

Componente formativo

**Selección de la arquitectura de software**

**Breve descripción:**

En el presente componente formativo se estudiarán diferentes conceptos como *networking*, modelos de arquitectura, lenguajes de programación, servidores web, protocolos de comunicación, entre otros, los cuales le brindarán las bases para su trabajo como desarrollador *full stack.*

**Área ocupacional:**

Ciencias naturales, aplicadas y relacionadas

**Noviembre 2022**

Tabla de contenido

[Introducción 3](#_Toc142488594)

[1. Networking 4](#_Toc142488595)

[2. Sistemas operativos de red e infraestructura 10](#_Toc142488596)

[3. Modelos de arquitectura 14](#_Toc142488597)

[4. Lenguajes de programación y stack tecnológico 19](#_Toc142488598)

[5. Gestor de base de datos y servidores 23](#_Toc142488599)

[6. Protocolos de comunicación 30](#_Toc142488600)

[Síntesis 33](#_Toc142488601)

[Material complementario 34](#_Toc142488602)

[Glosario 36](#_Toc142488603)

[Referencias bibliográficas 37](#_Toc142488604)

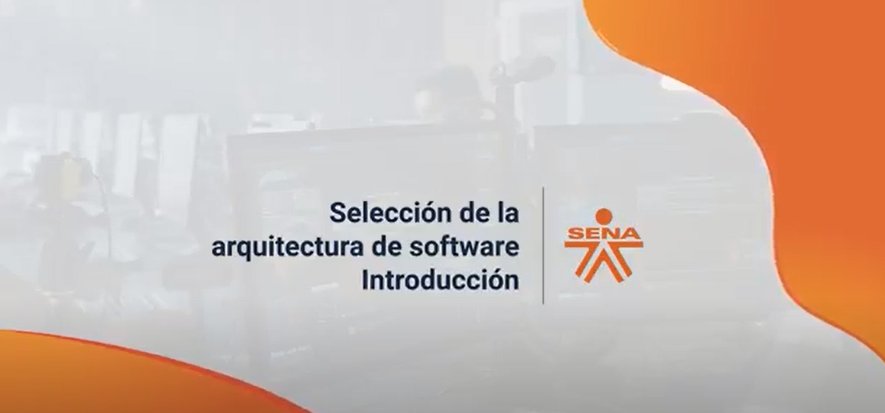
[Créditos 38](#_Toc142488605)

Introducción

Apreciado aprendiz, bienvenido a este componente formativo, donde se darán algunas bases para su futuro desempeño como desarrollador web full stack.

En el siguiente video conocerá, de forma general, la temática que se estudiará a lo largo del componente formativo.

1. Selección de la arquitectura de software - Introducción



[**Enlace de reproducción del video**](https://www.youtube.com/watch?v=fh8LtYRvL3s)

|  |
| --- |
| **Síntesis del video: Selección de la arquitectura de software - Introducción** |
| Bienvenido a este recorrido por el importante mundo de la programación, tendrá conocimiento de todos los conceptos básicos expuestos en el presente componente que le van a ayudar a entender mejor la construcción y el funcionamiento de las aplicaciones que desarrolle como profesional en este campo de la tecnología.  Estudiará sobre *networking* o redes, características y tipos que forman parte de su estructura. Sistemas operativos e infraestructura tecnológica sobre equipos de cómputo, los servidores más conocidos y utilizados en el mercado, memoria RAM, procesadores, almacenamiento.  Se reconocerán algunos patrones y modelos arquitectónicos de software para aplicaciones web, arquitectura orientada a servicios (SOA) y microservicios, lenguajes de programación y sus diferentes clasificaciones, stack tecnológicos y ejemplos de estos, algunos gestores de bases de datos, tanto de base de datos relacionales como no relacionales, servidores y sus tipos, protocolos de comunicación y sus características.  Estimado aprendiz: es durante el recorrido de este derrotero que logrará adquirir y afianzar los conocimientos que se han especificado con anterioridad. Son fundamentales su compromiso y la disciplina de estudio como ingredientes para culminar de manera satisfactoria el estudio de este componente. |

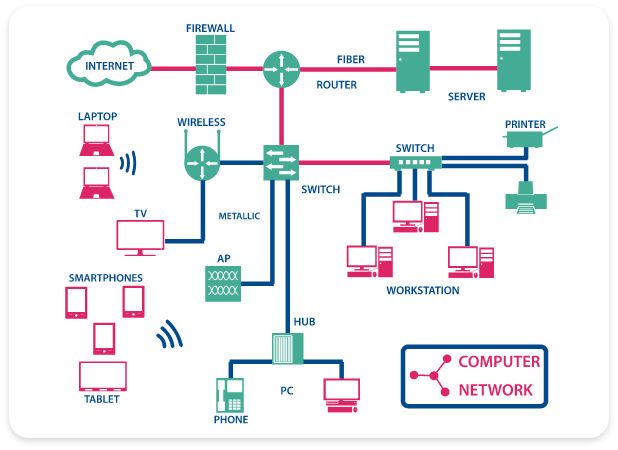
# Networking

Para dar inicio a esta temática, debemos tener claro que la conexión en terminales o estaciones de trabajo entre sí es fundamental para que las empresas o las personas, tengan buena comunicación. Un ejemplo de una gran conexión de conexiones es el internet.

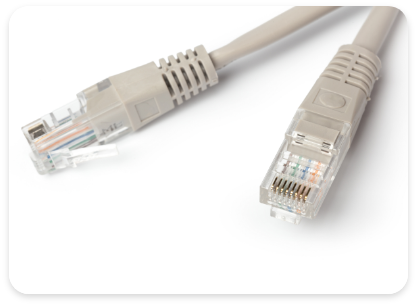
El término networking en el presente contexto se refiere a: “redes”, redes de datos para comunicación e intercambio de información entre dispositivos o terminales. Básicamente una red de datos está compuesta por dispositivos, medios de comunicación y servicios.

**Conozcamos algunos aspectos importantes del networking.**

1. **Dispositivos de red:** Se refiere a las terminales de trabajo como servidores, computadoras convencionales, tabletas, móviles, teléfonos IP, etc.; al igual que aparatos que logran conexión con estos como son: routers, switches, hubs, firewalls, etc.



1. **Medios de comunicación:** Son los canales a través de los cuales viaja la información: cables UTP, de fibra óptica o wifi. Dependiendo de las conexiones que tenga con los medios mencionados, tendrá una velocidad de transmisión. Por ejemplo, la conexión con fibra óptica es indicada para largas distancias de transmisión de datos, los cables UTP son recomendables para una conexión de máximo 100 metros, el wifi tiene menos alcance de conexión, y puede tener más interferencia, lo que implica menos velocidad de conexión.



