Università Degli Studi di Ferrara

Corso di Laurea in Informatica - A.A. 2023 - 2024

Tecnologie Web

Lez. 02 - Web Server

Tecnologie Web - Docente: Giacomo Piva - giacomo.piva@unife.it

Cap. 1

Tecnologie Server Side

In questa lezione…

- Modello Client-Server
- Web Server
- Soluzioni di Hosting

Modello Client-Server

Il sistema informativo Client-Server è formato da due entità:

- L’entità Client (che richiede il servizio)
- L’entità Server (che fornisce il servizio)

Le due entità che compongono il sistema riescono a comunicare tra di loro attraverso una connessione di rete.

Modello Client-Server

Il **Server** svolge le operazioni necessarie per implementare (fornire) un servizio, ad esempio:

- Inviare/Ricevere la posta elettronica (Mail server)
- Gestire le banche dati (DataBase server)
- Pubblicare documenti HTML (Web server)

Modello Client-Server

Il funzionamento di questo modello è legato alla reperibilità della rete di comunicazione fra i due “lati” che lo compongono e ha trovato una altissima diffusione nella rete Internet.

La comunicazione fra le due parti è legata anche al rispetto delle “regole” di comunicazione fra i due elementi che lo compongono, queste regole sono dette **Protocolli.**

Modello Client-Server

È un modello di comunicazione, diretta e asimmetrica

- Il **Client** contatta direttamente ed esplicitamente il server (uno solo)
- Il **Server** è in grado di fornire molti client contemporaneamente

Il modello Client/Server risolve il problema del **rendez-vous** (sincronizzazione iniziale tra i processi comunicanti) definendo il Server come un processo sempre in attesa di richieste di servizio.

Modello Client-Server

I Client devono poter specificare il servizio desiderato senza ambiguità

- I Server rendono pubblico un identificatore
- I Client devono reperire l’identificatore del servizio
- I Client devono usare l’identificatore per contattare il Server

Modello Client-Server

La parte **Client** è quella parte che l'utente vede e con la quale interagisce e fornisce un'interfaccia adatta a comunicare con gli esseri umani (gli utenti).

La funzione principale del Client è quella di contattare il Server per poter fruire del servizio inviandogli i dati necessari.

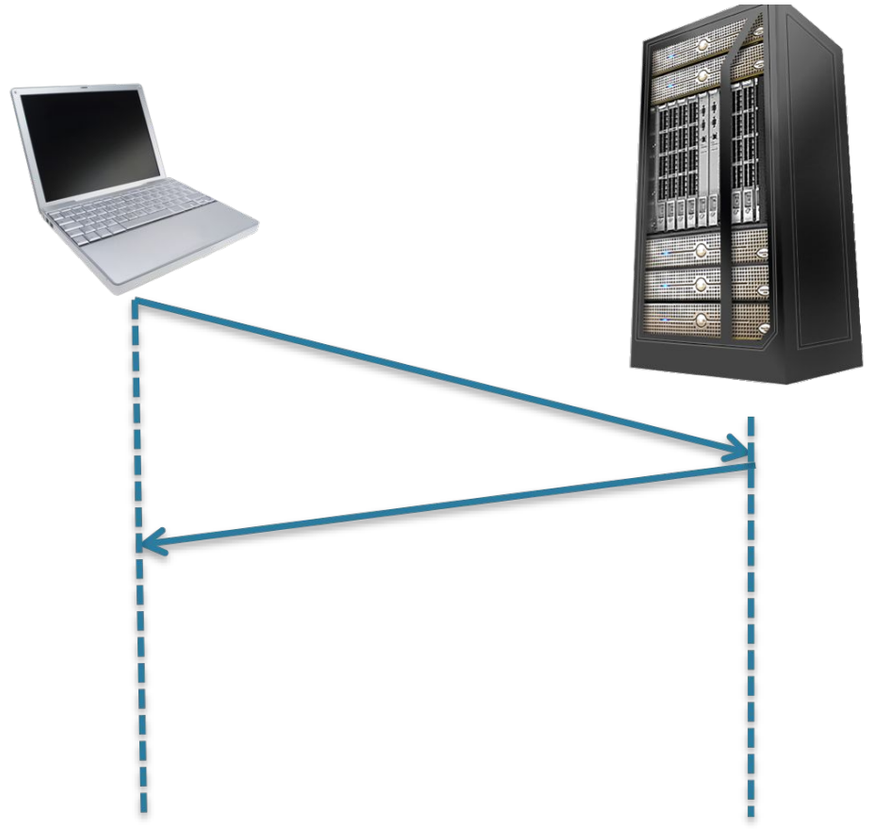
Il Client, generalmente si occupa di verificare i dati inseriti dall’utente e gestisce le risorse come ad esempio lo schermo, la tastiera ecc…

Modello Client-Server

Il modello di comunicazione client-server è basato su due entità: La prima richiede un servizio, la seconda rende disponibile il servizio.

Client

Effettua la richiesta del servizio designando esplicitamente il destinatario



Server

Risponde alle richieste del servizio accedendo a risorse e dati

Tipi di interazione

Connection oriented
Viene stabilito un canale di comunicazione virtuale prima di iniziare lo scambio dei dati (comunicazioni telefoniche).
Ad esempio il servizio ssh

Connectionless
Non c’è connessione virtuale, ma semplice scambio di messaggi (sistema postale).
Ad esempio il servizio di posta elettronica

Push e Pull

Modello Pull
Il Client richiede il servizio al server:

- Sincrono: Il Client si blocca e attende la risposta del server
- Asincrono: Il Client non si blocca, e controlla periodicamente (polling) se è arrivata la risposta dal Server

Modello Push
Il Client segnala al Server il proprio interesse ad ottenere un servizio.
Quando il servizio è disponibile, il Server contatta il Client inviando le informazioni.

Il metodo Pull con polling ed il metodo Push hanno comportamento simile e non distinguibile.

Stato

Stateful
Viene mantenuto lo stato dell’interazione, e quindi ogni messaggio dipende dal messaggio precedente.

