# Arquitectura cliente-servidor

La arquitectura cliente-servidor persigue el objetivo de procesar la información de un modo distribuido. De esta forma, pueden estar dispersos en distintos lugares y acceder a recursos compartidos.

Además de la transparencia y la independencia del hardware y del software, una implementación cliente-servidor debe tener las siguientes características:

* Utilizar protocolos asimétricos, donde el servidor se limita a escuchar en espera de que un cliente inicie una solicitud.
* El acceso es transparente, multiplataforma y multiarquitectura.
* Se facilitará la escalabilidad, de manera que sea fácil añadir nuevos clientes a la infraestructura —escalabilidad horizontal— o aumentar la potencia del servidor o servidores, aumentando su número o su capacidad de cálculo —escalabilidad vertical—

Componentes de la arquitectura cliente-servidor:

* **Cliente**: De manera genérica, nos referimos a una computadora, normalmente con prestaciones ajustadas, sin embargo, en entornos cliente-servidor, se utiliza el término front end, ya que es un proceso que solicita los servicios del servidor a través de una petición del usuario. Un proceso cliente se encarga de interactuar con el usuario, por lo que estará construido con alguna herramienta que permita implementar interfaces gráficas (GUI).
* **Middleware**: Es la parte del software del sistema que se encarga del transporte de los mensajes entre el cliente y el servidor y facilita la interconexión de sistemas heterogéneos sin utilizar tecnologías propietarias. Por lo cual, se ejecuta en ambos lados de la estructura.

El middleware permite independizar a los clientes y a los servidores. Además, ofrece más control sobre el negocio, debido a que permite obtener información desde diferentes orígenes —uniendo tecnologías y arquitecturas distintas— y ofrecer de manera conjunta.

Otra característica es que los sistemas están débilmente acoplados ya que interactúan mediante el envío de mensajes.

* **Servidor**: Genéricamente un servidor es un ordenador, pero con prestaciones elevadas. Sin embargo, desde este enfoque, un servidor es un proceso que ofrece recursos y servicios a los clientes que lo solicitan (back end).

Según el tipo de servidor implantado, tendremos un tipo de arquitectura cliente-servidor diferente. A su vez, debido a que los programas y datos se encuentran centralizados, se facilita la integridad y el mantenimiento.

Conocidas estas características casi innegociables a la hora de elegir un S.O. para servidores, es común preguntarnos por qué se utilizan sistemas operativos y versiones específicas para el entorno de servidores y no usamos los mismos S.O. que tenemos instalados habitualmente en nuestras computadoras o estaciones de trabajo; eso se basa en algunas diferencias, a destacar:

* Manejo diferente del hardware: Debido al diferente propósito, los S.O. de estaciones de trabajo no pueden aprovechar todo el hardware disponible, como, por ejemplo, el manejo de memoria RAM —teniendo el caso de Windows 10 64-bit que puede manejar 6TB de RAM, mientras que Windows Server 2019 alcanza los 24TB—.
* Características soportadas: Hay funcionalidades que nativamente un S.O. de estación de trabajo no es capaz de brindar, ya que en su versión de kernel están limitadas o deshabilitadas —casos tales como la virtualización en algunas versiones de Windows 10—.
* Soporte: Algo muy importante a tener en cuenta es, cuando nuestro negocio o aplicación depende de un S.O., es el soporte por parte del fabricante/desarrollador. En el caso de los S.O. de estación de trabajo, el soporte/cobertura que tenemos es para un uso específico, si sobre esa base quisiéramos desplegar una arquitectura, por ejemplo, de servidor web, si bien es probable que nos funcione, vamos a carecer de soporte técnico ya que el fabricante nos indicará que para ese propósito esta la version “Server” del producto

En el grupo de los UNIX (o UNIX-like) tenemos a:

* GNU Linux
* FreeBSD
* macOsServer

En el grupo de Windows, integrado por toda la familia de Windows Server, tenemos las versiones que van desde la versión 2003 a 2019.