1. **Servicios:** Se refiere a los programas y aplicaciones que están a lo largo de la red de comunicaciones. Ejemplo: servicio de correo, servicios de páginas web, etc.

El proceso de comunicación en una red de datos se asemeja al proceso de comunicación común, en el que participan de forma básica: un emisor, un canal de comunicación, un receptor, un mensaje y un código; puede existir retroalimentación.

Imagine que está chateando por WhatsApp con un compañero del programa para ponerse de acuerdo en realizar una actividad. El emisor sería usted y el receptor su compañero, quien responde el mensaje; entonces, hay una retroalimentación. Supongamos que el mensaje es un texto (código), este necesita un medio para viajar, pero resulta que está conectado al wifi de su casa; así, primero viajaría por ondas magnéticas, luego el dispositivo que emite dicho wifi está conectado a un cable y por allí también viajaría la información, ese sería entonces el canal o medio de comunicación.



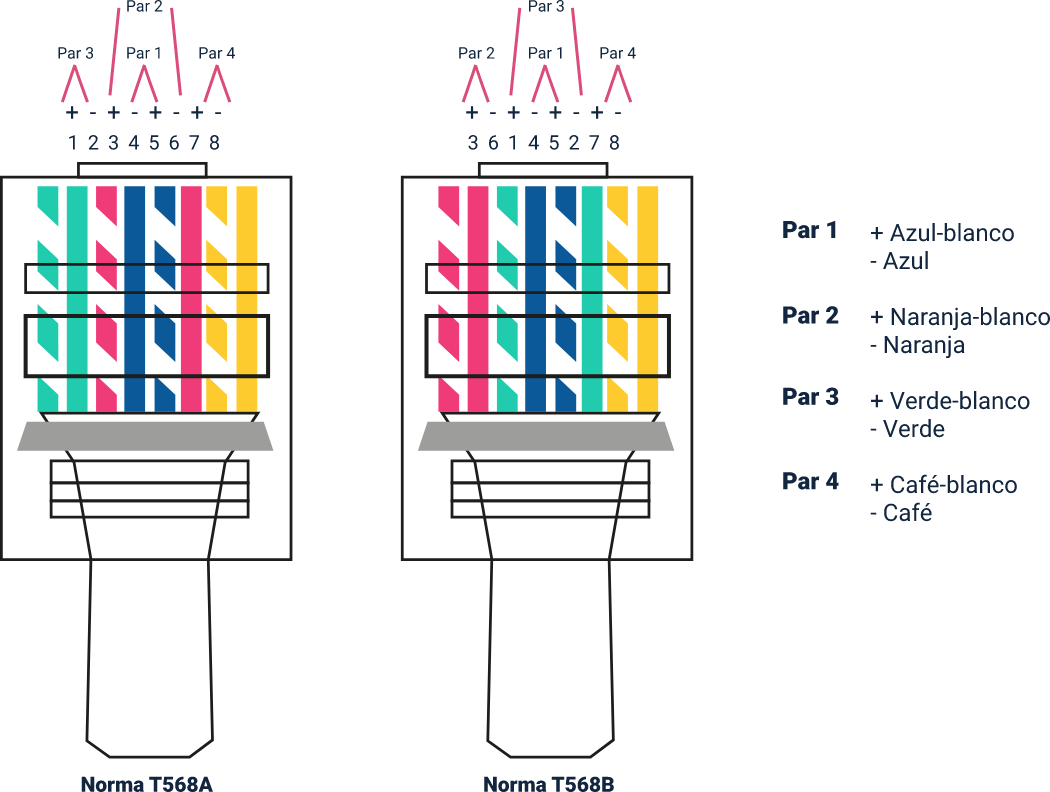
Ahora, en los medios de comunicación, es importante tener en cuenta la velocidad y frecuencia de conexión con cables UTP dependen de algunas categorías que se detallan en la siguiente tabla:

1. Velocidad y frecuencia de conexión

| Categoría | Velocidad | Frecuencia |
| --- | --- | --- |
| Cat 5 | 100 Mbits/s | 100 MHz |
| Cat 5E | 1000 Mbits/s | 100 MHz |
| Cat 6 | 1000 Mbits/s | 250 MHz |
| Cat 6a | 10000 Mbits/s | 500 MHz |
| Cat 7 | 10000 Mbits/s | 600 MHz |

Según la anterior tabla, debería configurar su cable con conectores RJ45, lo que en el común se escucha como “ponchar”, tal como lo muestra la siguiente imagen:

1. Configuración cable



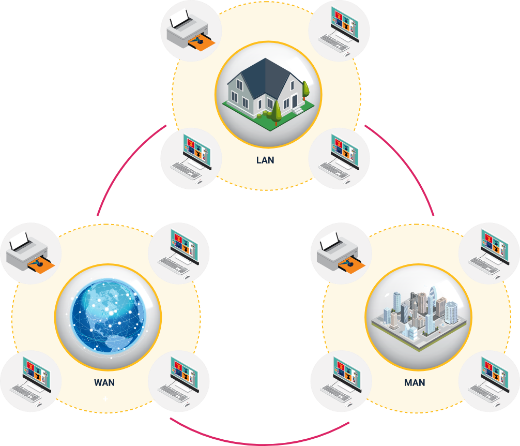
**Internet**

El internet y la web son dos términos que se suelen confundir en el diario vivir, pero son distintos. La internet, es la llamada “red de redes”, no es una red única, y la web se ejecuta sobre internet. Es un “sistema distribuido”. No cabe duda de que ambos son de los avances tecnológicos más importantes en los últimos años, y de la informática en general.

En la actualidad, basta con acceder a un navegador web (Chrome, Mozilla, Safari, etc.), ingresar la dirección de una página web y allí poder realizar un sinfín de operaciones: compras, apuestas, gestionar un negocio, ver películas.

Uno de sus objetivos como como desarrollador web full stack es que pueda construir aplicaciones mediante las que se realicen tareas como estas; y aún más.

El networking posee unas características entre las que se encuentran la tipología y la topología de redes. A continuación, conoceremos cada una de ellas.