Stateless
Lo stato dell’interazione non viene mantenuto, quindi ogni messaggio è indipendente dagli altri.

Il compito di memorizzare lo stato dell’interazione è affidato in genere al Server

Protocollo HTTP

Il protocollo su cui si basa il Web è il protocollo HTTP che è un protocollo Stateless.

Ogni richiesta fatta dal Client al Server deve essere indipendente e deve contenere tutte le informazioni per essere portata a termine senza ‘dipendere’ dalla precedente.

Protocollo HTTP

Ci sono servizi che vengono comunemente forniti sul Web, ad esempio la posta elettronica oppure i siti di e-commerce che per forza di cose fra una pagina e l'altra devono riconoscere che chi sta visitando la pagina ha fatto il login, oppure tutti gli oggetti che sono stati messi nel carrello.

I limiti del protocollo HTTP si superano a livello dell'applicazione, cioè è l'applicazione che ha il compito di implementare meccanismi che vanno a superare queste limitazioni, ad esempio attraverso i cookies.

Server concorrenti

Mentre sul lato Client eventuali esecuzioni concorrenti sono gestite dal multitasking del sistema operativo, sul Server la concorrenza è fondamentale per poter offrire un servizio a più client contemporaneamente.

Modello **iterativo**: Processa le richieste di servizio una alla volta. Basso consumo di risorse, semplicità progettuale.

Modello **concorrente**: Gestisce molte richieste di servizio alla volta. Alto consumo di risorse, maggiore complessità progettuale.

Prestazioni modello iterativo

Dal punto di vista del Client, definiamo il tempo di risposta **Tr** come il ritardo totale tra l'invio della richiesta e l'arrivo della risposta dal Server.

Ts è il tempo di elaborazione di una singola richiesta
Tc è il tempo di comunicazione medio
Tq è il tempo di accodamento medio

$$\mathbf{Tr} = \mathbf{Ts} + 2\mathbf{Tc} + \mathbf{Tq}$$

Con lunghe code di richieste, il tempo di risposta può diventare anche molto maggiore del tempo di elaborazione della richiesta

Prestazioni modello iterativo

Nel caso di Server iterativo, che risponde a una singola richiesta alla volta e accoda le altre, il tempo di risposta è circa proporzionale alla lunghezza della coda.

Con lunghe code di richieste, il tempo di risposta può diventare anche molto maggiore del tempo di elaborazione della richiesta

- Limitare la lunghezza della coda (Non può essere infinita)
- Le richieste a coda piena vengono rifiutate (Conseguenza)

Prestazioni modello concorrente

Concorrenza riesce a migliorare il tempo di risposta:

- Se la risposta richiede un tempo significativo di attesa di I/O
- Se le richieste richiedono tempi di elaborazione molto variabili;
- Se il Server esegue in un sistema hardware multiprocessore

Considerando:

Tc tempo di comunicazione medio

Tq tempo di accodamento medio (a volte trascurabile)

Tg tempo di generazione di un eventuale servitore

$$\mathbf{Tr} = \mathbf{Ts} + 2\mathbf{Tc} + \mathbf{Tq} + \mathbf{Tg}$$

Tecnologie Server Side

Durante la nostra ricerca delle tecnologie utilizzate dai siti Web più visitati, abbiamo visto:

Tecnologia	Categoria
Apache, NGINX, IIS	Web Server
MySQL, Cassandra	Database
PHP, Perl, Python, Ruby	Linguaggi di programmazione

Tutte queste sono tecnologie Server Side, ovvero tecnologie che risiedono e vengono eseguite sul Server allo scopo di fornire il servizio.

Tecnologie Server Side

Abbiamo visto come Il protocollo HTTP ed il solo linguaggio HTML abbiano dei limiti, in particolare:

- Protocollo **Stateless**;
- Tecnologia Pull;
- Linguaggio **Statico** (non è possibile eseguire condizioni o altri statement)

Tecnologie Server Side

È possibile superare queste limitazioni sfruttando altre tecnologie che risiedono sul **Server** e che vanno ad integrare le tecnologie già presentate:

- Database (Per conservare i dati)
- Linguaggi di scripting (Per scrivere algoritmi)

Queste tecnologie “affiancano” il Server Web per implementare vere e proprie applicazioni dette **Applicazioni Web.**

Le tecnologie Server Side, oltre a risiedere sul Server, vengono anche eseguite sul Server con le risorse (CPU, Ram, Disco) del server stesso.

Tecnologie Server Side

Per realizzare una Applicazione Web, abbiamo bisogno di:

- Web Server
- Database Server
- Linguaggio di programmazione

LAMP

Una delle soluzioni più utilizzate è l'installazione di uno stack **LAMP**.

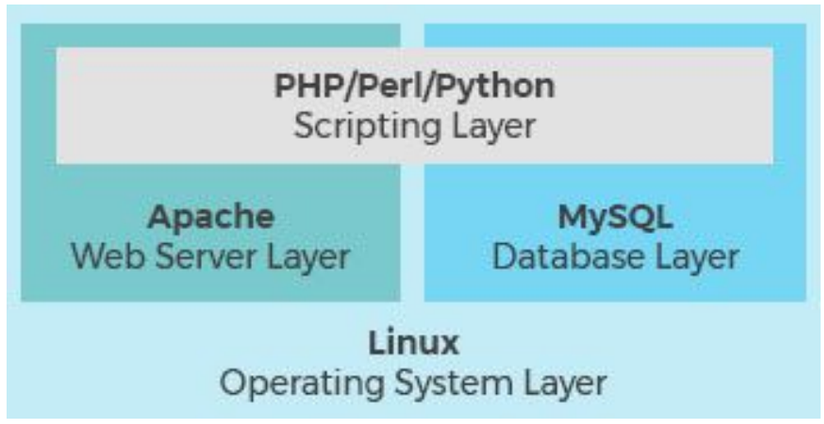
L'acronimo deriva dai componenti base dello stack:

GNU/**L**inux: il sistema operativo;

Apache: il server web;

MySQL o MariaDB: il DBMS;

PHP, Perl o Python: i linguaggi di programmazione.