# Servicios que ofrecen los sistemas operativos

Un servicio es una puerta de entrada que el S.O. nos ofrece para que podamos ingresar y consumirlo. Los mismos pueden ser útiles para que ejecutemos programas, manipulemos archivos o asignemos recursos.

Los sistemas operativos nos pueden ofrecer una amplia variedad de servicios, en algunos casos de manera nativa o instalando un software específico para que funcionen de manera simultánea, tales como:

* Servicio de publicación web: Es el software que se encarga de despachar el contenido de un sitio web al usuario. Este proceso de despacho que, a simple vista parece muy simple, es en realidad más complejo de lo que parece, pues toda la “magia” de un webserver ocurre fuera de quien está navegando por un sitio web. El proceso de despacho comienza en nuestro navegador web. Al escribir la dirección de un sitio web y presionar Enter, comienza la siguiente secuencia: el sistema hace una búsqueda DNS para encontrar en cuál servidor está alojado el sitio en cuestión. Cuando el server es encontrado, el navegador le pide el contenido del sitio web y, acto seguido, el webserver procesa este pedido y envía dicho contenido al navegador, lo cual da como resultado la visualización del sitio en nuestra pantalla.
* Servicio de base de datos: Un servidor de base de datos, también conocido como database server o RDBMS (Relational DataBase Management Systems) en caso de bases de datos relacionales, es un tipo de software de servidor que permiten la organización de la información mediante el uso de tablas, índices y registros. En la actualidad se hace cada vez más extendido el uso de bases de datos no relacionales, conocidas popularmente como No-SQL. Si vamos a la función fundamental de un servidor de base de datos, encontraremos que es el servicio que provee de información a otras aplicaciones web o equipos/hosts, tal como se especifica en el modelo cliente servidor. En las diferentes bases de datos mediante un usuario, contraseña y nombre de host, se le permite realizar diferentes tareas, dependiendo del nivel de privilegios que posea, tales como selección, actualización y borrado de datos.
* Servicio de correo electrónico: Un servidor de correo es el encargado de enviar y recibir mensajes de correo electrónico entre hosts, usuarios o servidores. Entre sus funciones se incluyen el procesado de los mensajes, filtrado, almacenamiento, envío, recepción y reenvío de correos. Es una de las aplicaciones más populares en usar el protocolo TCP/IP, y que permite en cuestión de segundos comunicarnos con cualquier persona en otra parte del mundo. El servidor consta de dos componentes fundamentales:
  + MTA (Mail Transport Agent): En palabras simples, un agente de transferencia de correo es un software encargado de transferir el email de un host (equipo informático) a otro. Qmail, Exim, Postfix, Microsoft Exchange Server, Courier, Cyrus.
  + MDA (Mail Delivery Agent): Tiene las funciones de recibir el correo de un MTA y llevarlo al inbox de la casilla de correo, que previamente se comunicó con el servidor POP o IMAP. Existen muchos formatos utilizados por las casillas de correo, dependiendo de la configuración que haya aplicado el administrador de sistemas a cargo. Dovecot, Procmail, Maildrop.
* Servicio de archivos: Un file server (o servidor de archivos) es un servidor central de una red de ordenadores, que pone a disposición del cliente conectado un conjunto de archivos o parte del mismo. De este modo, proporciona a los usuarios un lugar de almacenamiento centralizado para los archivos en sus propios soportes de datos, disponible para todos los clientes autorizados. El administrador del servidor establece unas directrices estrictas para determinar qué usuarios tienen derechos de acceso y a qué datos. Cada sistema operativo tiene su propio servicio de compartición de archivos, es por ello que la elección muchas veces depende de las plataformas de clientes. Los servicios de archivos más utilizados son:
  + CIFS/Samba (utilizado en Linux)
  + NTFS Share (nativo de los sistemas Microsoft)
  + NFS (desarrollado por Sun y originario de los sistemas UNIX)
* Servicio de red: Es muy extendido el uso de los propios servicios del sistema operativo con el propósito de satisfacer necesidades de ruteo, firewall o proxy. Este tipo de implementaciones se aplica más tomando como base sistemas Linux, implementando sobre ellos una serie de paquetes/softwares que dan como resultado routers, firewalls o proxies. Las soluciones más utilizadas son:
  + PFSense: Stack basado en BSD, que cumple con múltiples funciones tales como Router, VPN, Firewall.
  + OPNSense: Derivado del primero y de Monowall, muy presente en el mercado de los routers corporativos.
  + DD-WRT: Basado en Linux, extendido su uso en routers hogareños, destaca por los pocos requerimientos para funcionar.
* Servicio de dominio: Un controlador de dominio (DC) es un servidor que responde a las solicitudes de autenticación y verifica a los usuarios en las redes informáticas. Los dominios son una forma jerárquica de organizar usuarios y equipos que trabajan juntos en la misma red y, el controlador de dominio mantiene todos esos datos organizados y protegidos. La responsabilidad principal del DC es autenticar y validar el acceso de los usuarios a la red. Cuando los usuarios inician sesión en su dominio, el DC verifica su nombre de usuario, contraseña y otras credenciales para permitir o denegar el acceso. Los más utilizados son:
  + En el ecosistema Windows, Active Directory, presente desde Windows 2000. Utiliza como base los protocolos LDAP, Kerberos y DNS.
  + En el ecosistema Linux, existe NIS y NIS+, originalmente llamado YellowPages. También es posible implementar LDAP en Linux y convivir con ActiveDirectory.

