

Maestría en Desarrollo y Operaciones de  
Software

---

# Cloud Computing, DevOps y DevOps Culture

Cloud Computing, DevOps y DevOps Culture

---

## Tema 1. ¿Qué es DevOps?

# Índice

## Esquema

### Ideas clave

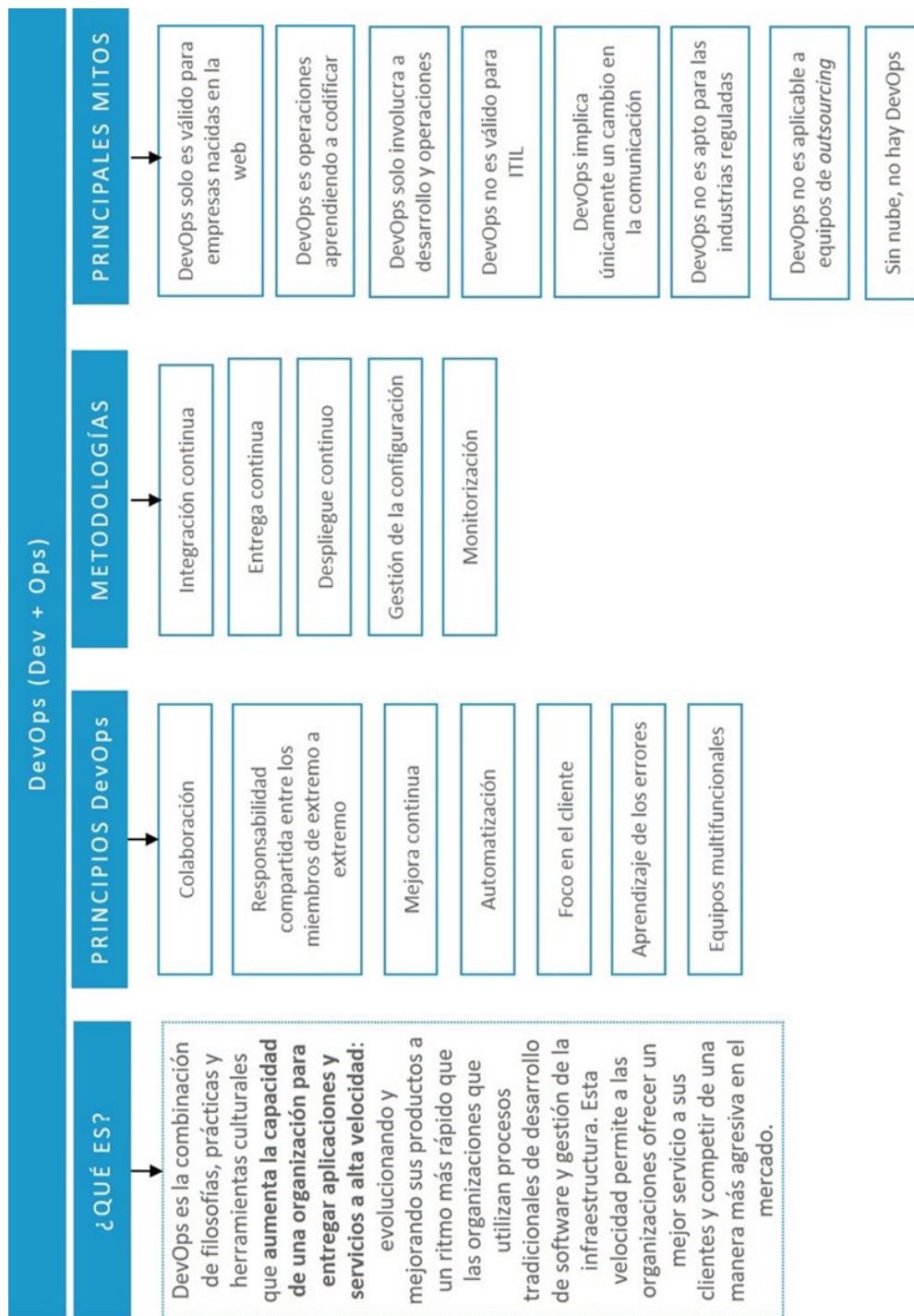
- 1.1. Introducción y objetivos
- 1.2. ¿A qué llamamos DevOps?
- 1.3. Principios DevOps
- 1.4. Metodologías y estrategias DevOps
- 1.5. Las capacidades de DevOps
- 1.6. Mitos DevOps
- 1.7. Referencias bibliográficas

### A fondo

¿Qué tareas hace un profesional de DevOps?

What is DevOps?

### Test



## 1.1. Introducción y objetivos

DevOps no puede entenderse únicamente como un conjunto de tecnologías y mejores prácticas, sino que debe entenderse como una revolución cultural. Es una corriente de cambio dentro de las organizaciones en la que se prima la colaboración y la interacción abierta entre los equipos de tecnología de la información (en siglas, TI) y aquellos de desarrollo. El objetivo primordial de esta sinergia es la de alcanzar la máxima eficiencia y agilidad en el desarrollo, despliegue y mantenimiento de las aplicaciones.

Numerosas organizaciones han incluido la virtualización como la mejor forma de optimizar sus recursos computacionales, ya que da nueva vida a infraestructuras obsoletas y obtienen una flexibilidad mucho mayor a la hora de utilizar sus recursos computacionales. La revolución de la virtualización ya no es tal, y el nuevo ciclo de cambio ya está en marcha. Ya no es suficiente hablar de Cloud Computing como una forma de obtener recursos computacionales de manera elástica y en un modelo de pago por uso, sino que además aparece el concepto de *datacenter* virtual, donde podemos tener las ventajas de un *datacenter* con los beneficios de la nube.