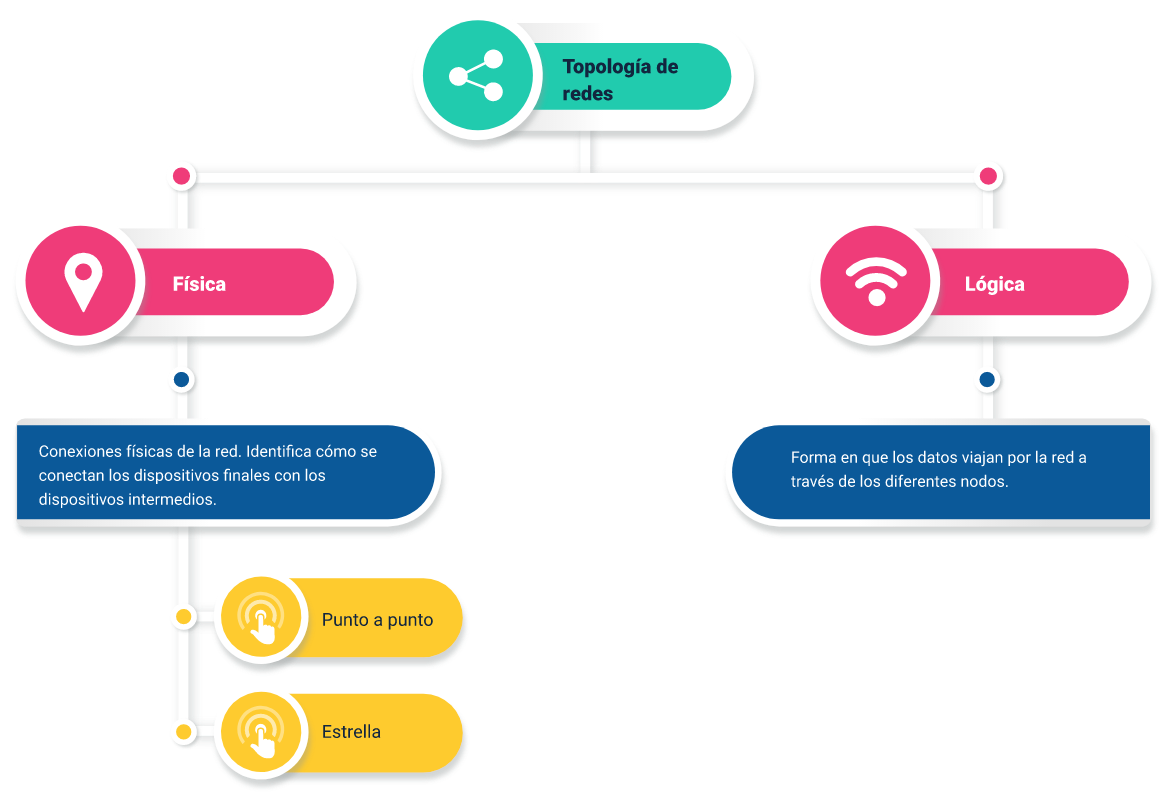
**En la tipología de redes se encuentran:**

1. **LAN (Red de área local):** Son redes privadas de escasos kilómetros de longitud que se encuentran en una misma vivienda, edificio o campus.
2. **MAN (Red de área metropolitana):** Es una red que abarca una ciudad; ejemplo: la televisión por cable.
3. **WAN (Red de área amplia):** Es aquella que abarca un área geográfica grande: un país o un continente.

**Topología de redes**

Representa la forma como se interconectan los dispositivos de una red. Se encuentran en los tipos mencionados anteriormente (LAN, WAN, MAN), y se clasifican en dos: física y lógica. El siguiente mapa conceptual nos presenta un resumen de la topología de redes.

1. Topología de redes



Ahora, en el networking encontramos dos modelos que debemos tener presente: el modelo OSI y el modelo TCP/IP.

**Conozcamos cada uno de ellos:**

1. **El modelo OSI:** El Modelo OSI (Interconexión de Sistemas Abiertos) se encuentra dentro de los modelos de referencia: son abstracciones para facilitar la comprensión de los protocolos de comunicación y la arquitectura de los sistemas utilizados para interrelacionar distintos programas y equipos.

Es un modelo propuesto por la Organización Internacional de Estándares (ISO) como un primer paso hacia la estandarización internacional de los protocolos utilizados en varias capas.

1. **El modelo TCP/IP:** Es un modelo de referencia utilizado en ARPANET y en el Internet. Puede encontrar información sobre ARPANET en la página web oficial de [Mozilla](https://developer.mozilla.org/es/docs/Glossary/Arpanet).

Las capas de los modelos OSI y TCP/IP se resumen en la siguiente imagen:

1. Capas de los modelos OSI y TCP/IP

*Capas de aplicación: compuestas por, la aplicación, la presentación, y sesión. 
*Capas de flujo de datos: compuestas por, transporte, red, enlace de datos y física. 

# Sistemas operativos de red e infraestructura

Un sistema operativo (SO), en términos generales, es un software especial que “controla” el hardware; la parte física de un “sistema de cómputo” es lo que se llama hardware, o sea el monitor, teclado, mouse o ratón, CPU, entre otros. Software se refiere a lo que no es físico; por ejemplo, el paquete ofimático de Office, LibreOffice, un videojuego, una aplicación web, una aplicación de escritorio.

En el siguiente esquema se puede observar un resumen de las principales capas de un SO:

1. Principales capas de un SO

Un Sistema Operativo, se compone de: 1.
1.Kernel o núcleo: administración y uso de componentes del SO: procesos, entradas y salidas, memoria y archivos. 
2. Servicios: funciones o componentes para fácil uso de hardware: interface gráfica, servicios de red, ficheros, procesos y memoria.
3. Intérprete de comandos: interface entre usuario y SO: línea de comandos. 

**Tipos**

Quizás conoce o ha escuchado sobre sistemas operativos para estaciones de trabajo convencionales, o quizás en estos momentos esté trabajando con alguno de ellos en su computadora: GNU/Linux, Windows, MacOS, Unix.

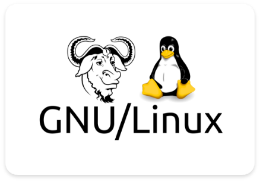
1. GNU/Linux cuenta con distribuciones y versiones como Ubuntu (uno de los más usados) y Fedora, entre otros.
2. Windows, por ejemplo, tiene Windows 7, Windows 10, Windows 11.
3. Mac OS presenta Monterrey, Big Sur, Sierra.

Existen otros sistemas operativos, pero los mencionados son los más populares, conocidos y usados.

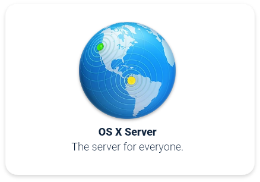
También existen versiones exclusivas de estos sistemas operativos para servidores y redes, pero con características especiales para trabajar en red. Estas versiones son:



1. **Windows:** Propiedad de la compañía Microsoft Corporation. Entre las distribuciones y versiones más conocidas y utilizadas en el mercado actual, se encuentra Microsoft Windows Server, que surgió a partir del año 2003; normalmente, cada año se publica una versión, y su nombre está ligado con el año de distribución como, por ejemplo, Windows Server 2012. Las versiones de Windows Server son de pago.