LAMP

Il pacchetto software è disponibile anche per sistemi operativi diversi da Linux.

Sono disponibili anche:

- WAMP per Windows
- MAMP per MacOS

XAMPP è il pacchetto multipiattaforma disponibile per tutti gli OS unix-like e per Microsoft Windows (la X infatti sta per x-platform)

Installazione LAMP

Per installare lo stack ci sono due possibilità:

1. Digitare il comando
   `sudo apt-get install lamp-server`
2. ..ma avendo Apache già installato, possiamo semplicemente aggiungere manualmente i pacchetti rimanenti. Dovremo quindi:
   - Installare il database (mysql)
     `sudo apt-get install mysql-server`
   - Installare PHP ed il modulo per il suo interprete in Apache
     `sudo apt-get install php7 libapache2-mod-php7`

A questo punto è sufficiente riavviare Apache
`sudo /etc/init.d/apache2 restart`

LAMP

Possiamo facilmente verificare il corretto funzionamento dei vari componenti.

- Web server: è sufficiente visitare il sito `http://127.0.0.1`

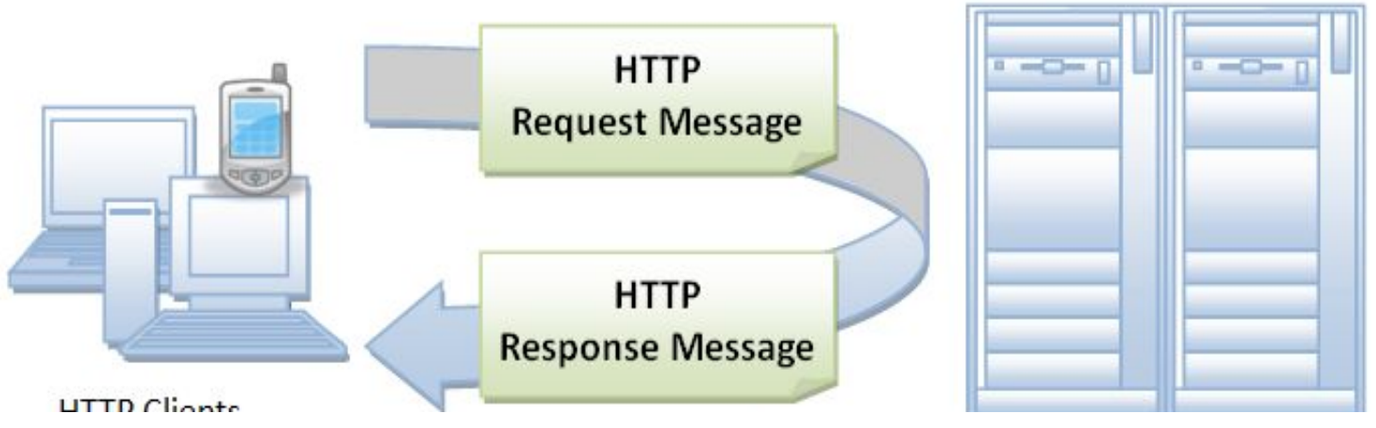
- Database: possiamo verificare che risponda semplicemente digitando il comando
  `sudo mysql –u root -p`

- PHP: basta eseguire il seguente comando, che esegue codice php senza la necessità di creare un file:
  `php -r 'echo "\n\n PHP is working fine.\n\n”;'`

Cos’è un Web Server?

E’ un software, in esecuzione su un server, in grado di gestire le richieste di trasferimento di pagine web verso un client, tipicamente un browser. La comunicazione tra server e client avviene tramite il protocollo HTTP, che utilizza la porta 80.

La versione sicura di HTTP è HTTPS (HTTP + SSL) e usa la porta 443.



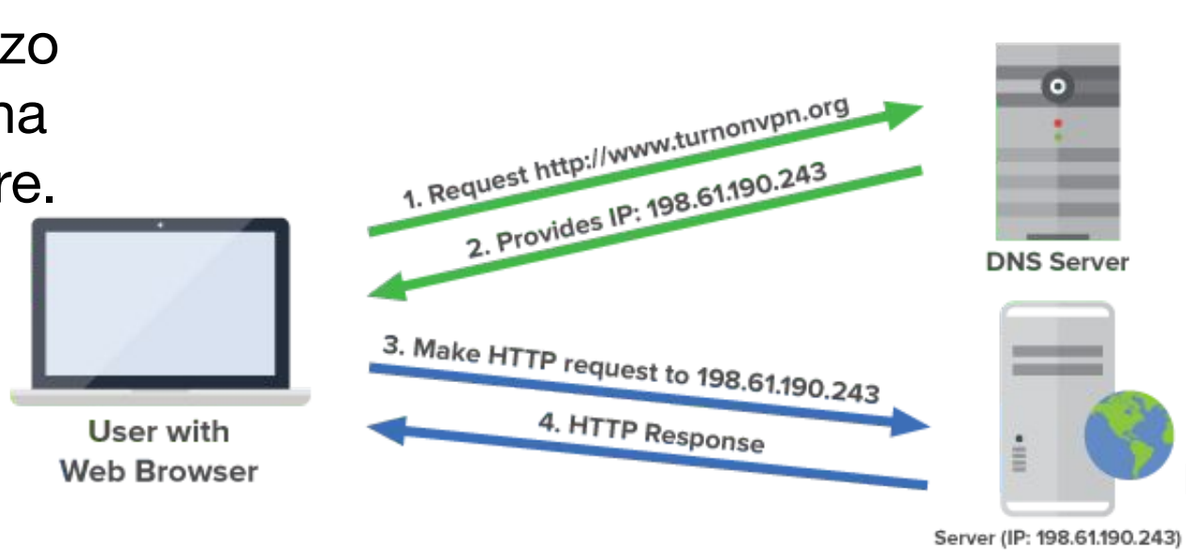
Come funziona?