# Diferencias entre Windows y Linux

Linux es un software de cuya filosofía es de código abierto (open source), esto quiere decir que dicho código está a disposición para ser revisado, modificado y mejorado. En el único caso que debemos pagar es para obtener soporte técnico específico. Su denominación técnica (GNU/Linux) deriva de los proyectos GNU (sistemas open source derivados de UNIX) y del kernel Linux.

Una de las características más destacadas en Linux es la gestión de paquetes de software, la cual permite de manera muy precisa que se instale la versión/compilación de software más adecuada a nuestra distribución. Tiene dos componentes principales

* Gestor de paquetes: es quien se encarga de administrar las dependencias (por ejemplo, si tal o cual software precisa otro paquete adicional) y comparar nuestras versiones instaladas con las últimas publicadas.
* Repositorios: permiten ofrecer un catálogo de aplicaciones gestionado desde un lugar centralizado y supervisado por la comunidad de desarrolladores y usuarios. Esto da garantías al usuario de que todo el software que ha descargado es legítimo, y que además estará soportado.

A pesar de haber decenas de distribuciones “importantes”, en lo que se refiere a tipos de paquetes, existen 3:

* DPKG (Debian Package Manager): originalmente creado para Debian, y utilizado por sus derivados como Ubuntu, Linux Mint o Elementary Os.
* RPM (Red Hat Package Manager): creado por RedHat, utilizado en sus propias distribuciones, como así también en Fedora y OpenSuse.
* Pacman Package Manager: creado para ArchLinux, y presente en la mayoría de las distribuciones “livianas”, ya que aprovecha muy bien los recursos.

En su base, todos los Windows Server (considerando la última versión lanzada), son iguales, pero, en pos de ofrecer una variedad de funcionalidades y costos, Microsoft comercializa diferentes tipos de licencia, las cuales permiten acceder a ciertas funciones del sistema o poder aprovechar todos los núcleos de proceso de un servidor. Veamos algunos ejemplos:

* Essentials: apuntado para empresas de hasta 25 clientes.
* Standard: sin límite de clientes, pero con funciones de virtualización limitadas.
* Datacenter: la más completa, destaca el soporte completo para virtualización y contenedores.