1. **GNU/Linux:** GNU es un sistema operativo de *software* libre, es decir, respeta la libertad de los usuarios. Linux es el Kernel o núcleo de GNU y la combinación de ambos, el sistema operativo GNU/Linux. Existen versiones para servidores “gratuitos”: Ubuntu Server, CentOS, Fedora, y otras de pago como RedHat Enterprise Server.



1. **Mac OS X Server:** Son los menos usados en el mercado para propósito de servidores.

El sistema operativo puede manipular los componentes de un equipo de cómputo, como: administración de recursos, acceso a dispositivos de entrada y salida, servidor la interfaz entre sistema y usuario.

A su vez un SO está conformado por varios programas, y sus componentes principales son la gestión de procesos, entradas y salidas, memoria, sistema de archivos.

Una vez se tiene el sistema operativo instalado en un dispositivo, al hardware en particular o un software especial para que pueda ser manipulado; es lo que se llama drivers. Muchas veces se conecta algún periférico: impresora, cámara, lector de huella, etc., pero el computador no es capaz de reconocerlos, surge entonces la necesidad de instalar dichos drivers o controladores para que se puedan realizar los trabajos con este periférico de manera adecuada. Los sistemas operativos modernos ya poseen la mayoría de los drivers para los dispositivos periféricos comunes comerciales, simplemente se conecta y, de manera casi inmediata, es reconocido.

**Infraestructura**

Los servidores físicos son equipos especiales, normalmente cuentan con recursos físicos mayores a un equipo de cómputo convencional de uso personal.

**Normalmente se les instala un sistema operativo para servidor y poseen los siguientes recursos:**

1. **Memoria RAM:** Es un dispositivo de hardware, normalmente en forma de tarjetas modulares que se insertan en la placa base o Motherboard. Se encarga del almacenamiento e intercambio de información de manera temporal. Mientras trabaja con su equipo de cómputo estos datos son volátiles y suelen borrarse cuando se apaga la computadora. Tiene capacidades de almacenamiento en bytes y las memorias modernas vienen con capacidades hasta de 128 GB y 256 GB, con frecuencia de intercambio de información hasta 5 200 MHz, tanto para computadores personales como para servidores.
2. **CPU:** Es la unidad de almacenamiento central o procesador, llamado muchas veces como el “cerebro” de la computadora porque es el que lleva a cabo las principales funciones de gestión y cálculos que se realizan en esta.
3. **Disco duro o rígido:** Es un dispositivo donde se almacenan y persisten los archivos de la computadora o servidor; en él se instalan el sistema operativo y otras informaciones importantes. Aún existen los discos duros de lectura convencionales, de plato, discos duros de estado sólido y discos M2.
4. **Tarjeta de red:** Es necesaria para la comunicación por cable, o inalámbrica, entre una o más estaciones de trabajo.

La infraestructura convencional para una empresa, en lo que se refiere a redes y servidores, anteriormente estaba conformada por equipos físicos armados como dispositivos de redes (switch, routers, cableado), servidores físicos, racks, UPS, termostatos, sensores de humo, temblor, etc.; organizados y almacenados en un cuarto climatizado y “asegurado”. La tendencia en la actualidad para estas infraestructuras es la basada en la nube o cloud, más conocida como infraestructura como servicio (IaaS).

Los proveedores más conocidos en la actualidad para estos servicios son los gigantes de:



Amazon (Amazon Web Service)



Microsoft (Azure)



Google (Google Cloud)

También se puede optar por una infraestructura híbrida, es decir, en donde se incluya infraestructura convencional e infraestructura en la nube. Las pequeñas, medianas y grandes empresas pueden construir su propia infraestructura de acuerdo con sus necesidades, ventajas y desventajas, costo / beneficio, y dependiendo de las variables tiempo y cantidad de clientes, entre otras.

# Modelos de arquitectura



La mayoría de los conceptos e ideas de arquitectura de software, provienen de arquitectura de edificios. Quizá no somos arquitectos, pero tenemos la idea que, un arquitecto, antes de llevar a cabo sus proyectos, diseña los planos de sus casas o edificios y presenta sus estilos arquitectónicos; presenta unos tiempos y unos materiales necesarios para la construcción.

Existen estilos arquitectónicos propuestos y llevados a cabo por autores que han sido probados en grandes organizaciones y han funcionado con éxito, y por eso se usan aún; algunos no, o llegan nuevos que los sustituyen. Incluso, el arquitecto puede crear su propio estilo o patrón.

Arquitectura de software es una definición algo relativa. Se podría decir que no hay concepto exacto de ello, de hecho, puede ser complejo. A continuación, presentamos una definición que parece bastante acertada y comprensible:

La arquitectura del software de un programa o sistema informático, se refiere a la estructura o estructuras del sistema, que se fundamentan en componentes de software; las propiedades de esos componentes son visibles al exterior, y a sus relaciones.

**Existen tres conceptos que tienden a confundirse:**

1. Patrón arquitectónico.
2. Patrón de diseño.
3. Estilo arquitectónico.

Un estilo arquitectónico, tal como lo hablamos en la comparación con edificios, es el cómo está construido: la forma como está hecho, la manera como se construye. Con respecto a los patrones arquitectónicos un estilo puede tener uno o varios. Entonces, un estilo arquitectónico de software se refiere a ese conjunto de decisiones para construir ese software en determinado contexto.

Es entonces que, cuando vamos a construir un software, no debemos ir al grano sin antes analizar bien y responder las siguientes preguntas:

1. ¿Qué tipo de software o aplicación vamos a construir?
2. ¿Para quién la vamos a construir?
3. ¿Para cuándo debe estar listo?

En la fase de la arquitectura, donde vamos a representar el sistema con un alto nivel de abstracción, debemos tener claros todos los requerimientos del sistema. Esta fase se realiza antes de la fase de diseño del Ciclo de vida de desarrollo del software.

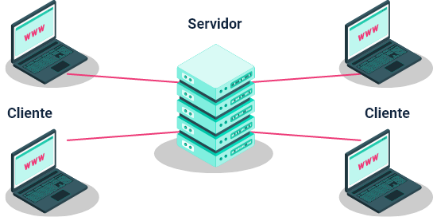
Un patrón en general es algo que se sigue para solucionar un problema en particular; entonces, un patrón arquitectónico brinda soluciones a problemas en desarrollo de software, pero debemos ver los patrones arquitectónicos como una abstracción. Muchas veces se suelen mirar como ilustraciones particulares o líneas de código de un lenguaje de programación que solucionan algo, pero esto es un concepto errado, al igual que para los patrones de diseño.

Los patrones y estilos arquitectónicos nos van a permitir organizar los componentes del software final que se requiere. También aportan a sus atributos de calidad, que resultan de los requerimientos no funcionales del sistema.

La arquitectura puede ilustrarse, representarse o describirse con lenguajes y estándares basados en el Lenguaje de Modelado Unificado (UML).