1

L’utente digita nel browser l’indirizzo URL della pagina che vuole visitare.

2

Il browser interroga il DNS che «traduce» l’indirizzo digitato in un indirizzo IP ed invia una richiesta HTTP con questo IP come destinatario.



3

Il server interpreta la richiesta HTTP ricevuta e serve all’utente i files che compongono la pagina Web richiesta.

Pagine statiche e dinamiche

Pagine statiche

- Contengono solo codice HTML
- Il contenuto non è modificabile
- Viene inviato lo stesso contenuto in risposta a tutte le richieste HTTP

Pagine dinamiche

- Contengono HTML insieme a istruzioni scritte in linguaggi di scripting server-side (PHP, Java, C#, ecc..).
- Il web server interpreta le istruzioni e genera «on the fly» il contenuto della pagina richiesta.
- L'HTML generato in risposta alle richieste HTTP può essere diverso

Path translation

Per servire la pagina richiesta dall'utente i web server devono mappare il path inserito nell'URL richiesto:

- In una risorsa locale presente nel file system (per le pagine statiche)
- Oppure nel nome di un programma interno o esterno (per le pagine dinamiche)

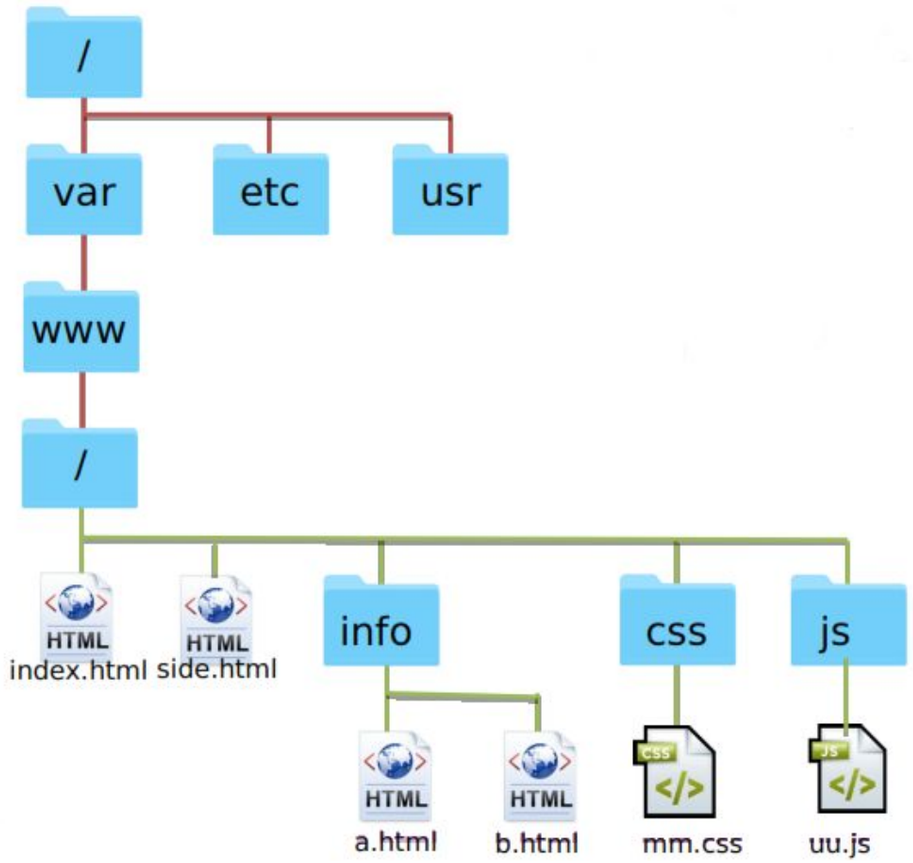
Retrieve di una pagina statica

L'URL inserito dall'utente

```
http://www.mysite.com/info/a.html
```

viene tradotto dallo user agent del client ed inserito nella richiesta HTTP

```
GET /info/a.html HTTP/1.1
Host: www.mysite.com
```