A través del estudio de este tema, se pretenden conseguir los siguientes objetivos:

- ▶ Comprender el concepto de DevOps y cómo se aplican las tecnologías y mejores prácticas que se conocen como DevOps.
- ▶ Conocer los factores que intervienen en la implementación del cambio cultural necesario para la adopción de la metodología DevOps en una organización.
- ▶ Analizar los beneficios que reporta DevOps en los proyectos de software.

## 1.2. ¿A qué llamamos DevOps?

**DevOps** es un término relativamente nuevo para describir lo que también ha sido llamado como **administración de sistemas ágiles**. DevOps se centra en el trabajo y la colaboración de los equipos de desarrollo y operaciones.

En la actualidad, las empresas se mueven a la velocidad de la nube, y es por ello por lo que DevOps se ha convertido en un enfoque cada vez más extendido y popular para la entrega de *software*. Los equipos de desarrollo y operaciones utilizan esta metodología para construir, probar, implementar y monitorizar las aplicaciones debido a que les permite: actuar con velocidad, mantener altos índices de calidad y controlar los cambios con suma rapidez.

DevOps es esencial para cualquier empresa que aspire a ser ágil y que pretenda ser capaz de responder rápidamente a las demandas del mercado. Por lo tanto, DevOps es un enfoque:

- ▶ Hacia la entrega de software ágil.
- ▶ Que promueve una colaboración estrecha entre las líneas de negocio, desarrollo y operaciones de TI.
- ▶ Que elimina las barreras entre las partes interesadas y los clientes.

Es necesario que todas las partes involucradas en el proceso de entrega de software estén dispuestas a colaborar entre sí. Los **equipos de desarrollo** necesitan diseñar, desarrollar, entregar y ejecutar el software de la manera más rápida y confiable posible. Los **equipos de operaciones** deben identificar y resolver los problemas de forma temprana mediante la monitorización, la predicción de fallos, la administración de los entornos y la solución de problemas. La combinación de este enfoque común permite monitorizar y analizar los cuellos de botella y optimizar las capacidades de

DevOps, que busca, ante todo, fomentar la colaboración entre el área de negocio y los departamentos de desarrollo y operaciones para entregar y ejecutar el software lo antes posible.

## DevOps como herramienta de cambio para las organizaciones

Para ir adentrándonos en la materia, necesitamos comprender la **necesidad** y el **valor** de DevOps a nivel de negocio, así como los **principios** de DevOps. Como sabrán, realizar un cambio en el «Business as usual» siempre es difícil y, por lo general, requiere de una inversión. Así que, cada vez que una organización adopta cualquier nueva tecnología, metodología o enfoque, la adopción de esta tiene que ser impulsada por una **necesidad de negocio**. Es decir que, al desarrollar un modelo de negocio para la adopción de DevOps, se debe entender la necesidad de la empresa, incluyendo los retos a los que se enfrenta.

En un entorno DevOps, los equipos son responsables de ofrecer nuevas características, pero también estabilidad, escalabilidad, fiabilidad y otro sinfín de propiedades comunes a un software de calidad. Es por ello por lo que las responsabilidades se equilibran para garantizar que ambos equipos (desarrollo y operaciones) tengan visibilidad del rendimiento de la aplicación a través de todas las etapas del ciclo de vida.

Debido a los beneficios que reporta a todo el conjunto de la organización, DevOps continúa ganando adeptos año tras año desde su aparición. Según los datos de la popular web DevOps.com, la tasa de adopción aumentó sustancialmente, de 2015 a 2016, del 66 % al 74 %. ¿Qué impulsa a las empresas hacia la adopción de DevOps? La respuesta es muy sencilla: los largos plazos de entrega del software hasta su paso a producción, derivado de los sistemas de trabajo tradicionales, impiden a las empresas brindar servicios de vanguardia y mejorar la experiencia del cliente. Para mantener el ritmo de las demandas del mercado, los equipos de TI deben construir, implementar, probar y lanzar software en ciclos cada vez más rápidos.

Debido a que DevOps mejora la forma en que una empresa entrega valor a sus clientes, proveedores y socios, podemos afirmar que es un proceso esencial de negocio y no solo una capacidad de TI. No debemos olvidar que esta metodología ofrece un retorno significativo de la inversión, en inglés *Return of Investment* (en adelante, ROI), por diversas razones:

## Aceleración de la innovación

Con un sólido equipo de operaciones y desarrollo que trabaje acompañadamente, las aplicaciones se pueden desarrollar y desplegar con mayor rapidez. Esto es vital, ya que **el éxito empresarial actual depende, en gran medida, de la capacidad de una organización para innovar más rápido que la competencia**. Los ingenieros DevOps pueden aprovechar las herramientas tecnológicas de las que disponen para comprobar los datos de rendimiento de sus aplicaciones en tiempo real, y así, comprender el impacto de los cambios introducidos en el código. A su vez, las soluciones a los problemas derivados del software se proveen en plazos de tiempo más cortos porque los miembros del equipo solo necesitan verificar los últimos cambios de código para detectar el origen de los errores.

## Mejora de la colaboración

En vez de intentar desdibujar las diferencias entre ambos departamentos (desarrollo y operaciones), un entorno DevOps exitoso procura **construir un puente basado en la comunicación y la colaboración para crear una sinergia**. La cultura de desarrollo de software, entonces, se enfoca en objetivos, responsabilidades y logros compartidos, en lugar de objetivos individuales. Cuando estos equipos adquieren un buen nivel de confianza y colaboración, pueden experimentar, investigar e innovar de manera más efectiva. El entorno de desarrollo se vuelve, progresivamente, más uniforme a medida que todos los miembros del equipo trabajan para alcanzar objetivos compartidos.