Los estilos arquitectónicos se seleccionan dependiendo del tipo de aplicación a construir. Pueden ser:

1. **Cliente-Servidor:** Se cataloga como un estilo arquitectónico; en esta arquitectura están basadas la mayoría de las aplicaciones que usan internet. Como su nombre lo dice, tiene dos componentes: cliente y servidor, además del medio de comunicación entre estos.



A su vez, esta arquitectura tiene varios tipos: 2-tier (2 niveles o capas).

En esta arquitectura el cliente y el servidor se comunican entre sí sin ningún punto o nodo intermedio; aunque proporciona un servicio rápido, puede ocasionar agujeros de seguridad y problemas de rendimiento. Ejemplo: Internet Explorer utiliza una arquitectura de servidor de cliente de 2 niveles, 3-tier (3 niveles o capas); en esta, un nodo más ―llamado nivel medio― se encuentra entre el cliente y el servidor. El nivel medio toma la solicitud del cliente, lo autentica, aprueba y luego lo pasa al servidor. También se habla de n-tier (multinivel).

1. **Arquitectura por capas (Layered):** También es catalogada como un estilo arquitectónico. En ejemplos de arquitectura por capas están el modelo OSI y el modelo TCP/IP, en donde hay una interacción entre los componentes de una misma capa o con las capas adyacentes. Todas trabajan por una tarea en común.



Esta arquitectura agrupa las responsabilidades del software en varias capas débilmente acopladas. Las capas pueden comunicarse entre sí a través de interfaces bien definidas. Dado que las capas están débilmente acopladas, pueden residir en una sola computadora o pueden estar distribuidas en varios ordenadores.

**Por lo general un sistema de software se divide en tres capas:**

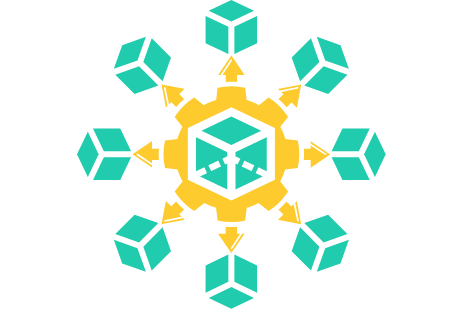
1. **Capa de presentación:** Esta capa incorpora funcionalidades de la interfaz de usuario. A través de esta capa el usuario puede interactuar con el sistema.
2. **Capa de negocios:** Esta capa incorpora la lógica de negocio del sistema.
3. **Capa de infraestructura:** Esta capa incorpora servicios de datos, servicios de red y otros servicios de infraestructura.

Observación: cuando se llevan al español los términos tier y layer se tienden a confundir, por eso es mejor hablar de tier como nivel y de layer como capa. Tier para referirse a lo físico, layer como parte del código o parte lógica.

1. **Arquitectura orientada a servicios (SOA)**

Es un estilo arquitectónico para sistemas distribuidos. Al igual que la arquitectura cliente – servidor, SOA, examina cómo construir, usar y combinar servicios. En lugar de crear grandes paquetes de software que lo hagan todo…divide y vencerás. SOA alcanza los objetivos del software mediante la creación y el uso de servicios y el diseño de una arquitectura que admita su uso.



1. **Microservicios:** Las pioneras de esta arquitectura fueron importantes plataformas de entretenimiento, servicio de streaming y de comercio electrónico. Fue tan bueno el resultado, que otras empresas decidieron adoptar el modelo y seguirlo. Tales plataformas se consideran también un estilo arquitectónico consistente en que las aplicaciones se dividen en sus elementos más pequeños e independientes entre sí. A diferencia del enfoque tradicional y monolítico de las aplicaciones, en el que todo se compila en una sola pieza, los microservicios son elementos independientes que funcionan en conjunto para llevar a cabo las mismas tareas. Cada uno de esos elementos o procesos es un microservicio.
2. **El Modelo Vista Controlador (MVC):** Según el concepto de patrón arquitectónico, el MVC se puede considerar así, un patrón de arquitectura, que, de hecho, es bastante común en el diseño de software. Cabe resaltar que otros autores lo pueden considerar como una estructura, y otros, de manera errada, mencionan que es un patrón de diseño.

La idea de este patrón es separar la vista (interfaz de usuario) con la lógica de negocios y los datos:

1. **Modelo:** mantiene el estado de una aplicación.
2. **Vista:** muestra la interfaz de usuario a dicho usuario.
3. **Controlador:** procesa las acciones del usuario (enviadas desde la vista) para actualizar el modelo y reenviar esas actualizaciones a la vista.

Con MVC puede lograr el diseño de buenas prácticas de código, creando estructuras de namespaces, paquetes o directorios para tener un proyecto organizado a nivel de código que puede ser más fácil de mantener y entender.

Un patrón arquitectónico es solo una abstracción. El MVC, por ejemplo, se puede ver o comparar con un sinnúmero de actividades o sucesos de la vida real. Imagínese llegar a una cafetería, sentarse en la terraza y pedir una bebida (usted sería la vista) a la persona que atiende (controlador), esta persona va a la cocina y les dice a los encargados (modelo) que preparen su pedido; ellos lo preparan, lo entregan al mesero, y el mesero lleva su pedido y usted lo recibe.

# Lenguajes de programación y stack tecnológico

Un lenguaje de programación es aquel lenguaje computacional que sirve para crear software en general, sean programas de consola, de escritorio, orientados a la web, móviles, dispositivos embebidos; incluso, con un lenguaje de programación se pueden crear otros lenguajes de programación. La creación de este software se realiza mediante algoritmos o instrucciones dadas, propias de cada lenguaje (sintaxis) y una lógica.

Según su ejecución se pueden encontrar tres clases de lenguaje de programación: compilados, interpretados e híbridos (compilados e interpretados). Algunos autores solo consideran los compilados e interpretados.



Los lenguajes de programación compilados son aquellos que necesitan de un traductor o compilador para ejecutarse, o sea, necesitan un compilador que convierte el código escrito en lenguaje, que es legible por el ser humano, a lenguaje máquina; ejemplo: C, C++.

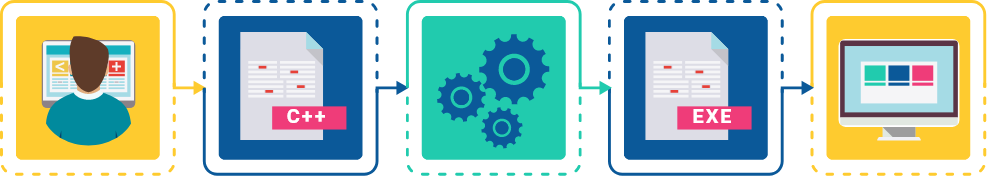


Un lenguaje interpretado es aquel que se ejecuta sin que se genere código - objeto; por ejemplo: PHP, Python, JavaScript.