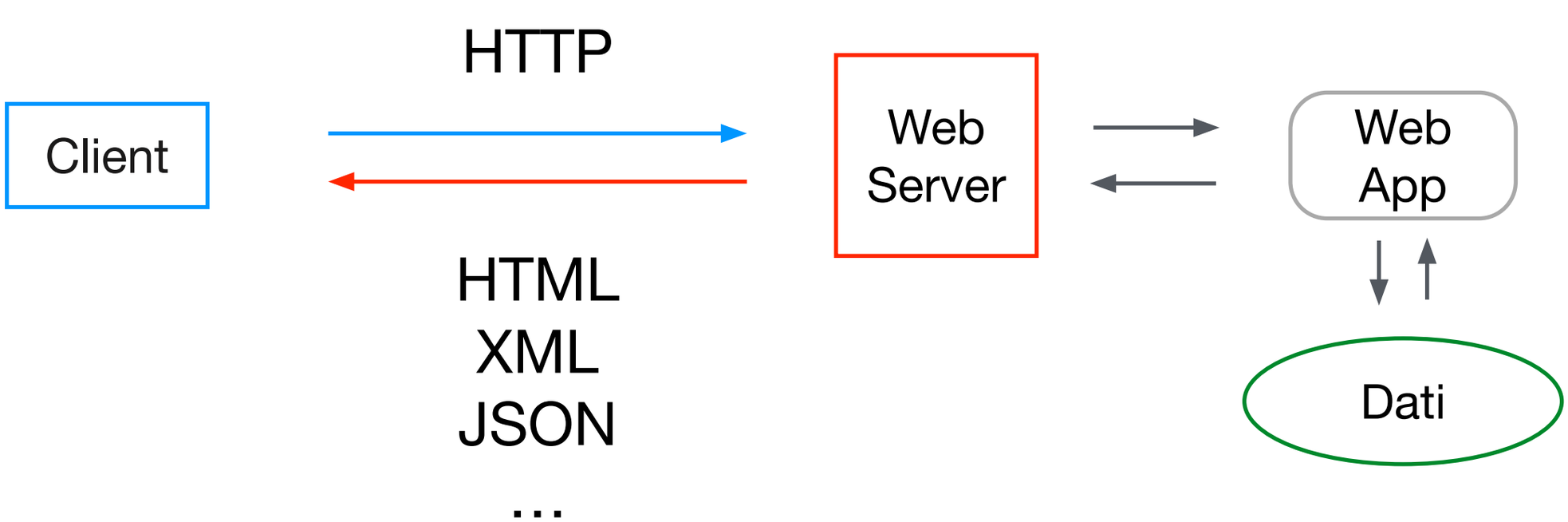


Il web server presente sull'host www.mysite.com inserirà accodando il path della richiesta a quello della sua root directory (sulle macchine Unix tipicamente corrisponde a /var/www). Nella risposta HTTP verrà inviato il file presente sul filesystem all'indirizzo