## Incremento de la eficiencia

Las herramientas automatizadas y las plataformas de producción estandarizadas son elementos clave de las mejores prácticas de DevOps, que ayudan a que las implementaciones sean más predecibles y liberen al personal de TI de tediosas tareas repetitivas. Con un entorno de pruebas e integración automatizadas, los desarrolladores no necesitan desperdiciar su tiempo confiando en que se completen los procesos de integración de código. Estas herramientas ofrecen oportunidades adicionales para mejorar la eficiencia:

- ▶ La **infraestructura escalable**, como las soluciones basadas en la nube, ayudan a acelerar las pruebas y los procesos de implementación al aumentar el acceso a los recursos de hardware.
- ▶ Las **herramientas de compilación y desarrollo** ayudan a acortar los ciclos de desarrollo y acelerar la entrega del producto.
- ▶ Los **flujos de trabajo de entrega continua** ayudan a entregar software de manera más rápida y frecuente.

## Reducción del número de fallos

Los ciclos de desarrollo más cortos, derivados de DevOps, promueven entregas de código más frecuentes. Con estas implementaciones modulares, los equipos pueden descubrir e identificar, de forma temprana, problemas asociados a la configuración, el código o la infraestructura. **DevOps eleva el grado de compromiso de los miembros del equipo durante todo el ciclo de vida de una aplicación, lo que da como resultado un código de mayor calidad.** Por lo general, se requieren menos cambios y modificaciones porque los desarrolladores identifican y eliminan posibles problemas a medida que escriben el código. Según un informe reciente elaborado por Puppet Labs, llamado [State of DevOps](#), las organizaciones que adoptan una cultura DevOps tienen 60 veces menos fallos que aquellas que no implementan DevOps.

## Aceleración del tiempo de recuperación

Como hemos dicho anteriormente, debido a que las implementaciones de DevOps son más frecuentes y «pequeñas», **los errores son fácilmente detectables y es por ello por lo que las soluciones se encuentran con mayor rapidez.** El equipo necesitará verificar los últimos cambios en el código para poder resolver una incidencia en vez de revisar enormes cantidades de código. Los tiempos de resolución son intrínsecamente más rápidos porque, además, la responsabilidad en la solución de los problemas y las correcciones pertinentes corresponden a un solo equipo: el equipo DevOps. En el informe [State of DevOps](#) de Puppet Labs, se establece que los equipos DevOps de alto rendimiento se recuperan de los fallos 168 veces más rápido que aquellos con menor rendimiento.

## Aumento de la satisfacción laboral

DevOps promueve un ambiente de trabajo basado en el rendimiento, en contraposición con una cultura basada en reglas, estatus o poder. Esto reduce los obstáculos burocráticos y fomenta la responsabilidad compartida entre todos los

miembros. El resultado es un equipo de trabajo más satisfecho y productivo, que ayuda a impulsar el rendimiento del negocio. Los desarrolladores y los ingenieros de operaciones, generalmente, prefieren un entorno DevOps porque pueden trabajar de manera más eficiente y cambiar de roles cuando sea necesario. A su vez, tienen una mejor comprensión sobre cómo impacta su función dentro de TI y dentro de la organización en su conjunto.

La entrega rápida de software es crucial en la era digital actual y la cultura de DevOps es el motor de este proceso. En otras palabras, permite a las empresas acelerar la comercialización de sus servicios y productos al mercado y desplegar nuevas funciones, de manera rápida y eficiente. La adopción de DevOps no es un proceso simple, pero cuando se realiza correctamente, la inversión genera dividendos que duplican el esfuerzo inicial.

## 1.3. Principios DevOps

En el corazón de los principios DevOps se encuentra el **aprendizaje colaborativo** y las **relaciones de colaboración** entre desarrollo y operaciones. Estos se centran en aumentar el ritmo de trabajo de forma planificada para obtener despliegues e implementaciones más frecuentes, al tiempo que mejora la estabilidad, la resistencia y la seguridad del entorno de producción. Para que una organización pueda abrazar los principios de DevOps, debe reforzar y enfatizar este enfoque holístico en todas sus áreas, y no únicamente en los departamentos de desarrollo y operaciones.

Estos principios fundamentales pueden resumirse en la siguiente lista:

### Fomentar un ambiente de trabajo colaborativo

Como ya hemos comentado, la esencia de la metodología DevOps es el trabajo conjunto entre los equipos de desarrollo y operaciones, a fin de crear una sinergia que se centre en la consecución de objetivos comunes. Para lograr este objetivo, las organizaciones necesitan fomentar una comunicación eficiente y permitir que ambos equipos compartan ideas y resuelvan problemas de forma conjunta. Esta «rotura de los silos» permite a las empresas alinear a sus trabajadores, procesos y herramientas hacia un enfoque centrado en el cliente.

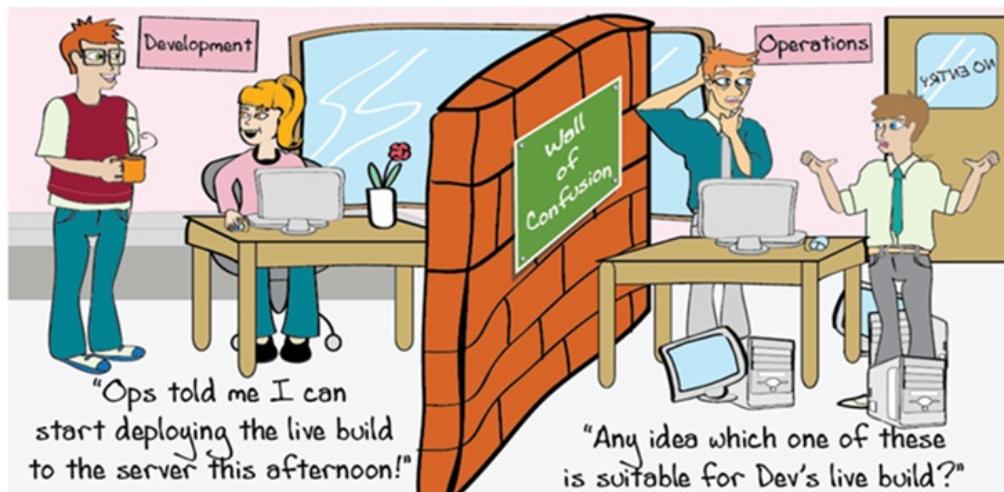


Figura 1. What is DevOps? Fuente: Buehring, 2020.

