**Tabla de Contenidos: Cómputo en la Nube (80 horas)**

| **Módulo** | **Tema** | **Subtemas** | **Horas** |
| --- | --- | --- | --- |
| 1 | **Introducción al Cómputo en la Nube** | - Historia y evolución - Conceptos clave: virtualización, escalabilidad, elasticidad - Modelos de servicio: IaaS, PaaS, SaaS - Modelos de despliegue: pública, privada, híbrida, comunitaria | 8 |
| 2 | **Servicios en la Nube** | - Servicios de cómputo (VMs, contenedores) - Almacenamiento (objetos, bloques, archivos) - Bases de datos (SQL, NoSQL) - Redes y balanceo de carga - Servicios gestionados | 12 |
| 3 | **Plataformas de Cómputo en la Nube** | - Introducción a AWS, Azure y Google Cloud - Comparación de servicios equivalentes - Creación de cuentas gratuitas - Despliegue básico de aplicaciones | 12 |
| 4 | **Seguridad en la Nube** | - Principios de seguridad en la nube - Autenticación y autorización (IAM) - Cifrado de datos en tránsito y en reposo - Cumplimiento y normativas (ISO, GDPR, etc.) - Buenas prácticas y casos de estudio | 10 |
| 5 | **Herramientas y Tecnologías de Administración** | - Consolas de administración y CLI - Automatización con scripts y SDKs - Infraestructura como código (IaC) con Terraform o CloudFormation - Monitoreo y logging (CloudWatch, Stackdriver, etc.) - Gestión de costos y presupuestos | 14 |
| 6 | **Laboratorios y Proyectos Prácticos** | - Despliegue de una app web en la nube - Configuración de una base de datos escalable - Implementación de políticas de seguridad - Automatización de infraestructura - Presentación de proyectos finales | 24 |

**Modulo I: Temario Detallado**

* 1. **Definición y evolución histórica**

**¿Qué es el cómputo en la nube?**

En términos generales, se puede decir que el cómputo en la nube es un modelo que permite usar múltiples recursos bajo demanda con un mínimo de configuración por parte del proveedor de servicio. Gracias a este modelo, los usuarios finales pueden enfocarse en sus tareas sin tener la necesidad de mantener y/o administrar un centro de datos. Más aún, no se preocupan de estar al pendiente de los espacios físicos, costos energéticos y demás sistemas que se requieren para administrar un centro de datos. De esta manera, vía Internet los recursos de cómputo están disponibles, sin la necesidad de gestionar la infraestructura y sin la responsabilidad que conlleva la seguridad y privacidad de la información.

**- Historia: desde mainframes hasta la nube moderna**

Es un concepto que tiene sus raíces en los años sesenta; está basado en las ideas de John McCarthy (profesor emérito de la Universidad de Stanford y cofundador del Laboratorio de Inteligencia artificial del MIT), que predijo, en 1961, durante un discurso para el centenario del MIT, que la tecnología de tiempo compartido (que estaba en auge en aquel momento) de las computadoras podría conducir a un futuro en el que el poder del cómputo e incluso aplicaciones específicas podrían ser vendidas como un servicio.

Joseph Carl Robnett Licklider (conocido como J.C.R. o simplemente Lick), que ha sido una figura muy importante en el ámbito de la ciencia computacional y recordado, particularmente, por ser uno de los primeros que imaginó la computación interactiva moderna y su aplicación a diferentes actividades con una visión de la interconexión global de ordenadores (mucho antes que fuera construida).

J.C.R. trabajó y aportó financiación a proyectos como Arpanet (predecesor de internet) o la interfaz gráfica de usuario que permitieron el acceso a servicios y recursos a personas no especialistas; ideas que, en las opiniones de expertos actuales, coinciden en que se asemejan mucho a la idea del cloud computing tal y como lo conocemos hoy en día.

En cuanto a los desarrollos más vinculados al cloud actual, cabe destacar el desarrollo de la Web (Tim Berners-Lee propone las ideas en 1989, pero no es hasta 1990 cuando hay un prototipo del WorldWideWeb) y su evolución más reciente, Web 2.0, y el desarrollo de internet.

A partir de entonces comienzan a surgir una serie de servicios, básicamente aplicaciones centradas en búsqueda, correo y páginas como Wandex, en 1993, que acabó siendo el primer buscador, y luego Yahoo! en 1994, Altavista en 1995 o Hotmail en 1996, o más recientemente también en las búsquedas de Google en 1998.

Para algunos autores, uno de los grandes hitos vinculados al cloud es la llegada, en 1999, de Salesforce.com, que fue una de las primeras en ofrecer a sus clientes lo que hoy se llama SaaS (software as a service) y que era un software para la automatización de ventas a través de una simple página web.

En 2002, Amazon anuncia su infraestructura en la red y su ampliación en 2006 con el lanzamiento de Amazon Web Service, AWS como un servicio orientado a negocio que permitió a las pequeñas empresas y particulares usar equipos en los cuales se ejecutan sus propias aplicaciones y con un modelo de monetización basado en el «pago por uso».

En 2009 Google y otros grandes proveedores empezaron a ofrecer aplicaciones basadas en navegador popularizándose y ganando en seguridad y servicio.

También otros grandes proveedores implementan sus propias infraestructuras cloud; por ejemplo, Microsoft Azure fue anunciado en octubre de 2008 (pero no es hasta 2010 que comienza a prestar servicios), IBM en 2011 (lanza SmartCloud Framework para dar soporte a su proyecto Smarter Planet) u Oracle en 2012 (Oracle Cloud).