Por otro lado, los híbridos, son los que se encuentran en un estado entre compilado e interpretado; Java es uno de ellos, aunque algunos autores categorizan a Java como un lenguaje compilado.

Según su finalidad y nivel de abstracción los lenguajes de programación pueden ser de tipo máquina, assembler, bajo nivel y alto nivel.

**¿Cómo funciona el proceso de compilación?**



1. Abre un programa IDE (Entorno integrado de desarrollo) y escribe su código en un lenguaje de programación de alto nivel, el cual es legible y entendible por usted.
2. Compila su programa. Aquí ese código legible se convierte a código máquina, el código entendible por la CPU de su computadora: unos y ceros.
3. La máquina entiende ahora ese código, se puede ejecutar en ella y le muestra los resultados de su programa.

Existe un sinnúmero de lenguajes de programación, pero mencionaremos los más populares y demandados por la industria hoy día, aunque esto es relativo porque su uso varía con respecto a regiones geográficas. Por otra parte, hablando en términos de remuneración económica, unos son mejor pagos que otros.

Uno de los más populares es Python, y una de las razones principales de su popularidad es el crecimiento de la ciencia de datos en los últimos años; otras razones: la simplicidad, conjunto de librerías para distintos propósitos y facilidad de programación.

A nivel mundial en la industria se trabaja mucho con Java y JavaScript, pero se trabaja fuertemente usando librerías y marcos de trabajo (frameworks), temas que se tomarán más adelante. PHP sigue siendo popular porque la mayoría de las páginas y aplicaciones web se crearon en este lenguaje y aún se siguen creando páginas con gestores de contenido como Wordpress, Joomla, Drupal, entre otros.

Otro lenguaje de programación que ha tomado fuerza en los últimos años es C y C++. Se puede afirmar que ha sido por su eficiencia con el hardware y el crecimiento del Internet de las cosas, donde se puede resaltar el desarrollo de videojuegos.

Existen otros lenguajes de programación tales como Ruby, Dart, Kotlin, Swift, C, TypeScript, Go, Rust, SQL (para bases de datos relacionales), Scala, R, Perl, Shell Script, etc.

En la siguiente tabla encontrará una lista de algunos lenguajes de programación y su uso, propósito o plataformas en las que puede construir software con ellos.

1. Lenguajes de programación

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Tipo de aplicación o software (videojuegos, utilidades, empresarial) |  |  |  |  |
| **Lenguaje de programación** | **Escritorio (videojuegos, utilidades, empresarial)** | **Web** | **Móvil** | **Embebido** | **Consola** |
| Java | X | X | X | X | X |
| PHP |  | X |  | X | X |
| Python | X | X | X |  | X |
| JavaScript | X | X | X |  | X |
| TypeScript | X | X | X |  | X |
| C | X |  |  | X | X |
| C++ |  |  |  | X | X |
| C# | X | X | X | X | X |

Hoy en día existen librerías y frameworks para hacer posible que, por ejemplo, lenguajes de programación como PHP construyan aplicaciones de escritorio y móvil, pero no es su propósito principal, y su performance puede no ser el mejor.

**Stack tecnológico**

Stack, en español, traduce pila. Entonces, se puede mirar un stack tecnológico como una pila de tecnologías y cómo ellas se entremezclan como una sola, para construir aplicaciones, según las necesidades que se presenten.

Antes de mencionar algunas miraremos qué son un framework (marco de trabajo) y una librería, para que pueda establecer la diferencia con un lenguaje de programación; estos tres términos suelen confundirse muy comúnmente cuando se está iniciando en este mundo. Existe un término más, “entorno de ejecución”, el cual es relativo y no tiene una definición exacta.

Con un lenguaje de programación se pueden crear “librerías”, esto normalmente surge cuando se están haciendo una o varias aplicaciones software distintas y se necesita resolver una tarea en común y evitar el trabajo de reescribir código una y otra vez para lo mismo; entonces se crea una librería y se “importa” o se usa cada vez que se necesite hacer dicha tarea.

Un framework es un conjunto de librerías ―patrones de diseño, patrones arquitectónicos― con el que se pueden trabajar, en muchas ocasiones, “buenas prácticas” de código y organización. En consecuencia, un framework puede contener varias librerías, y una librería no contiene frameworks.

A continuación, se mencionan algunos frameworks y los lenguajes de programación que usan: Spring (Java), Laravel (PHP), Express (NodeJS/Javascript), Flask y Django (Python), Ionic (TypeScript, Javascript), Flutter (Dart).

Entre los stack más comunes en la actualidad se pueden encontrar:

1. **MERN:** Estas son las siglas de MongoDB, Express.js, React JS, Node JS. MongoDB es un motor de base de datos no relacional. Express es un framework que usa el entorno de ejecución de Node JS que se usa para trabajar BackEnd. React JS es una librería que originalmente empezó usando el lenguaje de programación Javascript para construcción de interfaces gráficas de usuario (FrontEnd).
2. **MEAN:** Son siglas de un stack tecnológico, al igual que MERN, pero a diferencia de este, la A se refiere al framework de Angular, que principalmente hace el uso del lenguaje de programación TypeScript.
3. **LAMP:** Son las siglas de Linux, Apache, MySQL, PHP. Linux es un sistema operativo que serviría de plataforma para la instalación del servidor de aplicaciones Apache, y MySQL es un motor de base de datos.
4. **.NET:** Es un ecosistema o framework, propiedad de Microsoft para la construcción de aplicaciones; usa lenguajes de programación como Visual basic o C#, entre otros.
5. **Ruby On Rails:** Es un framework para construir aplicaciones full stack con el lenguaje de programación Ruby.

# Gestor de base de datos y servidores

Un ejemplo claro de base de datos es una biblioteca, allí se encuentra la información clasificada y organizada. Otro ejemplo son los libros contables en papel ―no sistematizados― de los pequeños negocios, incluso hojas de Excel donde alguien lleve sus cuentas.

Hoy en día, los sistemas de información de cualquier tamaño poseen una base de datos, así sea almacenada en archivos planos de texto. Anteriormente, en la mayoría de las aplicaciones persistían los datos en archivos o ficheros. Luego, se optó por usar motores de bases de datos por todas las ventajas que poseen, tales como: la velocidad de acceso a la información, el fácil mantenimiento, la consistencia de los datos, etc. En sus aplicaciones web será indispensable que cree bases de datos y las integre con estas.