## Establecer una responsabilidad de extremo a extremo

En el modelo tradicional de desarrollo de software, desarrollo y operaciones tenían roles separados. Pero en un entorno DevOps, ambos grupos trabajan como un equipo que es totalmente responsable de la aplicación, de principio a fin. Tradicionalmente, los desarrolladores escribían código y operaciones desplegaba ese código, pero eso producía todo tipo de ineficiencias, problemas de rendimiento y entornos impredecibles. Todos los miembros del equipo DevOps son responsables de velar por la calidad y fiabilidad de las aplicaciones que desarrollan.

## Fomentar la mejora continua

La responsabilidad de extremo a extremo involucra a todas las áreas de la organización, ya que estas deben adaptarse continuamente a las circunstancias cambiantes (ya sea el surgimiento de nuevas tecnologías, las nuevas y variadas necesidades de los clientes, o los cambios en la legislación). DevOps pone énfasis en la mejora continua para optimizar el rendimiento, los costes y la velocidad de entrega.

## Fomentar la automatización

Para hacer frente a los numerosos ciclos de desarrollo y responder de inmediato a los comentarios de los clientes, las organizaciones deben implementar procesos automatizados. Afortunadamente, en los últimos años han surgido variadas y numerosas herramientas de automatización que agilizan la gestión de procesos en gran medida.

## Dar prioridad a las necesidades del cliente

DevOps requiere que las organizaciones tengan la misma flexibilidad que las *start-up lean*, ya que estas pueden innovar continuamente, cambiar de estrategia cuando sea necesario e invertir en la creación de nuevas características (*features*) para aumentar la satisfacción de sus clientes. Los equipos de DevOps deben estar al día de todo lo que ocurre en el mercado para satisfacer constantemente las necesidades y demandas cambiantes de los consumidores. Los datos recopilados de los procesos automatizados deben revisarse con frecuencia para garantizar que se cumplen los objetivos de rendimiento establecidos por el área de negocio.

## Aceptar el fracaso y aprender de él

La aceptación del fracaso fomenta un clima de aprendizaje que tendrá un **impacto positivo en la cultura organizacional**. Cuando los equipos se sienten psicológicamente seguros para actuar y tienen la capacidad de transformar radicalmente su trabajo, pueden incurrir en fallos. Resulta vital convertir esos fallos en oportunidades de aprendizaje.

## Formar equipos multifuncionales

Es necesario que los equipos de DevOps se involucren en cada etapa del ciclo de vida del desarrollo de software, desde la planificación, la construcción, la implementación, y la retroalimentación hasta la mejora continua. Para ello, es necesario que el equipo sea multifuncional y equilibrado, y que cada miembro posea un conjunto habilidades idóneas para su rol en TI.

Además de estos principios, existen otros que se centran, más específicamente, en el área de TI y que abogan por una mejora integral de sus procesos. Sin embargo, tienen un enfoque holístico para DevOps y organizaciones de todos los tamaños pueden adoptarlos. Estos principios son:

- ▶ **Trabajar en entornos similares a producción durante el desarrollo del software y la ejecución de pruebas.** Hace alusión al concepto DevOps de «desviación a la izquierda» o *shift left*. El objetivo que persigue este principio es el de permitir que los equipos de desarrollo y calidad trabajen en entornos similares a producción a fin de que puedan verificar el comportamiento y desempeño de la aplicación mucho antes de que esté lista para su despliegue. Desde una perspectiva de operaciones, este principio tiene un enorme valor porque permite que el equipo pueda ver, desde una fase temprana del ciclo, cómo se comportará su entorno cuando soporte a la aplicación.
- ▶ **Realizar despliegues mediante procesos repetibles y fiables.** Permite que el área de operaciones dé soporte a un proceso de desarrollo de software ágil e iterativo, durante todas las fases del ciclo de vida que atraviesa el código. La automatización es esencial para crear procesos que sean iterativos, frecuentes, repetibles y fiables, por lo que la organización debe crear una fuente o suministro de entregas que permita que el despliegue sea continuo, automatizado y validado a través de pruebas. Los despliegues frecuentes también permiten a los equipos poner a prueba sus propios procesos y, a largo plazo, hace que se reduzca el riesgo de fallos.
- ▶ **Supervisar y validar la calidad operativa.** Pone énfasis en la monitorización del código desde una fase temprana, al exigir que el código sea validado a través de pruebas automatizadas al comienzo del desarrollo y con una alta frecuencia, a fin de controlar las características funcionales y no funcionales de la aplicación. Esta supervisión frecuente proporciona una alerta temprana sobre cuestiones operativas o de calidad, que puedan ocurrir más tarde en producción.

- ▶ **Amplificar los bucles de retroalimentación.** Exige a las organizaciones la creación de canales de comunicación eficaces que permitan a todas las partes interesadas acceder y participar en un ciclo comunicativo. De ello se desprende que el desarrollo de código tendrá el foco puesto en las prioridades del proyecto y podrá ajustarse a los requisitos del proyecto si estos permanecen. También implica que los pasos a producción se harán mediante la mejora de los entornos de producción y que los planes de entrega se modificarán en pos del negocio.

## 1.4. Metodologías y estrategias DevOps

DevOps tuvo sus orígenes en Agile. Para comprender mejor esta afirmación, es importante recordar que Agile promueve el desarrollo de software a través de iteraciones cortas, la entrega continua de nuevas características, así como correcciones de errores en ciclos rápidos, de entre dos a cuatro semanas. Por su parte, DevOps elimina los silos que separan a los equipos de desarrollo y operaciones con el objetivo de disminuir el tiempo de respuesta a sus clientes, y garantizar la entrega continua de software.