Se puede argumentar que en la actualidad es poco frecuente que una institución/empresa mediana o grande no tenga externalizados algunos servicios en el cloud para sus trabajadores/usuarios finales o no utilice el cloud para algunos de los aspectos vinculados al negocio de la misma (correo, web, intranet, etc.).

- Evolución de la virtualización y redes

**1.2 Características clave**

De forma descriptiva podemos enunciar las siguientes características clave para la computación en nube (orden alfabético) [Dcc11]:

1. **Agilidad y autoservicio**: rapidez y capacidad de proveer recursos (de forma casi inmediata) sin grandes intervenciones ni acciones por parte de las dos partes (esto se logra teniendo un flujo de trabajo predefinido muy bien ajustado y automatizando la provisión de recursos). Es decir, el usuario puede, en forma no asistida, aprovisionar capacidades de cómputo/almacenamiento/redes según le sea necesario de forma automática sin necesidad de interacción humana con el proveedor de servicios. Estos recursos estarán disponibles, en un intervalo corto de tiempo (segundos o a lo sumo unos pocos minutos), a través de la red.
2. **Costo**: dada la eficiencia y automatización en la gestión y administración de recursos, los precios resultantes para los proveedores de cloud son reducidos y pueden trasladar estos al usuario final; el usuario final no debe hacer frente a los gastos derivados de la implantación de infraestructura civil/servicios y a la inversión en máquinas, software y recursos humanos, por lo cual contabilizando todo el modelo sale favorable en relación con el cloud. Además, como el control y la monitorización son automáticos, se pueden realizar medidas con un cierto nivel de abstracción apropiado para el tipo de servicio (por ejemplo, cuentas/cuentas activas, almacenamiento, procesamiento o ancho de banda); es posible ajustar el costo a modelos de, por ejemplo, pago-por-uso, pago-porpeticiones, etc. proporcionando transparencia, tanto para el proveedor como para el consumidor del servicio utilizado.
3. **Escalabilidad y elasticidad**: estos conceptos se refieren al aprovisionamiento de recursos en (casi) tiempo real y adaptabilidad a las necesidades de carga/uso de los mismos. Es decir, si se puede prever la carga/utilización no es necesario aprovisionar todos los recursos desde el inicio como si de una inversión local se tratara, sino planificarlos para cuando la demanda lo requiera con la consiguiente reducción del costo.
4. **Independencia entre el dispositivo y la ubicación**: independizar el recurso del acceso y facilitar que estos puedan ser desde una red local, corporativa u otro tipo y desde diferentes dispositivos (capacidad/tipología).
5. **Mantenimiento y licencias**: el modelo de actualizaciones y licencias se simplifica ya que el software se encontrará en el o los servidores del cloud y no existirá nada instalado en el dispositivo del usuario final.
6. **Rendimiento y gestión de recursos**: control exhaustivo y monitorización eficiente de los servicios para lograr una alta disponibilidad y utilización óptima de los recursos de forma automática. Estas características permiten reducir al máximo las ineficiencias aportando control y notificación inmediata, así como transparencia y seguimiento tanto al proveedor como al usuario final. Además, como los recursos se pueden configurar para servir a múltiples usuarios en un modelo de tenencia múltiple, los recursos físicos y virtuales pueden ser asignados dinámicamente y reasignados de acuerdo con la demanda de los consumidores. Esto proporciona un sentido de independencia de la ubicación y el cliente no tendrá ningún control o conocimiento sobre la ubicación exacta de los recursos proporcionados (solo a un nivel de abstracción muy alto, por ejemplo, país o centro de datos).
7. **Seguridad**: puede incrementarse dada la particularidad de tener los datos centralizados. Corresponderá al responsable de la aplicación la seguridad de la misma, pero al proveedor la responsabilidad de la seguridad física. Con ello, la seguridad será al menos igual que en los sistemas tradicionales, pero podrá ser mejorada (y deberá estipularse qué nivel se desea en la SLA –Service Level Agreement–) ya que el proveedor tiene interés en que su infraestructura sea segura para sus clientes como base de la calidad del negocio y podrá invertir en ella globalmente, mientras que esta inversión puede ser inaceptable para un cliente si debe hacer lo mismo sobre la infraestructura local.
8. **Virtualización como base para el aprovisionamiento**: esto permite compartir y optimizar el uso del hardware reduciendo los costos, la energía consumida/servidor y el espacio facilitando el despliegue rápido de servicios y garantizando la alta disponibilidad (servicios en stand by) y movilidad por carga (movimiento de máquinas virtuales a servidores más ociosos cuando la carga aumenta).

//De aquí lo pasamos a ventajas

**1.3 Modelos de despliegue**

**-** **Nube pública, privada, híbrida y comunitaria**

Existen diferentes taxonomías para describir los servicios y la forma de despliegue del cloud. En su documento The NIST Definition of Cloud Computing del NIST (National Institute of Standards and Technology), se definen cuatro modelos de despliegue y tres modelos de servicio. Como modelos de despliegue, podemos enumerar:

1. **Cloud público**: en este caso, el sistema es abierto para su uso general bajo diferentes modelos de negocio (desde pago por recursos, por uso, cuota o libre). Este tipo de recurso es mantenido y gestionado por un tercero y los datos y aplicaciones de los diferentes clientes comparten los servidores, sistemas de almacenamiento, redes y otras infraestructuras comunes; normalmente los recursos se utilizan y gestionan por medio de internet. Generalmente, un cliente no tiene conocimiento sobre con qué otros usuarios comparte la infraestructura y su utilización se contrata a través de un catálogo de recursos disponibles; su aprovisionamiento es automático (o semiautomático) después de haber cumplido con una serie de trámites en cuanto al modelo de pago y SLA. El propietario del cloud (proveedor de recursos y servicios) puede ser una empresa privada, una institución académica o una organización gubernamental, o una combinación de ellas, generalmente en función del tipo de clientes y recursos que se deberán proveer. Técnicamente puede haber pocas diferencias (o ninguna) con alguno de los otros modelos (especialmente con la privada); sin embargo, puede haber diferencias sustanciales en relación con la seguridad, que en el cloud público puede estar disponible en una red abierta como internet y sin mecanismos de cifrado.
2. **Cloud privado**: en este caso, la infraestructura funciona exclusivamente para una organización/empresa y es utilizada por sus unidades de negocio o departamentos; esta puede ser la propietaria, la administradora y la operadora, o algunos de estos apartados puede ser subcontratado a un tercero. Es la opción más favorable para las compañías que necesitan alta protección de datos y un alto nivel de servicio, ya que los recursos y su gestión se encuentran bajo el control y cuidado de la propia empresa/organización.
3. **Cloud híbrido**: aquí se combinan modelos públicos y privados; el propietario dispone de una parte privada (con acceso controlado y bajo su control) y comparte otras, aunque de una manera controlada. Las nubes híbridas ofrecen la posibilidad de escalar muy rápidamente con aprovisionamiento externo bajo demanda, pero implican una gran complejidad a la hora de decidir cómo se distribuyen los datos y las aplicaciones en una banda u otra. Si bien es una propuesta atractiva para las empresas, los casos habituales de uso no pasan de aplicaciones simples sin restricciones de sincronización con bases de datos sofisticadas.
4. **Cloud comunitario**: los servicios están diseñados para que puedan ser utilizados por una comunidad, organizaciones o empresas bajo el mismo alineamiento de objetivos o negocio (por ejemplo, bancos, distribuidores, arquitectos...) o que requieran unas características específicas (por ejemplo, seguridad y privacidad). Las empresas que forman la comunidad pueden ser los propietarios de infraestructura, así como los gestores u operadores; algunos de estos roles pueden ser subcontratados a terceros, pero bajo las indicaciones y reglas que aplica a la comunidad.

Diagrama

El contenido generado por IA puede ser incorrecto.

La figura siguiente muestra un esquema de estas cuatro formas de implantación y desarrollo de los clouds.

**- Ventajas y desventajas de cada modelo**

**1.4 Modelos de servicio**

Otra de las clasificaciones habituales es por el nivel de servicio que prestan:

**- IaaS (Infraestructura como Servicio): Ej. Amazon EC2**

En la **infraestructura como servicio (IaaS)** se encuentra la capa de recursos básica (generalmente máquinas virtuales, redes y almacenamiento) donde el cliente podrá poner sus SO y sus aplicaciones e implementar un servicio o utilizarlas para ejecutar sus aplicaciones. El cliente no podrá manejar o controlar el hardware subyacente, pero sí a partir del SO, almacenamiento y aplicaciones/servicios implementados sobre esas máquinas, así como algunos aspectos de la conectividad (subredes, firewalls, dominios...).

La reflexión del usuario en esta modalidad sería «¿por qué comprar, instalar y probar infraestructura cada N años (obsolescencia) y no alquilarla y pagar por uso?». Los recursos son obtenidos sin hacer obra civil (centro de datos), con mínimos recursos humanos (no especialistas en IT), sin gran inversión inicial, escalable y eficiente y a costos aceptables

**- PaaS (Plataforma como Servicio): Ej. Google App Engine**

La **plataforma como servicio (PaaS)** es la encapsulación de un ambiente de desarrollo y la provisión de una serie de módulos que proporcionan una funcionalidad horizontal (persistencia de datos, autenticación, mensajería, etc.). De esta forma, una estructura básica de este tipo podría consistir en un entorno que contenga servicios/aplicaciones, librerías y API para un fin específico.

Es decir, el cliente no gestiona ni controla la infraestructura (ni servidores, ni sistemas operativos, ni almacenamiento, ni ningún tipo de elementos de red o seguridad, etc.), pero tiene el control de las aplicaciones desplegadas y de la configuración del entorno de ejecución de las mismas (bases de datos y middelwares). Es habitual que una PaaS pueda prestar servicios en todos los aspectos del ciclo de desarrollo y pruebas de software o en un desarrollo, gestión y publicación de contenidos.

Como resumen en esta modalidad el deseo del usuario sería «Necesito este software instalado sobre este operativo y esta base de datos y con esta API», y recibiría todo el conjunto (hw, SO, base de datos, librerías, API, seguridad, GUI y otras herramientas) listo para usar en pocos segundos y desarrollar aplicaciones empresariales/móviles, páginas web, contenidos, etc.

**- SaaS (Software como Servicio): Ej. Google Workspace, Microsoft 365**

Finalmente, softwarecomoservicio (SaaS) se encuentra en la capa abstracta (si bien existe una tendencia bastante extendida a considerar, que el SaaS es la capa más simple del PaaS, o la más baja) y caracteriza una aplicación completa ofrecida como un servicio, bajo demanda generalmente con tenencia múltiple (arquitectura de software en la cual una sola instancia de la aplicación se ejecuta en el servidor, pero sirviendo a múltiples clientes u organizaciones). En general, las aplicaciones bajo este modelo de servicio son accesibles a través de un navegador web y el usuario no tiene control sobre ellas, aunque en algunos casos se le permite realizar algunas configuraciones. Esto le elimina la necesidad al cliente de instalar la aplicación en sus propios ordenadores, eliminando el soporte y mantenimiento del hardware y software y mejorando el control y gestión de licencias si son necesarias.