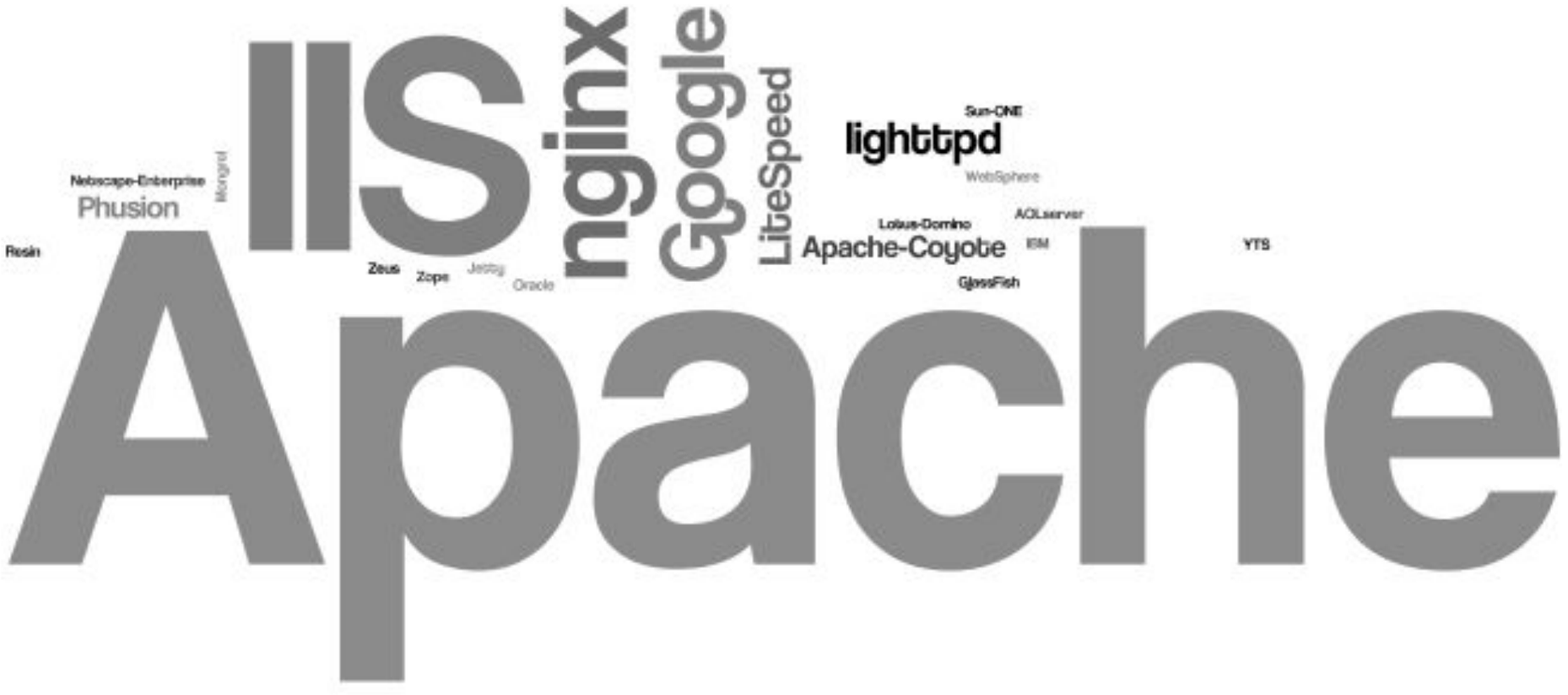
```
/var/www/path/<nomefile>.html
```

Retrieve pagina dinamica

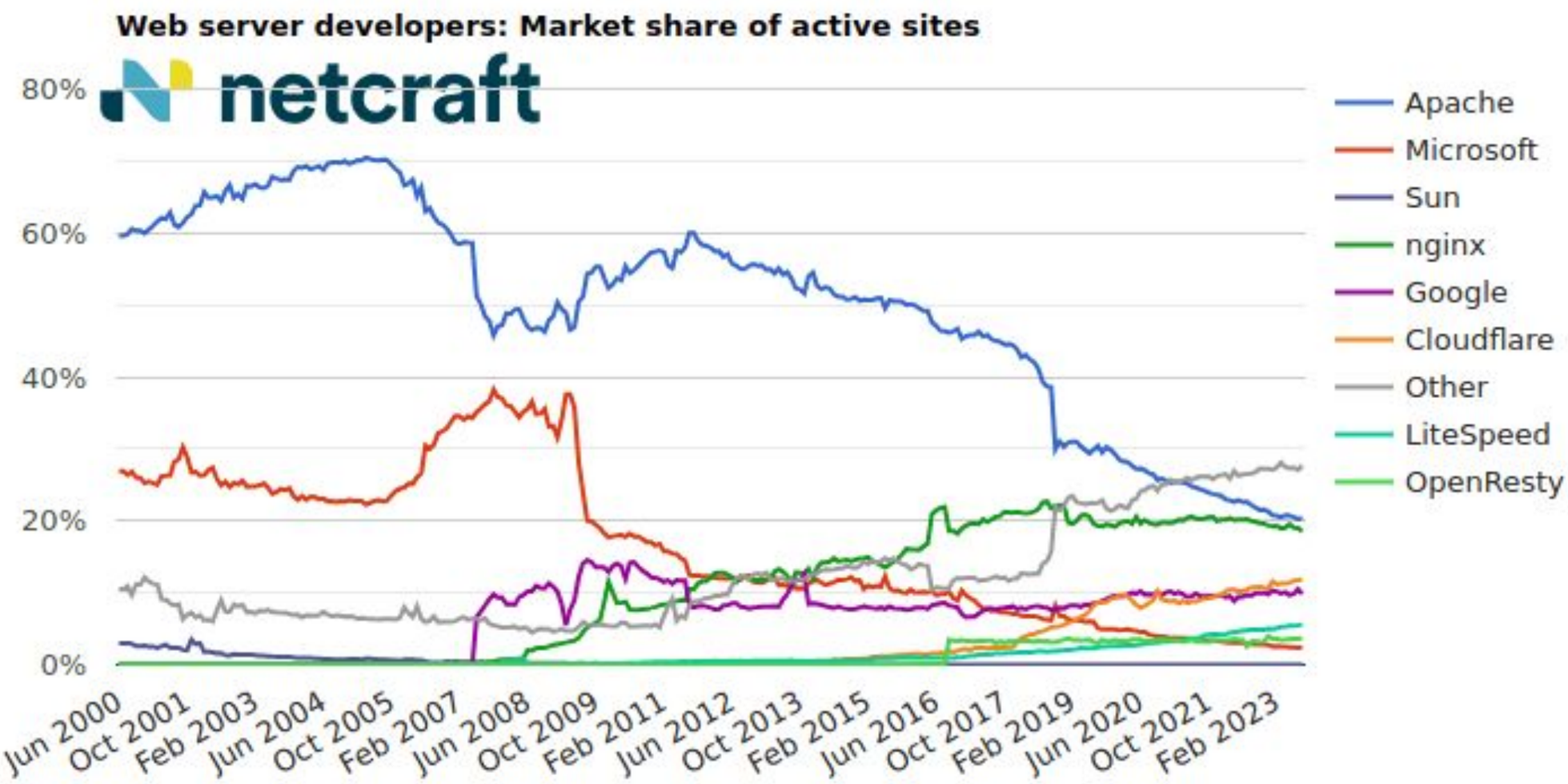
Il Web Server prima di restituire la pagina al client, esegue lo script PHP, Python, ecc…



Alcuni dei web server più diffusi



Un po’ di numeri…



https://www.netcraft.com/blog/august-2023-web-server-survey/

Tree structure

I file di configurazione di Apache sono organizzati secondo la seguente gerarchia, con radice localizzata nella directory

`/etc/apache2/`

I file di log, invece, sono nella directory

`/var/logs/apache2`

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-available
|       `-- *.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-available
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-available
|       `-- *.conf
`-- sites-enabled
        `-- *.conf
```

Directories

`mods-enabled/`

`conf-enabled/`

`sites-enabled/`

Sono directories che contengono rispettivamente snippet per la gestione di moduli (es il modulo per attivare l'interprete php), frammenti per la configurazione globale e i file per la configurazione dei virtual hosts.

Le configurazioni presenti nei file vengono attivate tramite un symlink dalle rispettive controparti `*-available/`

Files

`/etc/apache2/apache2.conf`

È il file principale che unisce i vari pezzetti di configurazione inseriti negli altri file presenti nella directory di root e nelle sotto-directory `mods-enabled` oppure `conf-enabled`.

`/etc/apache2/ports.conf`

è sempre incluso dal file precedente e indica le porte che rimangono in ascolto in attesa delle connessioni da parte dei client.

Configurazione Apache

Questa struttura gerarchica permette ad Apache di essere molto flessibile:

nelle directory `*-available` sono presenti tutte le risorse disponibili, che rimangono semplicemente disattivate se non sono necessarie. Quando servono, è sufficiente creare un link simbolico nelle rispettive directory `*-enabled` ed inserire l'include delle risorse desiderate nei file di configurazione.

Al riavvio del server la nuova configurazione sarà già attiva.

Subdirectory

Comunemente un server ospita più siti, per farlo ci sono due possibilità.

Prima modalità: subdirectories

All’interno della directory di root `/var/www/` si crea una directory per ogni sito che si vuole ospitare.

È una soluzione molto semplice e veloce per lavorare in locale ma, in produzione, presenta problemi relativi alla condivisione del contesto (cookie) in quanto l’host risulta essere lo stesso per tutti i siti.

Virtual Hosts

Seconda modalità: virtual hosting

Il **virtual hosting** è un metodo utilizzato per ospitare sullo stesso server il sito web per più di un nome di dominio, talvolta anche sullo stesso indirizzo IP.

In Apache, i siti disponibili (v-hosts) per l'attivazione sono contenuti nella directory `sites-available/`

Ogni file presente nella cartella corrisponde ad un sito per cui, all'installazione, è presente un solo file chiamato `000-default.conf`

000-default.conf

```
<VirtualHost *:80>

        /*Parametri principali*/
        ServerName www.example.com
        DocumentRoot /var/www/html

        /*Directory e file in cui salvare i log del sito*/
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

Quindi, se vogliamo abilitare un nostro sito…

1. Creiamo la directory che ospiterà i files del sito, ad esempio
   `mkdir /var/www/sito1/`
   in cui inseriamo una semplice pagina HTML chiamata index.html.

2. Ci spostiamo nella directory `/etc/apache2/sites-available` e copiamo il contenuto del file `000-default.conf` nel nuovo file `sito1.conf`
   `cp 000-default.conf sito1.conf`

3. Apriamo il file appena creato e modifichiamo parametri relativi al nostro sito.
   `ServerName sito1`
   `DocumentRoot /var/www/sito1/`

. . .

