in the tree has a name and can store data.
- Clients can read/write data in the tree.



Technical details: ZooKeeper is organized as a shared hierarchical namespace in which a name is a sequence of path elements separated by a backslash.

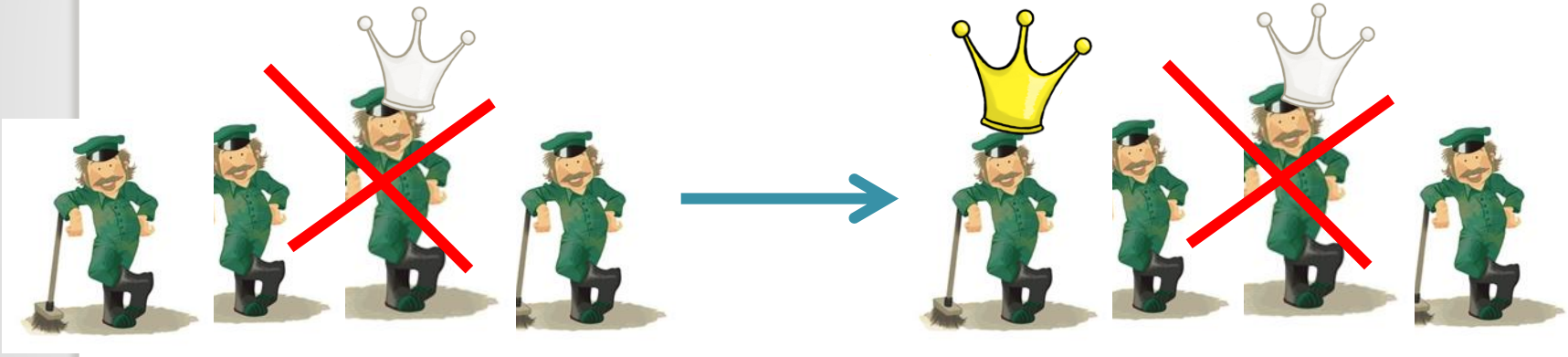
The Zoo Keeper model

Only seven operations can be performed on the tree:

- *create* – add a node at a given location on the tree.
- *delete* – delete a node.
- *get data* – `read` data from a node.
- *set data* – `write` data to a node.
- *get children* – retrieve a list of the children of the node.
- *synch* – wait for the data to propagate.

The Zoo Keeper model

If the leader fails (crash), then a new leader will be elected automatically to replace the leader.



This is the main idea about the ZooKeeper model (it is actually more complex).

The Zoo Keeper model

- **ZooKeeper** is used to perform coordination in several cloud systems.
  - It is used in **Yahoo Message Broker**,
  - It is used in the **Hadoop** stack.

Conclusion

- Today, we have:
  - we have briefly reviewed the topic of cloud infrastructure
  - we have discussed **cloud applications**

- Next week: The MapReduce model

http://philippe-fournier-viger.com/COURSES/CLOUD/

References

- Chaptre 4. D. C. Marinescu. Cloud Computing Theory and Practice, Morgan Kaufmann, 2013.