En esta modalidad se podría resumir que el pensamiento del usuario sería «Ejecute esto por mí» sin responsabilidades en la gestión de hardware o software, utilizando un navegador web o evitando la instalación de software cliente, basado en un servicio bajo demanda, con escalabilidad casi inmediata, acceso desde cualquier sitio y con cualquier dispositivo, bajo un modelo de soporte de 24/7 y un modelo de pago como pay-as-they-go y pay-as-they-grow.

**1.5 Beneficios del cómputo en la nube**

**- Reducción de costos**

**- Mayor disponibilidad**

**- Flexibilidad operativa**

**- Innovación más rápida**

Para convencer a los más reticentes, se pueden enumerar un conjunto de ventajas que aporta el cloud como paradigma, que son (orden alfabético):

1. **Fácil integración y aceptación**: teniendo en cuenta su desarrollo y base en estándares, se puede integrar con mucha mayor facilidad y rapidez con el resto de las aplicaciones empresariales; el usuario final queda totalmente convencido cuando se le permite acceder desde cualquier dispositivo sin requerimientos especiales en cuanto a las necesidades del mismo.
2. **Servicio global**: ubicuidad y acceso a los servicios desde cualquier lugar con tiempo de inactividad reducidos al mínimo y con alta disponibilidad de los recursos.
3. **Simplicidad**: roles y responsabilidades muy bien definidos, separación de las actividades bien estipuladas (por ejemplo, proveedor de contenidos, proveedor de aplicación, proveedor de infraestructura y usuario final), lo cual se traduce en una forma óptima de trabajar y menor inversión por todas partes.
4. **Reducción de riesgos y rapidez**: no es necesario una gran inversión ni adecuación de sitios/entornos, siendo posible acelerar al máximo la implantación de nuevos servicios en las diferentes etapas de desarrollo hasta producción.
5. **Reducción del uso de energía (consumo eficiente)**:dada la tecnología utilizada y la utilización eficiente de los recursos (favorecido por la virtualización), solo se consume lo necesario, a diferencia de los centros de datos tradicionales en los que existe un consumo fijo no dependiente de la carga/utilización.

Si bien las ventajas son evidentes y muchos actores de esta tecnología la basan principalmente en el abaratamiento de los costes de servicio, comienzan a existir opiniones en contra que consideran que un cloud público puede no ser la opción más adecuada para determinado tipo de servicios/infraestructura.

**1.6 Retos y limitaciones**

**- Seguridad y privacidad**

**- Dependencia del proveedor**

**- Latencia y conectividad**

**- Cumplimiento normativo**

Entre las principales causas de desventajas que esgrimen algunos expertos, están (orden alfabético):

1. **Centralización**: tanto de los datos como de las aplicaciones; genera una dependencia en relación con el proveedor; si no dispone de la tecnología adecuada (monitorización y detección) y recursos apropiados (alta disponibilidad), puede generar cortes o inestabilidades en el servicio. En estos casos toma especial relevancia la SLA (service level agreement), que especificará a qué está obligado el proveedor y las indemnizaciones que deberá hacer frente por ello.
2. **Confiabilidad**: la «salud» tecnológica y financiera del proveedor será un elemento clave en la continuad de su negocio y el del cliente, por lo cual las decisiones que tome el proveedor afectarán directamente al negocio cliente, y si estas no son las adecuadas, afectarán directamente a la empresa. También la empresa quedaría a merced de un mercado muy dinámico en cuanto a fusiones y monopolio (o pseudomonopolios) con el impacto que podría tener en los costos de los servicios. Un problema importante es que el proveedor por diversos motivos (legales, financieros, económicos) cierre su actividad de forma abrupta y se produzca el data lock-in de la empresa en los servidores del proveedor sin posibilidad de poder recuperarlos (situaciones que han pasado con el bloqueo judicial de un proveedor por diversas razones con el bloqueo de los datos de los clientes hasta que se resuelva el conflicto judicial).
3. **Escalabilidad**: a medida que el proveedor dispongan de más clientes, más usuarios habrá sobre el hardware, la sobrecarga en estos aumentará, y si el proveedor no dispone de un plan de escalabilidad a medio y largo plazo, para asegurar un crecimiento sostenible desde el punto de vista de las necesidades de sus clientes, se puede llegar a la saturación de los servicios, con la consiguiente degradación y pérdida de prestaciones.
4. **Especialización o cualificación**: la necesidad de servicios «especiales» o cualificados podrían tener una prioridad muy baja (y su consiguiente retardo en su despliegue) para el proveedor si estos no son requeridos por otros clientes, afectando al negocio de uno en particular que sí los necesita para el suyo.
5. **Disponibilidad**: el principal punto débil de una infraestructura en cloud es el acceso a internet. Si no se dispone de él y es confiable y de un ancho de banda aceptable, el cloud deja de tener efectividad.
6. **Riesgo y privacidad**: los datos «sensibles» de negocio no residen en las instalaciones de las empresas y la seguridad no depende de los recursos humanos de esta sino del proveedor del servicio. Si se asume que estos datos son de alto valor en un contexto vulnerable, el riesgo puede ser muy alto por su posible robo (copia), acceso a la información (lectura) o destrucción (borrado).
7. **Seguridad**: dado que la información deberá atravesar diferentes canales y servicios, puede resultar que cada uno de ellos sea un foco de inseguridad. Si bien esto puede ser resuelto mediante canales y servicios seguros, la posibilidad de un fallo en la cadena de cifrado de la información puede ser posible con los consiguientes problemas que representa. Además, en este caso el propietario de la información puede desconocer totalmente qué ha pasado y dónde ha estado el fallo.
8. **Vendor lock-in**: es un gran problema (que en la actualidad se ha demostrado como uno de los más frecuentes) que hace que un cliente dependa de un proveedor de productos y servicios, incapaz de usar otro proveedor sin costos de cambio sustanciales, aunque el cambio signifique una reducción de costos o mejores prestaciones. Muchos desarrolladores o usuarios finales son reticentes al cloud por este motivo, ya que significa un compromiso adquirido; después, si se desea cambiar el costo, será muy elevado.