Uno de los beneficios de ser una plataforma más cerrada es la estandarización de paquetes, en este caso se centran en archivos ejecutables (extensión .exe) o paquetes msi de Windows Installer (extensión .msi). Si bien con el paso del tiempo y la acumulacion de versiones, suele diversificarse las versiones, en general existen estos paquetes para un software Windows: 32bits y 64bits. Con el paso del tiempo, aparecieron gestores de paquetes desarrollados por terceros, que emulan el comportamiento de la gestión de paquetes de Linux, en pos de facilitar el mantenimiento y la actualización del software instalado. Los más destacados son Chocolatey y Ninite.

|  |  |  |
| --- | --- | --- |
|  | Windows | Linux |
| Kernel | En dos capas, una de kernel y otra de usuario.  Usa una lista de control de acceso.  Incluye el kernel los aspectos relativos a la interfaz gráfica.  Almacena sus configuraciones en registros. | De una sola capa, la del propio kernel.  Usa los permisos tradicionales de Unix para el control de acceso a archivos.  Los aspectos relativos a la interfaz gráfica se apoya en la capa de usuario.  Almacena sus configuraciones en archivos. |
| Sistema de archivos | FAT/ FAT16 / FAT32, heredado desde el MS-DOS. Tiene un límite máximo de 4 GB por archivo.  NTFS, el más utilizado en Windows ya que brinda la posibilidad de tener campos específicos para incorporar permisos de Active Directory, en la actualidad va por su 3er evolución, incorporando soporte nativo para SSD. | EXT, el cual esta ya es su cuarta evolución, es el más utilizado ya que tiene amplio soporte para las unidades SSD.  ReiserFS, de propósito general, apunta a la baja corrupción de archivos.  swap, es el sistema de archivos para los archivos de intercambio. |
| Niveles de ejecución | Modo seguro con funciones de red. Inicia en modo seguro e incluye los controladores y servicios de red  Modo seguro con símbolo del sistema. Inicia Windows en modo seguro con una ventana de símbolo del sistema en lugar  Habilitar el registro de arranque. Crea un archivo llamado ntbtlog.txt en el que que puede resultar útil para la solución avanzada de problemas.  La última configuración válida conocida. Inicia Windows con la última configuración del registro y los controladores que funcionó correctamente.  Deshabilitar reinicio automático tras error. Impide que se reinicie automáticamente en caso de que un error  Iniciar Windows normalmente. | Nivel 0: Sistema Apagado  Nivel 1: Solo usuario root, sin red ni demonios  Nivel 2: Multiusuario sin red  Nivel 3: Inicio normal, con consola  Nivel 5: Inicio normal, con consola y ademas interfaz gráfica  Nivel 6: Reinicio del sistema |

# BASH

Ya vimos que para poder interactuar con los S.O. teníamos a la interfaz gráfica (GUI) o la interfaz de línea de comandos (CLI), y supimos que la primera es más utilizada en Windows y la segunda, en Linux.

La CLI de Linux, además de ser muy potente, tiene de manera nativa la interpretación de lenguaje llamado BASH, o Bash Shell, el cual tiene como propósito la creación de scripts que facilitan la administración del sistema.

Las funciones Bash pueden:

* Eliminar tareas repetitivas.
* Ahorrar tiempo.
* Proporcionar una secuencia de actividades bien estructurada, modular y formateada.
* Con scripts, podemos proporcionar valores dinámicos a comandos usando argumentos de línea de comando.
* Puede simplificar comandos complejos en una sola unidad en ejecución.
* Una vez creada, se puede ejecutar cualquier cantidad de veces por cualquier persona. Construye una vez y ejecuta muchas veces.
* Los flujos lógicos se pueden construir utilizando funciones Bash.
* Las funciones Bash se pueden ejecutar al inicio del servidor o agregando un cron job programado.
* Los comandos pueden ser depurados.
* Tener comandos de shell interactivos.