Es importante aclarar y saber diferenciar entre:

1. **Gestor de base de datos:** Es aquel software especializado para realizar conexión con un motor de base de datos y todo tipo de operaciones: gestión de usuarios, bases de datos y sentencias con el motor como consultas, creación, borrados y actualizaciones que suelen ser de forma “gráfica” o “consola”. Ejemplo de estos: DBEaver, MySQL Workbench, PgMyAdmin, PHPMyAdmin, SQL Server.
2. **Sistema gestor de base de datos:** Se refiere a todo el conjunto que entra en juego con la base de datos: los usuarios que interactúan y las herramientas necesarias o software para trabajar con las bases de datos, motores, incluso el gestor de base de datos y la infraestructura asociada.
3. **Motor de base de datos:** Es un software para administración de bases de datos.

En la actualidad, a nivel empresarial se trabaja con motores de bases de datos tanto relacionales como no relacionales.

1. **Relacionales:** Se pueden mencionar algunos como MySQL (es uno de los más populares), PostgreSQL, Oracle, Microsoft SQL Server.
2. **No relacionales:** Entre ellos MongoDB, Cassandra, Redis.



El lenguaje de programación para los motores de bases de datos relacionales mencionados es SQL. Los no relacionales almacenan formato JSON o BSon.

SQLtiene 2 tipos de verbos-instrucciones:



DDL

DDL (Lenguaje de definición de datos): CREATE: definición.



DML

(Lenguaje de manipulación de datos): SELECT, realizar consultas; INSERT, insertar registros; UPDATE, actualizar registros; DELETE, borrar registros.

Entre las bases de datos relacionales y no relacionales (No-SQL) hay ciertas diferencias respecto a sus elementos, incluso sus propósitos. Pero se mencionan en la siguiente tabla algunas relacionadas con estructuras y usos.

1. Diferencias bases de datos relacionales y no relacionales

|  |  |  |
| --- | --- | --- |
| Tipo | SQL | NoSQL |
|  | Base datos | Base de datos |
|  | Tabla | Colección |
|  | Fila | Documento |
|  | Columna | Campo |
|  | Llave primaria | Llave primaria |
| Usos | Web, móvil, empresarial, datamart. | Videojuegos, redes sociales, IoT, web, móvil, empresarial |

Para ampliar la información, lo invitamos a leer en el material complementario, los siguientes artículos:

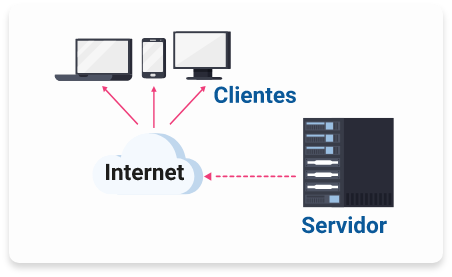
1. Utilidad y funcionamiento de las bases de datos NoSQL
2. Un estudio comparativo de Bases de Datos Relacionales y Bases de Datos NoSQL

**Servidores**

Los servidores son los que prestan el servicio requerido por un cliente.

Algunos tipos de servidores son:

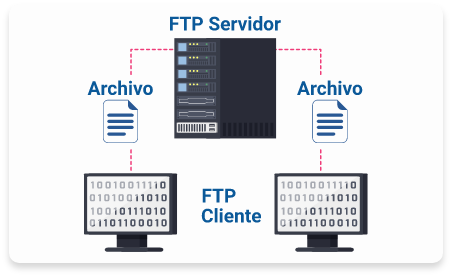
**Servidores web**



Es un software especial para el procesamiento de peticiones en doble dirección cuando un cliente le solicita. También se le llama servidores HTTP, servidores que deben contar con un hardware. Hoy en día se habla de servidores web en la nube, pero al final estos también necesitan un hardware para funcionar.

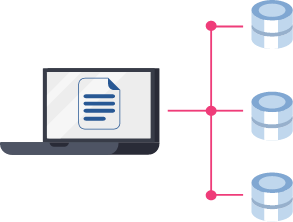
En este tipo de servidores, normalmente se alojan las páginas web que se consultan día a día, lo mismo que aplicaciones de compra, plataformas streaming, de música, etc., a las cuales se accede mediante el navegador.

**Servidores de FTP**



Las siglas FTP significan “Protocolo de transferencia de archivos”. Son servidores para el propósito de transferir archivos por internet. Funcionan cuando un cliente realiza la petición para enviar un archivo a ese servidor o se pide transferir dicho archivo desde ese servidor al cliente. Para realizar dicha transferencia debe contar con los respectivos permisos, sean: escritura, lectura o ambos. Un ejemplo de servidor FTP es Filezilla Server, que también tiene una versión cliente: Filezilla Client.

**Servidores de bases de datos**



Son aquellos en los que se alojan motores de bases de datos, relacionales o no relacionales. Incluso, pueden ser simples archivos que sean la base de datos de una aplicación.

**Ampliemos la información sobre los servidores web.**

Anteriormente se hablaba de “sistemas operativos para red”, los cuales se instalan en servidores físicos utilizados para distintos propósitos, entre ellos el de “servir” para la comunicación web. Se podría afirmar que al referirnos a un servidor web se está hablando de un servidor físico con un sistema operativo especial para red o un software como servidor, el cual se menciona a continuación:

Pero antes de hacer mención veamos qué diferencia hay entre un servidor web y un contenedor de aplicaciones. Un contenedor de aplicaciones es un software que puede alojarse dentro de un servidor web.

[De click aquí para ir al enlace de Apache Tomcat](https://tomcat.apache.org/)

Algunos servidores web bastante demandados en el mercado son:

1. **Apache**



Servidor HTTP de código abierto; se puede instalar sobre distintas plataformas como Linux, Window, MacOS. El puerto HTTP por defecto de Apache Server es el 80 y el puerto seguro es el 443.

1. **Microsoft IIS**



Propiedad de Microsoft. Se pueden hacer despliegues de varios tipos de aplicaciones realizadas en PHP, JS, etc., pero es especial para aplicaciones construidas con el Framework .NET.

1. **Nginx**



Es un servidor web/proxy. Muchos han migrado a él porque han encontrado mejoramiento en su rendimiento. También es de código abierto.

1. **GWS**



El gigante de Google también tiene su servidor, usado para su infraestructura concerniente a la web.

# Protocolos de comunicación

Los protocolos son reglas que se deben seguir para tener un estándar de comunicaciones en las redes de datos y entre los clientes y los servidores.

Una hipotética situación en que cada uno cree su propia forma de comunicar a los clientes con los servidores se volvería un caos total, es cuando una autoridad llega y estandariza normas para que se cumplan en dichas comunicaciones. De esta manera habrá un número considerable y definido para cada tarea.

“El servicio TCP se obtiene al hacer que tanto el servidor como el cliente creen puntos terminales, llamados sockets. Cada socket tiene un número (dirección), que consiste en la dirección IP del host, y un número de 16 bits, que es local a ese host, llamado puerto [sic]”. (Hernández, G., s.f.)