**Actividades posibles:**

**Actividades Teóricas**

1. **Lectura guiada y resumen**
   * **Lectura de un artículo o capítulo sobre la historia del cómputo en la nube.**
   * **Elaboración de un resumen con línea del tiempo de su evolución.**
2. **Mapa conceptual**
   * **Crear un mapa conceptual que relacione los modelos de servicio (IaaS, PaaS, SaaS) y los modelos de despliegue (público, privado, híbrido, comunitario).**
3. **Debate en clase**
   * **Tema: *¿Es más seguro tener tus datos en la nube o en servidores locales?***
   * **Se divide la clase en dos grupos para argumentar a favor y en contra.**
4. **Cuestionario de repaso**
   * **Preguntas de opción múltiple, verdadero/falso y de desarrollo corto para reforzar conceptos clave.**

**Temario modulo 2 (*Principales tecnologías del cloud*)**

**2.1 Servicios de cómputo**

Se hará un breve recorrido en torno a las principales tecnologías en las cuales se apoya la metodología manejada, como manera de entender desde un punto de vista técnico el comportamiento de este nuevo paradigma en cuanto al uso y manejo de las herramientas existentes en el contexto global tecnológico bajo el cual se desarrolla el Cloud Computing.

**Equilibrado de carga**

**¿Qué es el equilibrio de carga?**

El equilibrio de carga es el método de distribuir el tráfico de red de manera equitativa en un conjunto de recursos que admiten una aplicación. Las aplicaciones modernas tienen que procesar millones de usuarios simultáneamente y devolver el texto, los videos, las imágenes y otros datos correctos a cada usuario de manera rápida y fiable. Para gestionar volúmenes de tráfico tan altos, la mayoría de las aplicaciones tienen muchos servidores de recursos con datos duplicados entre ellos. Un equilibrador de carga es un dispositivo que se encuentra entre el usuario y el grupo de servidores y actúa como un facilitador invisible, lo que garantiza que todos los servidores de recursos se usen por igual.

**¿Cómo funciona el equilibrio de carga?**

Las empresas suelen tener sus aplicaciones ejecutándose en varios servidores. Esta disposición de servidores se denomina “granja de servidores”. Las solicitudes de los usuarios a la aplicación van primero al equilibrador de carga. A continuación, el equilibrador de carga enruta cada solicitud a un único servidor de la granja de servidores más adecuada para gestionar la solicitud.

El equilibrio de carga es como el trabajo realizado por un gerente en un restaurante. Imagine un restaurante con cinco camareros. Si se le permitiera a los clientes elegir a sus camareros, uno o dos camareros podrían verse con una sobrecarga de trabajo mientras los demás están inactivos. Para evitar este escenario, el gerente del restaurante asigna los clientes a los camareros específicos que mejor servicio pueden ofrecerles.

¿Qué son los algoritmos de equilibrio de carga?

Un algoritmo de equilibrio de carga es el conjunto de reglas que sigue un equilibrador de carga para determinar el mejor servidor para cada una de las diferentes solicitudes de los clientes. Los algoritmos de equilibrio de carga se dividen en dos categorías principales.

**Equilibrio de carga estática**

Los algoritmos de equilibrio de carga estática siguen reglas fijas y son independientes del estado actual del servidor. Los siguientes son ejemplos de equilibrio de carga estática.

***Método* de *turnorotativo***

Los servidores tienen direcciones IP que indican al cliente dónde enviar las solicitudes. La dirección IP es un número largo y difícil de recordar. Para hacerlo más fácil, un sistema de nombres de dominio asigna los nombres de los sitios web a los servidores. Cuando escribe [aws.amazon.com](https://aws.amazon.com/) en su navegador, la solicitud va primero a nuestro servidor de nombres, que devuelve nuestra dirección IP a su navegador.

En el método de operación por turnos, un servidor de nombres autoritativo hace el equilibrio de carga en lugar de hardware o software especializado. El servidor de nombres devuelve las direcciones IP de los diferentes servidores de la granja de servidores por turnos o por turnos.

***Método de* *turno rotativo* *ponderado***

En el equilibrio de carga por turnos ponderados, puede asignar diferentes pesos a cada servidor en función de su prioridad o capacidad. Los servidores con ponderaciones más altas recibirán más tráfico de aplicaciones entrante del servidor de nombres.

***Método hash de IP***

En el método hash de IP, el equilibrador de cargas lleva a cabo un cálculo matemático, denominado “hash”, en la dirección IP del cliente. Convierte la dirección IP del cliente en un número, que luego se asigna a servidores individuales.

**Equilibrio de carga dinámico**

Los algoritmos de equilibrio de carga dinámico examinan el estado actual de los servidores antes de distribuir el tráfico. Los siguientes son algunos ejemplos de algoritmos de equilibrio de carga dinámico.