DevOps no solo involucra a todas las áreas de una organización en el proceso de desarrollo (líneas de negocio, proveedores involucrados en la entrega de software y propios consumidores), sino que además lo hace de una manera en la que acelera el desarrollo y mejora la calidad de todos sus procesos. Esto conduce a la creación de una cultura de la innovación, que permite a las organizaciones colaborar y reaccionar con agilidad ante los cambios del mercado.

Las metodologías de DevOps incluyen:

- ▶ Integración continua, que incluye a la creación de código, la construcción, la integración y las pruebas.
- ▶ Entrega continua, que incluye a la integración continua, pero se centra en las entregas.
- ▶ Despliegue continuo, que se centra en automatizar las entregas de la forma más rápida posible.
- ▶ Gestión de la configuración y monitorización continua.

## Estrategia DevOps

Como hemos visto, DevOps se basa en el concepto de continuidad. No es una casualidad que hayamos mencionado el concepto de integración, entrega y despliegue continuos. La estrategia de DevOps se centra en la capacidad empresarial para la entrega continua de software, que permite a los clientes aprovechar las oportunidades del mercado y, a la vez, reducir el tiempo de retroalimentación de estos.

Permite a los equipos trabajar de forma conjunta y esforzarse en  
solucionar problemas que puedan aparecer en cualquier parte del ciclo  
de vida del desarrollo de software. El cliente representa la prioridad la  
máxima de estos equipos y su bienestar es una responsabilidad  
compartida entre todos.

La estrategia DevOps acelera la entrega de software, ayuda a equilibrar la velocidad, los costes, la calidad y el riesgo, y aporta una mayor capacidad para innovar. Además, reduce el tiempo de retroalimentación de los clientes, lo que ayuda a mejorar su experiencia.

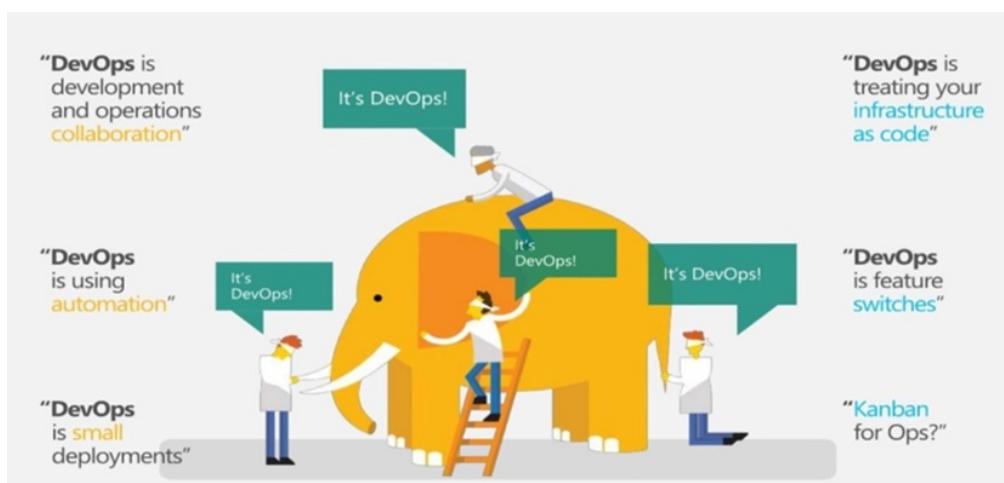


Figura 2. Cómo aplicar la cultura DevOps a tu empresa. Fuente: Bravent IT Consulting, 2019.



## 1.5. Las capacidades de DevOps

Las capacidades que conforman DevOps son un conjunto amplio que abarca todo el ciclo de vida de entrega del software. Cuando una organización comienza a trabajar con DevOps depende de sus objetivos y metas de negocio (cuáles son los desafíos que está intentando abordar y cuáles son los puntos de mejora en sus capacidades de entrega de software). DevOps tiene una arquitectura de referencia, es decir, un patrón o un conjunto de métodos y capacidades determinadas.

Estas capacidades son proporcionadas por un conjunto idóneo de personas, prácticas definidas y herramientas de automatización. La arquitectura de referencia de DevOps plantea los siguientes senderos o caminos de adopción:

- ▶ Planificación y medición.
- ▶ Desarrollo y pruebas.
- ▶ Lanzamientos y despliegues continuos.
- ▶ Supervisión y optimización.

### Planificación y medición

Este camino consiste en una práctica que se centra en las líneas de negocio y sus procesos de planificación: la planificación continua del negocio. Como ya hemos mencionado, las empresas tienen que ser ágiles y deben ser capaces de reaccionar rápidamente ante las necesidades y comentarios de los usuarios. En consecuencia, muchas empresas, hoy en día, emplean **técnicas de pensamiento Lean**. Estas técnicas comienzan a dar forma a sus proyectos a través de la identificación de los resultados deseados y los recursos necesarios para cumplir con sus metas de negocio y, luego, en base a la retroalimentación que reciben de sus clientes se adaptan y ajustan, según sea conveniente. En otras palabras, las organizaciones

modifican el curso de sus planes de negocio, lo que les permite tomar decisiones *trade-off* continuas en un entorno de recursos limitados.

## Desarrollo y pruebas

Este camino implica la adopción de dos prácticas: desarrollo colaborativo y pruebas continuas. Como tal, forma el núcleo de las capacidades de desarrollo y de garantía de calidad (en siglas, QA).

La entrega de software en una empresa involucra a una gran cantidad de equipos multifuncionales, que incluye a: los propietarios de las líneas de negocio, los analistas de negocios, arquitectos de software, desarrolladores, profesionales de control de calidad, el personal de operaciones, especialistas en seguridad, proveedores y socios. Los profesionales de estos equipos trabajan en múltiples plataformas y pueden estar dispersos en múltiples localizaciones. El desarrollo colaborativo permite que estos profesionales trabajen juntos y proporciona, así, un conjunto común de prácticas y una plataforma común que se puede utilizar para crear y desplegar software.