Un ejemplo sencillo de una función Bash sería

#!/bin/bash

testfunction

testfunction(){

echo "My first function"

}

El cual como resultado de ejecución devolverá en pantalla

My first function

Las funciones bash pueden aceptar cualquier número de parámetros. El siguiente

ejemplo acepta dos parámetros:

#!/bin/bash

testfunction(){

echo $1

echo $2

}

Los scripts bash soportan:

● Bucle while

● Bucle for

● Declaración if

● Elemento lógico and

● Elemento lógico or

● Declaración Else If

● Declaración case

# POWERSHELL

Vimos que la GUI es la manera más extendida de administrar servidores Windows. Hablamos de sus ventajas en lo que a curva de aprendizaje se refiere y sus pocos cambios con el paso del tiempo, lo que hace que una vez familiarizados con estas consolas podemos dominar en varios aspectos cualquiera de las versiones de Windows Server.

Las consolas de administración, al ser componentes específicos a determinadas funciones —administración de dispositivos, discos físicos, usuarios de dominio, etc.—en determinados escenarios, tales como la automatización de procesos, pueden no ser suficientes y es allí cuando nos surge la necesidad de administrar nuestro Windows con comandos, tal como lo haríamos en Linux.

Windows siempre tuvo una interfaz de línea de comandos, pero limitada en funciones y apoyada en un lenguaje casi obsoleto como Batch.

El lenguaje de la consola incluye declaración de variables, variables especiales predefinidas, operadores matemáticos, incluyendo igualdades y desigualdades numéricas, manejo de vectores, comparación entre estos, operadores de asignación, vectores asociativos —hashtables—, valores booleanos, ciclos y ruptura de los mismos, operadores de expansión para simplificación de ejecuciones complejas —creación de vectores por medio de llamados a procedimientos, creación dinámica de vectores, etc.—; comentarios, operadores de comparación binaria, caracteres de escape, orden de ejecución, ciclos del tipo "foreach", creación de procedimientos y funciones, creación de filtros, estructuras condicionales complejas —if/then/else/elseif/switch—, operador de invocación dinámica del contenido de variables.

Los comandos de PowerShell llamados cmdlets —por command-applets—. Están estructurados de la siguiente manera: un verbo y un nombre separados por un guión (-): verbo-nombre. Vamos a ver un ejemplo, para ello abrimos nuestra consola de Powershell y ejecutamos:

Get-Command: El verbo —evidentemente en inglés— describe la acción a realizar sobre el nombre. En el anterior ejemplo, recuperamos —Get— los comandos —Command—. En este caso, el comando nos devolverá una lista de los comandos disponibles de Powershell. Con PowerShell encontramos numerosos verbos genéricos tales como Get, Set, Add, Remove, entre otros, que se combinan con diferentes nombres como Path, Variable, Item, Object, Computer, etcétera. Los nombres que constituyen los comandos están siempre en singular y esto es válido también para los parámetros. Por lo tanto, es posible, mezclando verbos y nombres, acordarse fácilmente de un buen número de comandos. Hay que tener en cuenta que los comandos, así como sus parámetros asociados, pueden escribirse indistintamente en mayúsculas o en minúsculas. El analizador sintáctico PowerShell no es case sensitive.

**Modelo de parcial**

¿Cómo se denomina la parte del software que se encarga del transporte de los mensajes entre el cliente y el servidor?

1. Windows Server
2. Middleware
3. GNU/Linux

¿Cual es el protocolo encargado de enviar datos y confirmar su recepción?

1. TCP
2. UDP

c ) Ninguno de los anteriores

¿En cuales runlevels Linux inicia la consola por defecto?