***Método de conexión mínima***

Una conexión es un canal de comunicación abierto entre un cliente y un servidor. Cuando el cliente envía la primera solicitud al servidor, se autentica y establece una conexión activa entre sí. En el método de conexión mínima, el equilibrador de carga comprueba qué servidores tienen la menor cantidad de conexiones activas y envía tráfico a esos servidores. Este método supone que todas las conexiones requieren la misma potencia de procesamiento para todos los servidores.

***Método de conexión mínima ponderada***

Los algoritmos de conexión mínima ponderada suponen que algunos servidores pueden gestionar más conexiones activas que otros. Por lo tanto, puede asignar diferentes pesos o capacidades a cada servidor y el equilibrador de carga envía las nuevas solicitudes de clientes al servidor con la menor cantidad de conexiones por capacidad.

***Método de tiempo de respuesta mínimo***

El tiempo de respuesta es el tiempo total que tarda el servidor en procesar las solicitudes entrantes y enviar una respuesta. El método de menor tiempo de respuesta combina el tiempo de respuesta del servidor y las conexiones activas para determinar el mejor servidor. Los equilibradores de carga utilizan este algoritmo para garantizar un servicio más rápido para todos los usuarios.

***Método basado en recursos***

En el método basado en recursos, los equilibradores de carga distribuyen el tráfico analizando la carga actual del servidor. Un software especializado llamado “agente” se ejecuta en cada servidor y calcula el uso de los recursos del servidor, como su capacidad de computación y memoria. A continuación, el equilibrador de carga comprueba si hay suficientes recursos libres en el agente antes de distribuir el tráfico a ese servidor.

**Virtualización**

De manera genérica, la virtualización se puede definir como una tecnología capaz de gestionar y administrar de manera concurrente diferentes entornos y recursos dedicados (tanto software, como hardware), desde una sola unidad física operativa o CPU, abstrayendo y aislando entre si cada uno de ellos, de manera que el conjunto de los mismos sea ejecutado en paralelo de una forma opaca y totalmente independiente.

Durante este proceso, el software de virtualización se ejecuta por debajo de la capa hardware origen o CPU, dando como resultado una división del Sistema en diferentes entornos autónomos e independientes. Cada entorno generado es definido comúnmente bajo las siglas VM (Virtual Machine), y depende directamente de los límites definidos dentro del Sistema origen CPU, así como de la capacidad de gestión del software de virtualización, el cual, se hace cargo de repartir y gestionar paralelamente los diferentes recursos requeridos por cada entorno en base a sus necesidades operativas específicas.

Gracias a ello, la virtualización permite al usuario llevar a cabo diferentes procedimientos que, mediante la operativa y gestión de una única unidad física o CPU, serían inconcebibles a nivel técnico, como por ejemplo, la configuración de entornos y migraciones específicas en los procesos de validación de nuevo hardware, o la creación de diferentes bancos de pruebas en los procesos de testeo regresivo, sin la necesidad añadida de utilizar un hardware específico para cada una de las pruebas o servidores de desarrollo.

En materia de seguridad e integridad de la información y los datos manejados, la virtualización ha de integrase de tal forma que los entornos generados repliquen las medidas preventivas necesarias en la mitigación de ataques maliciosos de una forma común. En lo que se refiere a un entorno de desarrollo, cada usuario posee su propio espacio aislado, lo que evita que el código malicioso afecte de manera global al resto de entornos concurrentes que estén siendo simulados en la misma unidad física de origen.

**Principales Métodos de Virtualización**

Respecto a los tipos de virtualización, existen múltiples enfoques y métodos asociados a las diferentes características y recursos simulados en cada situación concreta, lo cual, plantea un gran abanico de posibilidades en relación a la combinatoria obligada de software y hardware. No obstante, a continuación, se presentan algunos de los principales tipos de virtualización desde un punto generalizado de uso:

**Emulación**: El proceso de simulación se realiza para un hardware completo, admitiendo en este punto un nuevo SO totalmente virtualizado, el cual, dote al usuario de la experiencia de obtener una CPU completamente diferente. Este método está orientado a la creación y desarrollo de software en máquinas que no se encuentren físicamente disponibles.

**Virtualización Nativa/Completa**: La VM simula las distintas características hardware requeridas por un SO, de tal forma que se pueda ejecutar un nuevo SO de forma aislada, sin modificar los parámetros y configuraciones de la CPU original.

**Virtualización Parcial o Address Space Virtualization**: En este punto, la VM simula instancias concretas del hardware que se desea virtualizar, concretamente, las llamadas Address Spaces o espacios de direccionamiento. Este método dota al usuario de la capacidad de compartir recursos y aislar procesos dentro de un marco virtualizado sin modificar las configuraciones del SO original.

**Virtualización de SO**: Este método permite virtualizar de manera completa un SO bajo un mismo servidor, permitiendo así crear múltiples servidores virtuales aislados y ejecutarlos de manera concurrente dentro de una misma CPU física. Las aplicaciones que corren dentro de cada uno de los diferentes SO virtuales disponen de independencia y opacidad entre sí, conformando para ellas un sistema autónomo en sí mismo por cada servidor generado.

**Virtualización de Aplicaciones**: La virtualización de aplicaciones está basada en la ejecución de un software o programa en un entorno local generado dentro de la misma CPU, con recursos locales a su vez. Mediante esta operativa, el usuario posee la capacidad de generar un pequeño entorno virtual con los componentes mínimos requeridos por las rutinas de la aplicación simulada, tales como entradas a registros, archivos u objetos globales de referencia. La principal ventaja que aporta este método es la eliminación de los posibles conflictos existentes entre las aplicaciones a ejecutar, y aplicaciones nativas del sistema físico original.