Una capacidad básica, incluida dentro del desarrollo de colaborativo, es la integración continua (una práctica en la que los desarrolladores de software integran, de forma frecuente, su trabajo con el de otros miembros del equipo de desarrollo). La integración continua se hizo popular gracias al **movimiento ágil (o Agile)**. Esta filosofía de trabajo tiene como objetivo que los desarrolladores integren regularmente su trabajo con el del resto de los desarrolladores de su equipo y luego ejecuten pruebas para comprobar que dicha integración ha sido exitosa.

En el caso de los sistemas complejos integrados por múltiples sistemas o servicios, los desarrolladores también integran regularmente su trabajo con otros sistemas y servicios. Esta integración, llevada a cabo con regularidad, es fundamental para la detección temprana de fallos y posibles riesgos derivados. Es necesario tener presente que, en los sistemas complejos, los principales riesgos se asocian a

problemas técnicos y, al tiempo, a una mala gestión en la planificación de las entregas.

Las pruebas sobre el código adquieren una especial relevancia debido a la práctica de la integración continua. Esta persigue varios objetivos:



Figura 3. Los objetivos de la integración continua. Fuente: elaboración propia.

Cuando hablamos de pruebas continuas, nos referimos a comenzar con la fase de pruebas desde el inicio del desarrollo y, de forma recurrente, a través del ciclo de vida de la aplicación. Esto da como resultado una reducción de costes y el acortamiento de los ciclos de *testing*, pero también sirve como retroalimentación de la calidad operativa. Estos objetivos son alcanzables mediante la implementación de pruebas automatizadas y la virtualización de servicios. Como ya hemos mencionado en otro apartado, el trabajo en entornos similares a producción hace factible la realización de pruebas continuas a nuestro código.

## Lanzamientos y despliegues continuos

**La entrega y el despliegue continuos** llevan el concepto de integración continua a otro nivel. La entrega continua permite a los desarrolladores automatizar las pruebas unitarias y verificar actualizaciones en las aplicaciones, antes de enviarlas a los clientes. Estas pruebas pueden ser sobre la interfaz de usuario o *user interface* (en

siglas, UI) de carga, de integración, de fiabilidad de la *Application Programming Interfaces* (en siglas, API), etc. De este modo, los desarrolladores pueden validar las actualizaciones de forma más exhaustiva y descubrir problemas por anticipado. Con la nube, resulta sencillo y rentable automatizar la creación y replicación de varios entornos de pruebas, algo que anteriormente era complicado en las instalaciones.

Una definición muy acertada es la que da Jez Humble (2010) en el libro *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*:

«La entrega continua es la habilidad de facilitar cambios de todo tipo (incluyendo nuevas características, cambios de configuración, soluciones de bugs y experimentos) a producción, o a los usuarios, de forma rápida, segura y sostenible».

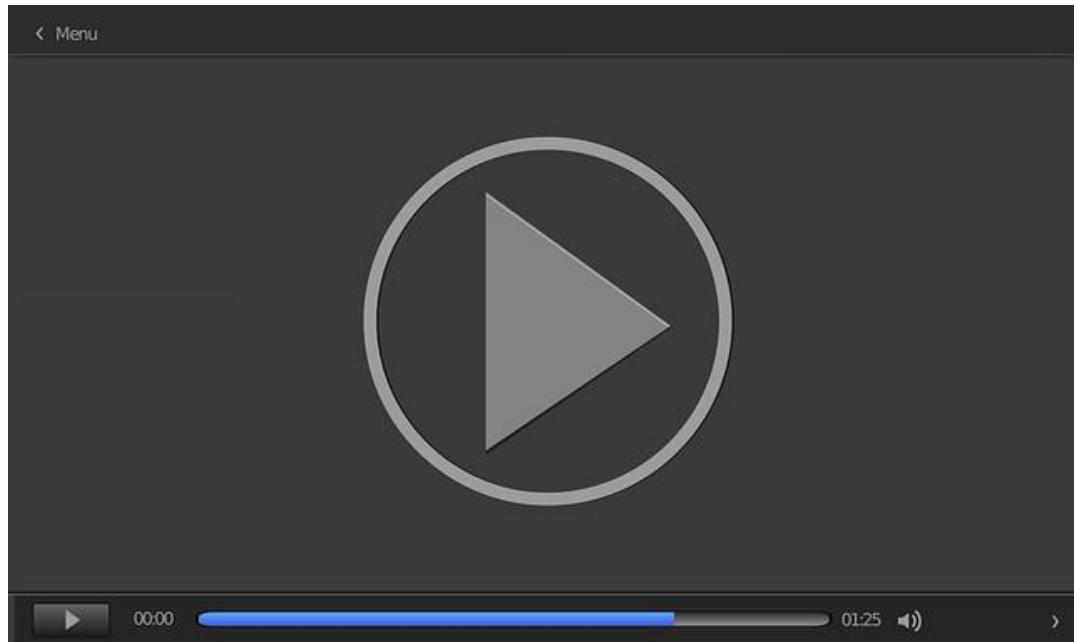
## Supervisión y optimización

La ruta de adopción, a través de la supervisión y optimización, permite a las empresas controlar cómo las aplicaciones están comportándose en producción y recibir retroalimentación de los clientes. La supervisión continua proporciona información y métricas sobre las aplicaciones durante las diferentes etapas del ciclo de entrega del software a los departamentos de operaciones, QA, desarrollo, el personal de las líneas de negocio y otras partes interesadas.

Las nuevas tecnologías permiten a las empresas capturar el comportamiento y los problemas que experimentan los clientes que utilizan la aplicación. Esta retroalimentación permite que las diferentes partes interesadas (accionistas o jefes de proyecto, por ejemplo) tomen medidas para mejorar las aplicaciones y la experiencia del cliente. Con esta información, las líneas de negocio podrán ajustar su estrategia, el departamento de desarrollo podrá ajustar las características que entrega y el departamento de operaciones podrá mejorar el entorno en el que se despliega la aplicación.