1. Runlevel 1
2. Runlevel 5
3. Runlevel 6

En seguridad informática, ¿qué significado cobra la sigla CIA? (con sus términos traducidos al español)

1. Confidencialidad, Integridad, Autenticación
2. Configuración, Integridad, Autenticación
3. Confidencialidad, Integración, Disponibilidad

PowerShell Core es…

1. Un lenguaje de scripting
2. Una consola multiplataforma
3. Todas las anteriores

Bash se puede usar para…

1. Automatización de tareas administrativas
2. Automatización de procesos
3. Todas las anteriores

¿Cuál es la capa del protocolo OSI que se encarga del ruteo de los paquetes?

1. Capa 1
2. Capa 3
3. Capa 4

PowerShell se puede usar para…

1. Automatización de tareas administrativas
2. Automatización de procesos
3. Configuration Management
4. Todas las anteriores

¿Cuáles de todas estas tecnologías no es un Hipervisor de Virtualización?

1. VMWare
2. VirtualBox
3. Debian

¿Qué es un algoritmo de transposición?

1. Un algoritmo que permite convertir un mensaje escrito en una matriz de forma horizontal y leerlo de forma vertical con el fin de cifrarlo.
2. Un algoritmo que permite cifrar un mensaje reemplazando caracteres por otros (ej. todas las 'a' se reemplazan por 'c', todas las 'b' se reemplazan por 'd', etc.)
3. Un algoritmo que permite ocultar un mensaje dentro de otro activo, como por ejemplo, una imagen.

Explicar con sus palabras el patrón cliente - servidor.

En el patrón cliente - servidor, el cliente realiza una petición, esta pasa por el middleware y llega al servidor. El servidor procesa la petición y envía una respuesta, la cual pasa por el middleware y llega al cliente.

Explicar con sus palabras el funcionamiento del comando ping.

El comando ping establece una conexión, enviando un paquete mediante el protocolo ICMP al destino, el cual responde confirmando la conexión. De esta manera se calcula el tiempo de respuesta y se comprueba que el destino se encuentra disponible.

Explicar con sus palabras los principales beneficios al virtualizar sistemas.

Mayor aprovechamiento de recursos.

Contar con diferentes sistemas operativos en un mismo equipo.

Proporciona un entorno aislado con fácil mantenibilidad.

Disponer de un nuevo ambiente/sistema sin la necesidad de adquirir nuevo hardware.

Backups y snapshots.

Explicar con sus palabras cuales son los usos tradicionales de los sistemas operativos en el contexto del centro de cómputos.

Para ofrecer servicios de almacenamiento y procesamiento de datos, alojamiento de aplicaciones web, servicio de correo electrónico, etc.

Explique con sus palabras cuál es la función de un algoritmo en criptografía.

Un algoritmo de criptografía permite cifrar y descifrar un mensaje utilizando una técnica específica y una clave, añadiendo una capa de seguridad de manera que el mismo solo pueda ser leído por el receptor deseado.

Al virtualizar sistemas se logra:

- Un mayor aprovechamiento de los recursos ya que se puede disponer de varios servidores virtuales en un único servidor físico, de manera que se reduzca el porcentaje de recursos ociosos.

- Posibilidad de correr diferentes sistemas operativos al mismo tiempo.

- Simplicidad a la hora de realizar backups y snapshots.

- Contar con un entorno aislado donde se pueden realizar pruebas.

- Reducción de costos y rápida provisión de servidores (sin la necesidad de tener que adquirir hardware cada vez).

- Rápida configuración y despliegue más sencillo.

- Posibilidad de contar con alta disponibilidad al hacer uso de la redundancia.

El protoclo UDP, a diferencia de TCP, no espera la confirmación de la recepción por parte del destinatario (ni realiza reintentos en caso de que no se reciba), de manera que la cantidad de información adicional que se transmite es menor, dando lugar a una transmisión más rápida. Esto da lugar a que sea usado en aplicaciones como juegos online, streaming, etc.