4. A questo punto possiamo «attivare» il nostro sito eseguendo il comando

   `$ a2ensite sito1`

   Ora, se andiamo a vedere il contenuto della directory `sites-enabled/` vedremo che è presente anche il link simbolico a “sito1"

5. L’ultimo passaggio per poter visitare il sito consisterebbe nella configurazione del DNS, per semplicità possiamo semplicemente editare il files `hosts` presente nella directory `/etc/`. Questo file viene sempre letto dal browser prima di consultare il servizio DNS, quindi permette di definire staticamente il mapping degli indirizzi che vogliamo rimangano invariati (tipicamente nessuno).

   Inseriamo all’inizio del file la riga:

   127.0.0.1 sito1

...

6. Facciamo il reload di Apache

   `service apache2 reload`

8. Se apriamo il nostro browser e digitiamo nella barra degli indirizzi:
   `http://sito1`
   viene visualizzato il contenuto della pagina index.html che abbiamo appena creato.

Servizi di Hosting

Esistono aziende il cui scopo è fornire ambienti integrati in cui sono rese disponibili tutte le tecnologie Server Side di cui si ha bisogno per realizzare Applicazioni Web.

Shared Hosting

Soluzione più comune ed economica:

- Risorse condivise e limitate (CPU, memoria, spazio disco, banda)
- Configurazioni limitate
- Nessuna “manutenzione”
- Economico

Esistono molte aziende che offrono un servizio di hosting condiviso molto affidabile anche per quelle attività chiave del business (es. e-commerce).

Virtual Private Server

Soluzione professionale

- Risorse condivise e limitate (CPU, memoria, spazio disco)
- Nessuna limitazione sulla configurazione
- Richiede che qualcuno si occupi della “manutenzione”
- Tutto sommato economico

Dedicated Server

Soluzione professionale per chi ha esigenze particolari

- Risorse dedicate
- Nessuna limitazione sulla configurazione
- Richiede che qualcuno si occupi della “manutenzione”
- Generalmente molto costosa

Cloud

Soluzione professionale per che tenta di mediare fra costi e flessibilità

- Risorse limitate **ma espandibili**
- Nessuna limitazione sulla configurazione
- Richiede che qualcuno si occupi della “manutenzione”
- Richiede competenze specifiche per la configurazione

Cloud

Il termine Cloud è usato per indicare una rete di server, dove ognuno dei quali ha una diversa funzione.

- Eseguire applicazioni
- Fornire servizi
- Archiviazione dei dati
- …

Il vero vantaggio è che come sviluppatori, non dobbiamo preoccuparci della gestione di un server, sarà il fornitore del servizio cloud a farlo per noi

Componenti Cloud

Storage
È il componente che si occupa di memorizzare i dati. Generalmente è composto da un NAS (Network Attached Storage) o da una SAN (Storage Attached Network).

Calcolo
Sistema che gestisce la virtualizzazione dei processi (solitamente KVM).

Controller
Il componente che gestisce tutta l'infrastruttura attraverso chiamate REST alle API.

Componenti Cloud

IaaS (Infrastructure As A Service)
Vengono virtualizzate tutte le risorse CPU, RAM, Disco …
Flessibilità di un'infrastruttura fisica senza l'onere della gestione della parte hardware (tutta la config. software è a carico dell'utente)

PaaS (Platform As A Service)
L'utente si preoccupa solamente di sviluppare il software e di caricarlo sulla piattaforma (con tecnologie compatibili alla piattaforma)

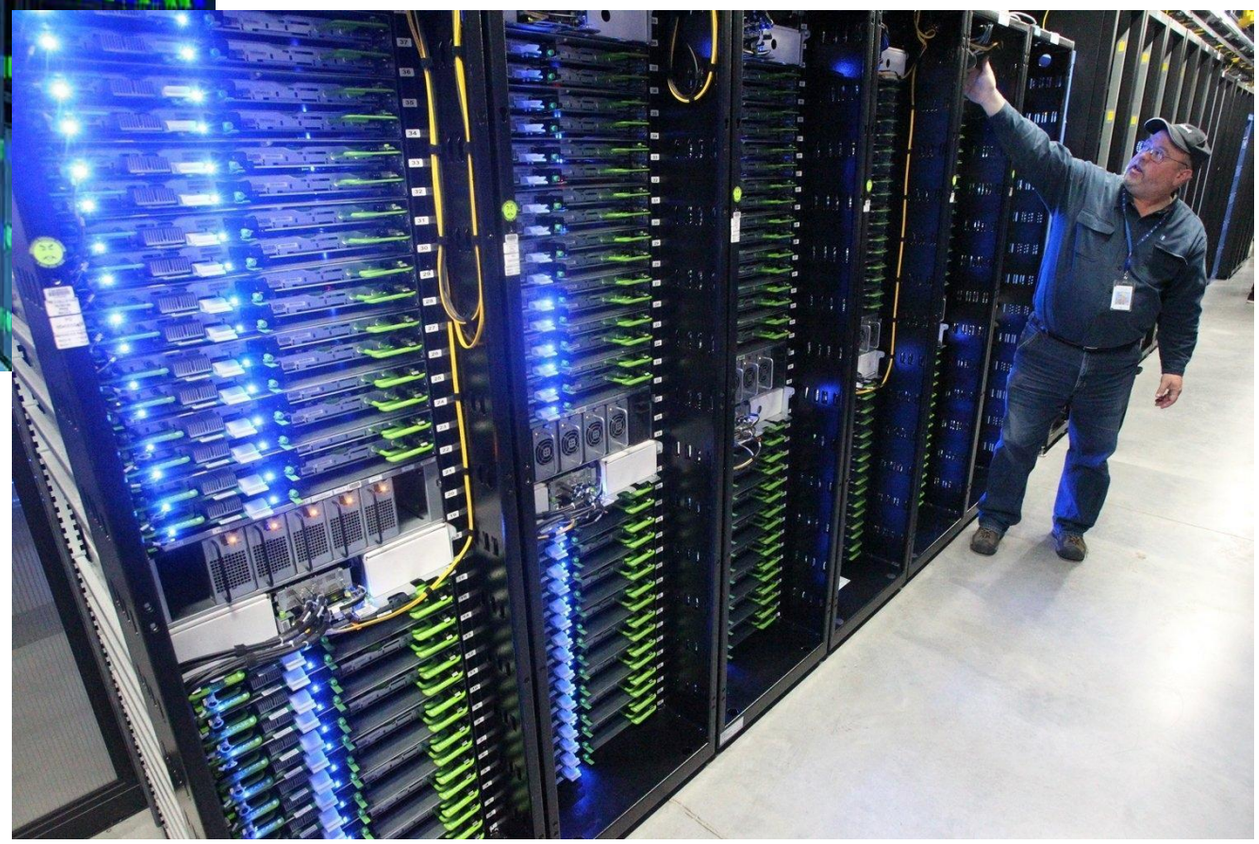
SaaS (Software As A Service)
L'utente utilizza il software caricato sulla piattaforma