Algunos tipos de puertos son:

1. **HTTP (Hyper Text Transfer Protocol):** Protocolo de transferencia utilizada en el World Wide Web, especifica cuáles mensajes pueden enviar los clientes a los servidores y qué respuestas obtienen. HTTP soporta distintos métodos que se relacionan en la siguiente tabla:

|  |  |
| --- | --- |
| Método | Uso / descripción |
| Get | Petición de lectura. |
| Head | Petición lectura de encabezado. |
| Put | Petición almacenamiento para edición. |
| Patch | Igual a put, pero para parte del recurso a editar. |
| Post | Petición de guardado de recursos. |
| Delete | Petición para eliminar recursos. |
| Trace | Repite solicitud entrante. |
| Connect | Reservado para uso futuro. |
| Options | Consulta ciertas opciones. |

1. **FTP (File Transfer Protocol):** Los servidores FTP casualmente son los que trabajan con el protocolo de transferencia de archivos. El puerto de conexión por defecto de FTP es el 21.
2. **XMPP (Extensible Messaging and Presence Protocol):** Estándar abierto para mensajería y presencia.
3. **SMTP (Simple Mail Transfer Protocol):** En Internet, el correo electrónico se entrega al hacer que la máquina de origen, establezca una conexión TCP con el puerto 25 de la máquina de destino. Escuchando en este puerto se encuentra un dominio de correo electrónico que habla con el SMTP (Protocolo Simple de Transporte de Correo); este demonio acepta conexiones de entrada y copia mensajes de ellas a los buzones adecuados. Si no puede entregarse un mensaje, se devuelve al remitente un informe de error, que contiene la primera parte del mensaje que no fue posible de entregar.

Para ampliar la información consulte los siguientes enlaces:

[Métodos de Petición HTTP.](https://developer.mozilla.org/es/docs/Web/HTTP/Methods)

[Códigos de estado de respuesta HTTP.](https://developer.mozilla.org/es/docs/Web/HTTP/Status)

**Instalación herramientas de trabajo**

Para empezar a desarrollar aplicaciones web, o de otro tipo, es necesario contar con la configuración de un ambiente de desarrollo a nivel de producción o local, y es importante contar con las distintas herramientas que se han mencionado en el presente componente formativo.

En el siguiente video se explicará cómo realizar la instalación de algunas de estas herramientas recomendadas de trabajo.

1. Instalación herramientas de trabajo



**Enlace de reproducción del video**

|  |
| --- |
| **Síntesis del video: Instalación herramientas de trabajo** |
| En este video tutorial vamos a realizar la instalación de algunas herramientas recomendadas para el programa y que a futuro le servirán como profesional. Son herramientas libres que se pueden descargar desde su web oficial.  Las herramientas que se muestran en el video son:  DBeaver: La versión DBeaver Community y la versión DBeaver PRO.  Apache NetBeans  Visual Studio Code  XAMPP |

Recuerde explorar los demás recursos que se encuentran disponibles en este componente formativo; para ello, diríjase al menú principal, donde encontrará la síntesis, una actividad didáctica para reforzar los conceptos estudiados, material complementario, entre otros.

Síntesis

A continuación, encontrará una síntesis de las temáticas trabajadas en el componente formativo.

Arquitectura de software.
1. networking: dispositivos de red, medios de comunicación y servicios. 
2. Servicios operativos de redes.
3. Modelos de arquitectura: cliente - servidos, arquitectura por capas, arquitectura orientada, microservicios y modelo vista controlador. 
4. Lenguajes de programación.
5. Stack tecnológico.
6. Gestor de base de datos.
7. Servidores.
8. Protocolos de comunicación.


Material complementario

|  |  |  |  |
| --- | --- | --- | --- |
| Tema | Referencia APA del Material | Tipo de material | Enlace del Recurso o  Archivo del documento material |
| Gestor de base de datos y servidores | Castro Romero, A., González Sanabria, J. S. & Callejas Cuervo, M. (2012). Utilidad y funcionamiento de las bases de datos NoSQL. Revista Facultad de Ingeniería, 21(33). p. 21-32 | PDF | <https://www.redalyc.org/articulo.oa?id=413940772003> |
| Lenguajes de programación y stack tecnológico | nicobytes. (2020). ¿Cuál es el mejor stack tecnológico para una aplicación? (video). YouTube. | Video | <https://www.youtube.com/watch?v=16CauMlYygs> |
| Modelos de arquitectura | hdeleon.net. (2020). Modelo de Arquitectura de Software C4 – Ejemplo práctico sencillo (video). YouTube. | Video | <https://www.youtube.com/watch?v=2IJ29e0xTDc> |
| Gestor de base de datos y servidores | Moreno, L., Olsowy, V., Thomas, P., Delia, L., Tesone, F., Fernández, J. & Pesado, P. (2019). Un estudio comparativo de bases de datos relaciónales y bases de datos NoSQL. | PDF | <http://sedici.unlp.edu.ar/handle/10915/91403> |

Glosario

**Ciclo de vida del desarrollo de software:** estructura que contiene los procesos, actividades y tareas relacionadas con el desarrollo y mantenimiento de un producto de software, que abarca la vida completa del sistema, desde la definición de los requisitos, hasta la finalización de su uso. (https://ungoti.com/es/soluciones/desarrollo-de-software)

**Lenguaje de programación:** es aquel lenguaje computacional que sirve para crear software en general, sean programas de consola, de escritorio, orientados a la web, móviles, dispositivos embebidos.

**Medios de comunicación:** son los canales por donde viaja la información como cables UTP, de fibra óptica o wifi.

**Networking:** redes de datos para comunicación e intercambio de información entre dispositivos o terminales.

**Patrón de diseño:** soluciones habituales a problemas comunes en el diseño de software. Cada patrón es como un plano que se puede personalizar para resolver un problema de diseño particular de un código. (<https://refactoring.guru/es/design-patterns>)

**Servicios:** programas y aplicaciones que están a lo largo de la red de comunicaciones.

**Topología de redes:** forma como se interconectan los dispositivos de una red.

Referencias bibliográficas

Developer Mozilla. (s.f.). *Códigos de estado de respuesta HTTP*. <https://developer.mozilla.org/es/docs/Web/HTTP/Status>

Garlan, D. (2001). *Sofware Architecture*. <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/able/ftp/encycSE2001/encyclopedia-dist.pdf>

GNU. 2022. *GNU*. <https://www.gnu.org/home.es.html>

Hernández, G. (s.f.). *Protocolo de control de transferencia (TCP).* Universidad Autónoma del Estado de Hidalgo. <https://www.uaeh.edu.mx/scige/boletin/huejutla/n3/r1.html#refe1>

Créditos

Elaborado por Innovative Education