En la arquitectura cliente-servidor existe un cliente (dispositivo del usuario, en general con pocos recursos) que realiza una petición, esta pasa por el middleware y llega al servidor. El servidor (equipo -uno o varios- con grandes prestaciones para atender las solicitudes de todos los usuarios) procesa la petición, realiza las acciones necesarias y envía la respuesta, la cual pasa por el middleware y llega al cliente.

Linux es un kernel (el núcleo o la parte más importante de un sistema operativo). Existen muchos sistemas operativos que hacen uso de dicho kernel y agregan sobre este diferentes aplicaciones para distintos usos y gustos. A estos sistemas operativos que hacen uso del kernel Linux se los conoce como distribuciones de Linux: Ubuntu, Debian, Fedora, etc.

A diferencia de lo utilizado habitualmente en nuestras computadoras, en los servidores se hace uso de diferentes servicios como los de publicación web (procesamiento de solicitudes http y respuesta con el contenido solicitado), correo electrónico (envío y recepción de emails), archivos (almacenamiento centralizado), bases de datos (almacenamiento y manipulación de datos), red (enrutamiento y firewall) y dominio

¿Qué otros servicios se ofrecen más allá de Compute y Storage? Enumere al menos 3

- Bases de datos

- Machine learning

- Internet de las Cosas (IoT)

- Analytics

Explicá con tus palabras los principales beneficios al virtualizar sistemas.

- Mayor aprovechamiento de los recursos ya que se puede disponer de varios servidores virtuales en un único servidor físico, de manera que se reduzca el porcentaje de recursos ociosos.

- Posibilidad de correr diferentes sistemas operativos al mismo tiempo.

- Simplicidad a la hora de realizar backups y snapshots.

- Contar con un entorno aislado donde se pueden realizar pruebas.

- Reducción de costos y rápida inicialización de servidores.

En cloud computing, de que consta el modelo de responsabilidad IaaS? Y Cómo se diferencia del resto de los modelos de responsabilidad

El modelo de infraestructura como servicio es el menos restrictivo de los que se ofrecen en las diferentes nubes públicas. En él, el proveedor se encarga de administrar los recursos físicos, brindándonos por ejemplo una instancia de EC2 (nos abstrae del hardware y la virtualización que haya en el servidor), pero es el consumidor el que se encarga de todo lo que está por encima: sistema operativo a instalar, configuraciones, aplicaciones, etc. Se diferencia de los demás en las tareas de las que se encarga cada parte y en la libertad que el consumidor tiene: en IaaS podemos instalar las aplicaciones que querramos (implicando eso sí mayor trabajo de nuestra parte), en PaaS ya dispondremos de, por ejemplo, un servidor de bases de datos pero nuestras acciones estarán limitadas a la configuración de esa aplicación, y en SaaS solamente hacemos uso de la aplicación proporcionada.

Cómo definirías elasticidad en Cloud computing?

La elasticidad consiste en la capacidad de modificar o adaptar los recursos aprovisionados de acuerdo a la demanda. De esta manera, se pueden aprovisionar más recursos en horas pico donde las cargas de trabajo son mayores y reducir los mismos en horas menos concurridas. Con el propósito de lograr esto, las diferentes nubes proporcionan funcionalidades de autoescalado.

Explicá con tus palabras el patron / arquitectura Cliente - Servidor.

La arquitectura cliente-servidor tiene como propósito el procesamiento de la información de manera distribuida, de manera que los usuarios accedan a ella desde cualquier lugar. En la misma existe un cliente, es decir, un dispositivo del usuario que normalmente interactúa a través de una interfaz gráfica y realiza una petición. Esa petición pasa por el middleware y llega al servidor. El servidor (equipo con grandes prestaciones para atender las solicitudes de todos los usuarios) procesa la petición, realiza las acciones necesarias (como por ejemplo consultas a una base de datos) y envía la respuesta, la cual pasa por el middleware y llega al cliente.