Soluzioni Cloud: Amazon

https://aws.amazon.com/it/

https://aws.amazon.com/it/what-is-aws/

Amazon Data Center



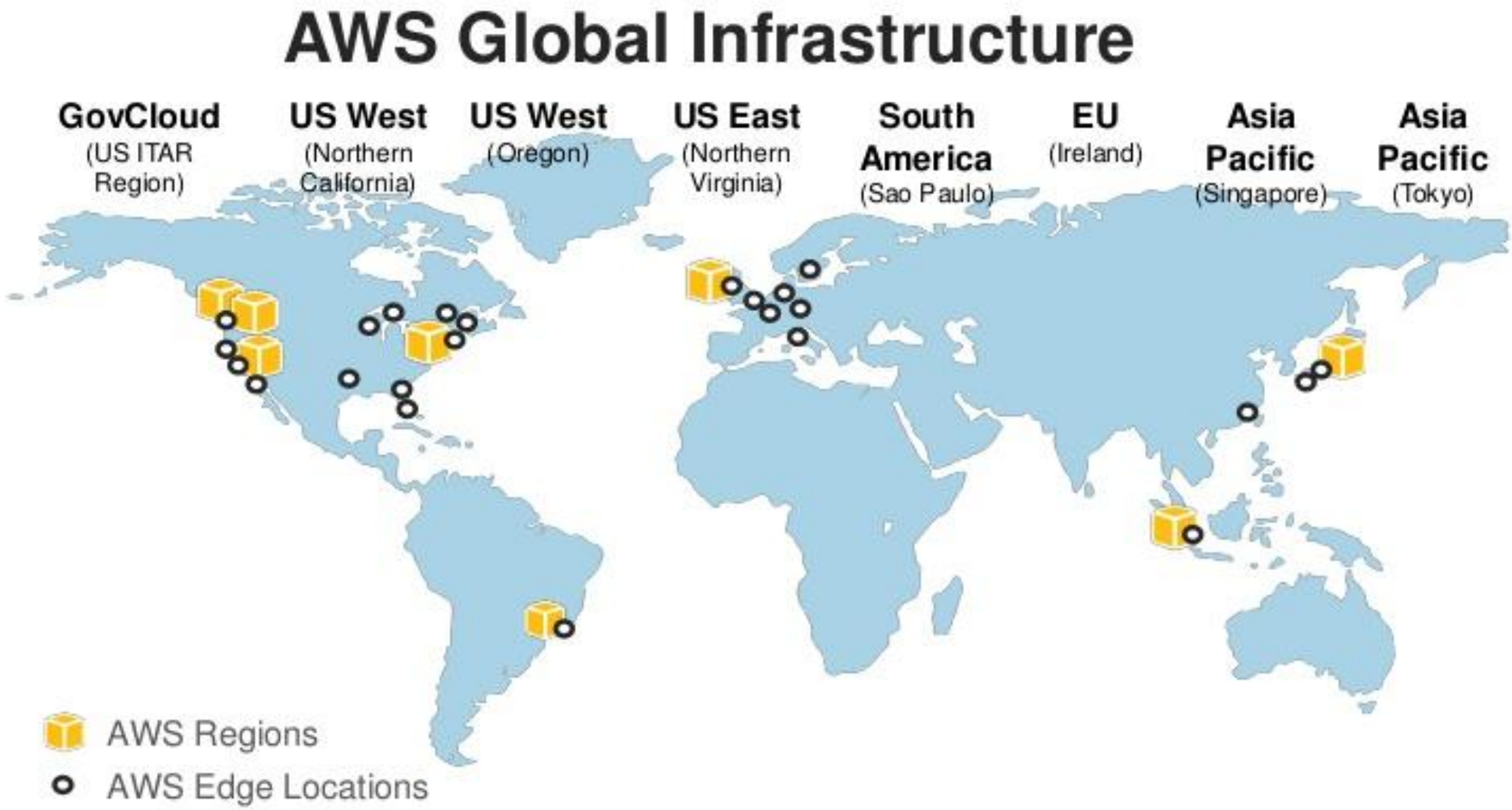
Amazon Web Service

Infrastruttura Globale

Regioni:
Insieme indipendente di risorse in una specifica area geografica

Zone:
Sottogruppi all'interno delle regioni che non condividono "point of failures" interconnesse da una rete a bassa latenza

Amazon Web Service



Esercizio

Web Server e Client

Esercizio

Realizzate, utilizzando i sorgenti a disposizione sul sito (`Server.java` e `Client.java`) un software client ed uno server.

- Il Client invia una richiesta al server che contiene il nome di un file
- Il Server ricevuta la richiesta, apre da una directory predefinita (a vostra scelta) il file con il nome inviato dal client.
- Se il file esiste il Server lo legge, conta le righe contenute nel file ed invia il valore al Client, se il file non esiste, il Server invia un messaggio di errore.

Domande?