**Bases de Datos no Relacionales**

Las bases de datos NoSQL, también conocidas como bases de datos personalizadas, están diseñadas para modelos de datos específicos y almacenan los datos en esquemas flexibles que se escalan con facilidad para aplicaciones modernas. Muchas cargas de trabajo de bases de datos pueden beneficiarse de la rentabilidad y el rendimiento de las bases de datos NoSQL.

**¿Cuáles son las ventajas de las bases de datos NoSQL?**

Las aplicaciones modernas se enfrentan a varios desafíos que las bases de datos NoSQL de pueden resolver. Por ejemplo, las aplicaciones procesan un gran volumen de datos de fuentes dispares, como las redes sociales, los sensores inteligentes y las bases de datos de terceros. Todos estos datos dispares no encajan perfectamente en el modelo relacional. La aplicación de estructuras tabulares puede provocar redundancia, duplicación de datos y problemas de rendimiento a escala.

Las bases de datos NoSQL están diseñadas específicamente para modelos de datos no relacionales y tienen esquemas flexibles para crear aplicaciones modernas. Son ampliamente reconocidas por su facilidad de desarrollo, su funcionalidad y el rendimiento a escala. Los beneficios de las bases de datos NoSQL se enumeran a continuación.

**Flexibilidad**

Las bases de datos NoSQL generalmente ofrecen esquemas flexibles que permiten un desarrollo más rápido y más iterativo. El modelo de datos flexible hace que las bases de datos NoSQL sean ideales para datos semiestructurados y no estructurados.

**Escalabilidad**

Las bases de datos NoSQL generalmente están diseñadas para escalar horizontalmente usando clústeres distribuidos de hardware, en lugar de escalar añadiendo servidores caros y sólidos. Algunos proveedores de la nube manejan estas operaciones en segundo plano, como un servicio completamente administrado.

**Alto rendimiento**

Las bases de datos NoSQL están optimizadas para modelos de datos y patrones de acceso específicos. Esto permite un mayor rendimiento que si intentara lograr una funcionalidad similar con bases de datos relacionales.

**Altamente funcional**

Las bases de datos NoSQL proporcionan API altamente funcionales y tipos de datos que están diseñados específicamente para cada uno de sus respectivos modelos de datos.

**¿Cuáles son los tipos de bases de datos NoSQL?**

Existen varios sistemas de bases de datos NoSQL diferentes debido a las variaciones en la forma en que administran y almacenan los datos sin esquemas. A continuación, explicamos algunos de los tipos más comunes.

**Base de datos de clave-valor**

Las bases de datos de clave-valor son altamente particionables y permiten un escalado horizontal en niveles que otros tipos de bases de datos NoSQL no pueden lograr. Una base de datos de clave-valor almacena datos como un conjunto de pares clave-valor en los que una clave sirve como un identificador único. Las claves y los valores pueden ser cualquier cosa, desde objetos simples hasta objetos compuestos complejos. Los casos de uso como juegos, tecnología publicitaria e IoT se prestan particularmente bien con el diseño de datos de almacén clave-valor. Amazon DynamoDB está diseñado para proporcionar un rendimiento constante con una latencia de milisegundos de un solo dígito para cualquier escala de cargas de trabajo.

**Bases de datos de documentos**

Las bases de datos de documentos tienen el mismo formato de modelo de documento que los desarrolladores utilizan en el código de sus aplicaciones. Almacenan los datos como objetos JSON que son flexibles, semiestructurados y de naturaleza jerárquica. La naturaleza flexible, semiestructurada y jerárquica de los documentos y las bases de datos de documentos permite que evolucionen según las necesidades de las aplicaciones. El modelo de base de datos de documentos funciona bien con catálogos, perfiles de usuario y sistemas de administración de contenido en los que cada documento es único y evoluciona con el tiempo. Amazon DocumentDB (con compatibilidad para MongoDB) y MongoDB son bases de datos de documentos conocidas que proporcionan unas API poderosas e intuitivas para un desarrollo flexible e iterativo.

**Bases de datos de grafos**

El objetivo de las bases de datos de grafos es facilitar la creación y ejecución de aplicaciones que funcionen con conjuntos de datos con un nivel alto de conexión. Usan nodos para almacenar entidades de datos y periferias para almacenar relaciones entre entidades. Un borde siempre tiene un nodo inicial, un nodo final, un tipo y una dirección. Puede describir las relaciones entre elementos principales y secundarios, las acciones, la propiedad y similares. No hay límite para la cantidad y el tipo de relaciones que un nodo puede tener. Puede usar una base de datos de grafos para crear y ejecutar aplicaciones que funcionen con conjuntos de datos con un nivel alto de conexión. Los casos de uso típicos para una base de datos de grafos incluyen redes sociales, motores de recomendaciones, detección de fraude y gráficos de conocimiento. Amazon Neptune es un servicio de base de datos de grafos totalmente administrado que admite el modelo Property Graph y el resource description framework (RDF, marco de descripción de recursos) con la opción de dos API de gráficos (TinkerPop y RDF/SPARQL).

**Bases de datos en memoria**

Mientras que otras bases de datos no relacionales almacenan datos en discos o SSD, los almacenes de datos en memoria están diseñados para eliminar la necesidad de acceder a los discos. Son ideales para aplicaciones que requieren tiempos de respuesta de microsegundos o que tienen grandes picos de tráfico. Puede usarlas en aplicaciones de juegos y tecnología publicitaria para funciones como tablas de clasificación, tiendas de sesiones y análisis en tiempo real. Amazon MemoryDB para Redis es un servicio de base de datos en memoria duradero y compatible con Redis que ofrece latencia de lectura de microsegundos, latencia de escritura de milisegundos de un solo dígito y durabilidad Multi-AZ. Amazon ElastiCache es un servicio de almacenamiento en caché en memoria completamente administrado compatible con Redis y Memcached, para atender cargas de trabajo de baja latencia y alto rendimiento. El Acelerador de Amazon DynamoDB (DAX) es otro ejemplo de un almacén de datos diseñado específicamente que hace que las lecturas de DynamoDB sean mucho más rápidas.