## Madurez en el desarrollo de software

Antes de continuar con el siguiente apartado, te invito a que veas la píldora *Modelos de madurez en el desarrollo del software*, donde analizaremos los diferentes niveles de madurez que han sido definidos por la industria y su conexión con DevOps.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=639043f1-f3c8-4cc9-a1e0-ad8d0104e260>

---

## 1.6. Mitos DevOps

El movimiento DevOps es joven y todavía emergente, especialmente en las organizaciones como conjunto. Al igual que cualquier nuevo movimiento o tendencia, este ha despertado mitos y falacias. Lo cierto es que algunos de estos mitos se han originado en empresas que probaron, sin éxito, la adopción de DevOps. A continuación, nombraremos algunos de ellos.

### **La metodología DevOps solo es adecuada para las organizaciones nacidas en la web**

Lo que generalmente se conoce como DevOps se originó en las empresas nacidas en la web, es decir, aquellas empresas que se originaron en Internet como Etsy, Netflix y Flickr. Sin embargo, empresas de gran renombre han estado implementando los principios y prácticas alineadas con DevOps para realizar entregas de software durante décadas. Por otra parte, **los principios DevOps actuales tienen un nivel de madurez que los hace aplicables a todo tipo de organizaciones que tengan tecnologías multiplataforma y equipos distribuidos.**

### **DevOps es operaciones aprendiendo a codificar**

Los equipos de operaciones siempre han gestionado los entornos y tareas repetitivas a través de *scripts*, pero con la evolución de la infraestructura como código, vieron la necesidad de gestionar estas grandes cantidades de código con prácticas de ingeniería del software como el control de código de versiones, o *el check-in y check out*, por ejemplo. Hoy en día, una nueva versión del entorno se lleva a cabo mediante una nueva versión del código que lo define. Esto no significa, sin embargo, que los equipos de operaciones necesiten aprender a codificar en Java o C#. La mayoría de las tecnologías de infraestructura como código utilizan idiomas como Ruby, que es relativamente fácil, especialmente para las personas que tienen experiencia con secuencias de comandos.

## DevOps solo involucra a las áreas de desarrollo y operaciones

Aunque este mito sugiere que DevOps únicamente involucra o afecta a los equipos de desarrollo y operaciones, como hemos visto, **la adopción de DevOps requiere que toda la organización se comprometa en la consecución de los objetivos.** Todas las partes interesadas en la entrega software —líneas de negocio, profesionales, ejecutivos, socios, proveedores, etc.— deben participar y colaborar en un esfuerzo conjunto.

## La metodología DevOps no es adecuada para organizaciones ITIL

Algunas personas temen que las capacidades de DevOps, tales como la entrega continua, sean incompatibles con los controles y procesos prescritos por la *Information Technology Infrastructure Library* (en lo sucesivo denominado ITIL, por sus siglas en inglés), un conjunto documentado de prácticas recomendadas (del inglés *best practices*) para los servicios de gestión de TI. En realidad, **el modelo del ciclo de vida ITIL es compatible con DevOps y lo mismo ocurre con sus principios (se alinean muy bien con los de DevOps).** La ITIL, sin embargo, ha tenido una mala reputación en algunas organizaciones debido a que se ha implementado, en su mayor parte, en las tradicionales metodologías en cascada que, como sabemos, se caracterizan por tener procesos lentos que no admiten cambios ni mejoras rápidas.

## La metodología DevOps no es apta para industrias reguladas

Las industrias reguladas necesitan contar con procedimientos y prácticas fiables, comprobaciones y balances que aseguren el cumplimiento de las normas y sean susceptibles de superar auditorías con éxito. **La adopción de DevOps mejora el cumplimiento de estos requisitos, si se hace correctamente.** La automatización del flujo de procesos y el uso de herramientas de software, que tienen la capacidad de monitorizar y hacer seguimiento de la información, pueden ser un gran aliado a la hora de recoger los datos o paquetes de datos necesarios requeridos en una auditoría.

## DevOps no es compatible con equipos de desarrollo externalizado

Los equipos de outsourcing deben ser vistos como proveedores o prestadores de una capacidad en el canal de entrega de DevOps. Es por ello por lo que **las organizaciones deben garantizar que las prácticas y procesos de estos equipos proveedores sean compatibles con los de sus propios equipos internos.** La planificación de entregas conjuntas, la gestión de los elementos de trabajo y las herramientas adecuadas mejoran significativamente la comunicación y la colaboración entre las líneas de negocio, los proveedores y los equipos de proyectos, abriendo paso a la implementación de las prácticas DevOps. El uso de herramientas de gestión de entrega de aplicaciones puede mejorar significativamente la capacidad de una organización para definir y coordinar todo el proceso de entrega a través de todos los participantes.

## Si no hay nube, no hay DevOps

Cuando se piensa en DevOps, a menudo se lo asocia a la nube por su capacidad de proveer dinámicamente recursos de infraestructura para desarrolladores y *testers* (trabajadores dedicados a la realización de pruebas) a fin de que obtengan rápidamente entornos de prueba sin tener que esperar días o semanas. Sin embargo, **la nube no es un requisito necesario para adoptar las prácticas**

**DevOps**, siempre y cuando la organización tenga procesos eficientes para la obtención de recursos, despliegues y pruebas. La virtualización en sí misma es opcional.

## DevOps no es aplicable a sistemas grandes y complejos

Los sistemas complejos requieren del mismo nivel de disciplina y colaboración que ofrece DevOps, ya que dichos sistemas suelen tener varios componentes de software y, cada uno de ellos, a su vez, tiene sus propios ciclos y plazos de entrega.