**Buscar en bases de datos**

Una base de datos de motores de búsqueda es un tipo de base de datos no relacional que se dedica a la búsqueda de contenido de datos, como los registros de salida de las aplicaciones que utilizan los desarrolladores para solucionar problemas. Utilizan índices para categorizar características similares entre los datos y facilitar la capacidad de búsqueda. Las bases de datos de los motores de búsqueda están optimizadas para clasificar datos no estructurados, como imágenes y videos. Amazon OpenSearch Service está diseñado para proporcionar visualizaciones en tiempo real y análisis de datos generados por máquinas al indexar, agregar y buscar registros y métricas semiestructuradas.

**Almacenamiento**

**¿Qué es el almacenamiento en la nube?**

El almacenamiento en la nube es un modelo de computación en la nube que permite almacenar datos y archivos en Internet a través de un proveedor de computación en la nube, al cual se accede mediante la red pública de Internet o una conexión de red privada dedicada. El proveedor almacena, administra y mantiene de manera segura los servidores de almacenamiento, la infraestructura y la red para garantizar que tiene acceso a los datos cuando lo necesite, prácticamente a cualquier escala y con capacidad elástica. El almacenamiento en la nube hace que ya no sea necesario comprar y administrar su propia infraestructura de almacenamiento de datos, lo que le brinda agilidad, escalabilidad y durabilidad, con acceso a los datos en cualquier momento y lugar.

**¿Cuáles son los tipos de almacenamiento en la nube?**

Hay tres tipos principales de almacenamiento en la nube: almacenamiento de objetos, de archivos y de bloques. Cada uno ofrece sus propias ventajas y tiene sus propios casos de uso.

**Almacenamiento de objetos**

Las organizaciones tienen que almacenar una cantidad masiva y cada vez mayor de datos no estructurados, como fotos, videos, machine learning (ML), datos de sensores, archivos de audio y otros tipos de contenido web, y encontrar formas escalables, eficientes y asequibles de almacenarlos puede ser un desafío. El almacenamiento de objetos es una arquitectura de almacenamiento de datos para grandes almacenes de datos no estructurados. Los objetos almacenan los datos en el formato en el que llegan y permiten personalizar los metadatos de forma que faciliten el acceso y el análisis de los datos. En lugar de organizarse en archivos o jerarquías de carpetas, los objetos se guardan en buckets seguros que ofrecen una escalabilidad prácticamente ilimitada. También es menos costoso almacenar grandes volúmenes de datos.

Las aplicaciones desarrolladas en la nube suelen aprovechar la gran escalabilidad y las características de los metadatos del almacenamiento de objetos. Las soluciones de almacenamiento de objetos son ideales para crear aplicaciones modernas desde cero que requieren escala y flexibilidad, y que también puede utilizar para importar almacenes de datos existentes para su análisis, copia de seguridad o archivado.

**Almacenamiento de archivos**

El almacenamiento basado en archivos o almacenamiento de archivos se utiliza mucho entre las aplicaciones y almacena los datos en un formato jerárquico de carpetas y archivos. Este tipo de almacenamiento se conoce a menudo como servidor de almacenamiento conectado a la red (NAS) con protocolos comunes para los archivoa de Server Message Block (SMB) que se utilizan en las instancias de Windows y Network File System (NFS) que se encuentran en Linux.

**Almacenamiento en bloques**

Las aplicaciones empresariales como bases de datos o sistemas de planificación de recursos empresariales (ERP) a menudo requieren almacenamiento dedicado y de baja latencia para cada host. Esto es similar al almacenamiento conectado directamente (DAS) o a una red de área de almacenamiento (SAN). En este caso, se puede utilizar un servicio de almacenamiento en la nube que almacena los datos en forma de bloques. Cada bloque tiene su propio identificador único para un rápido almacenamiento y recuperación.

**- Máquinas virtuales (VMs): creación, configuración y escalado**

**- Contenedores: introducción a Docker y Kubernetes**

**- Funciones como servicio (FaaS): introducción a AWS Lambda, Azure Functions**

**2.2 Servicios de almacenamiento**

**- Almacenamiento de objetos: Amazon S3, Azure Blob Storage, Google Cloud Storage**

**- Almacenamiento en bloques y archivos**

**- Gestión de backups y recuperación**

**2.3 Servicios de bases de datos**

**- Bases de datos relacionales (RDS, Cloud SQL)**

**- Bases de datos NoSQL (DynamoDB, Firestore)**

**- Comparación entre modelos y casos de uso**

**2.4 Servicios de red y conectividad**

**- Redes virtuales (VPC, VNets)**

**- Balanceadores de carga**

**- DNS en la nube**

**- Firewalls y gateways**

**2.5 Servicios gestionados y adicionales**

**- Servicios de mensajería (Pub/Sub, SQS)**

**- Servicios de monitoreo (CloudWatch, Azure Monitor)**

**- Servicios de inteligencia artificial y análisis de datos**

**2.6 Actividad práctica**

**- Despliegue de una aplicación sencilla que utilice al menos tres servicios: cómputo, almacenamiento y base de datos**