**DevOps facilita la coordinación de estos ciclos y planificación de entregas, a nivel de sistema.**

## DevOps es un cambio en la comunicación

Algunos miembros de la comunidad DevOps han acuñado términos graciosos como ChatOps o HugOps, dando a entender que DevOps solo se centra en la colaboración y la comunicación. Como hemos visto, **la idea de que la comunicación y la colaboración resuelven todos los problemas es errónea**. Si bien DevOps se apoya en la comunicación, si esta viene acompañada de procesos desorganizados, equipos disfuncionales, o malas inversiones, no conduce a mejores resultados ni a la obtención de beneficios.

## DevOps implica realizar cambios y despliegues a diario

Esta idea errónea proviene de las organizaciones que solo despliegan aplicaciones en la web. Algunas de estas empresas afirman con orgullo, en sus sitios web, que realizan despliegues en producción a diario. Esta práctica, sin embargo, no resulta en absoluto provechosa en organizaciones de gran tamaño que deseen desplegar aplicaciones complejas, sino que incluso también puede resultar imposible debido a ciertas restricciones regulatorias o a la política interna de la empresa. **La adopción de DevOps permite a las organizaciones hacer un *push* a producción cuando sea justificado y necesario, y no se rige por una fecha concreta marcada en un calendario.**



## 1.7. Referencias bibliográficas

Bravent. (2019, junio 3). *Cómo aplicar la cultura DevOps a tu empresa.*

<https://www.bravent.net/como-aplicar-la-cultura-devops-a-tu-empresa/>

Buhering, S. (2020, marzo 20). *What is DevOps: Free ebook.* Knowledge Train.

<https://www.knowledgetrain.co.uk/it/devops/what-is-devops>

Puppet. (2015). 2015 State of DevOps Report.

<https://puppet.com/resources/report/2015-state-devops-report/>

Humble, J. y Farley, D. (2010). *Continuos Delivery: Reliable Software Releases through Build, Test, and Deployment Automation.* Addison-Wesley Professional.

## ¿Qué tareas hace un profesional de DevOps?

Fundación Telefónica. (2018, noviembre 26). *¿Qué tareas hace un profesional de DevOps? | #ConectaEmpleo [Vídeo]*. YouTube. <https://www.youtube.com/watch?v=MagivhkTrGc>

Este breve vídeo es un resumen de las tareas que desempeña un profesional de DevOps en una organización. Te animo a que lo visualices para que puedas comprender cómo se aplican los contenidos que hemos revisado en este tema a la vida real de un trabajador del sector en España.

## What is DevOps?

IBM Cloud. (2019, septiembre 9). *What is DevOps?* [Vídeo]. YouTube.

<https://youtu.be/UbtB4sMaaNM>

Andrea Crawford, ingeniera y CTO (*Chief Technology Officer*) de DevOps en IBM, explica con sus propias palabras qué es DevOps, cuál es valor que aporta DevOps a la organización y cómo las prácticas y herramientas de DevOps intervienen en el ciclo de vida de una aplicación.

1. ¿Cuál de las siguientes prácticas se corresponde con las metodologías DevOps?

- A. Integración continua, entrega continua, gestión de la configuración y monitorización.
- B. Atraer y contratar a más trabajadores para contar con una fuerza de trabajo más fuerte y numerosa.
- C. Rediseñar la estrategia de marketing para atraer a una mayor cantidad de clientes.
- D. Reestablecer los objetivos de negocio para aumentar la facturación y extender las directrices necesarias al resto de la organización para alcanzar esta meta.

2. ¿Cuáles de las siguientes afirmaciones acerca de DevOps son falsas?

- A. La filosofía, cultura y herramientas DevOps son exclusivas para las *start-ups*.
- B. Para la adopción de DevOps, solo es necesario que las áreas de desarrollo y operaciones trabajen en equipo, mientras el resto de la organización persigue sus objetivos individuales.
- C. Para la implementación de DevOps es necesario que se haga un *push* a producción a diario.
- D. Todas las anteriores.

- 3.** ¿Qué objetivos persigue la estrategia DevOps?

  - A. Aumentar la capacidad de una organización para entregar aplicaciones y servicios a alta velocidad.
  - B. Mejorar sus productos de forma continua a un ritmo más rápido que las organizaciones tradicionales.
  - C. Ofrecer un mejor servicio a sus clientes y competir en el mercado de forma más eficiente.
  - D. Todas las anteriores.
- 4.** ¿Cuáles de los siguientes enunciados representan beneficios directos de la implementación de DevOps en una organización?

  - A. Impacto positivo en la cuenta de resultados debido a un aumento de la facturación anual.
  - B. Aumento de la satisfacción laboral.
  - C. Aceleración de la innovación.
  - D. B y C son correctas
- 5.** ¿Cuáles son los principios fundamentales de DevOps?

  - A. La colaboración, mejora continua, responsabilidad equitativa, automatización de procesos, foco en el cliente, equipos multifuncionales y el aprendizaje en base a errores.
  - B. La colaboración y la automatización.
  - C. La gestión de la configuración, la automatización y la colaboración.
  - D. La integración y entrega continuas.

6. ¿Cuál definición de DevOps es la más acertada?

- A. DevOps es una combinación de filosofías, prácticas y herramientas culturales que utiliza software ágil para mejorar, gestionar y automatizar sus procesos.
- B. DevOps es una metodología basada en la tecnología.
- C. DevOps es un enfoque basado en la metodología en cascada que aboga por la cooperación e interacción entre los miembros de los equipos IT.
- D. DevOps es una mejora en la comunicación entre los equipos.

7. ¿Qué impulsa a las empresas hacia la adopción de DevOps?

- A. La reducción del *time-to-market*.
- B. La mejora de procesos a través de la automatización.
- C. La mejora de la comunicación, la interacción y la colaboración entre los equipos.
- D. Todas las anteriores.

8. ¿Cuál de los siguientes enunciados es correcto?

- A. Los equipos de operaciones deben identificar y resolver los problemas de forma temprana mediante la monitorización.
- B. La monitorización es opcional para el enfoque DevOps, solo la virtualización es obligatoria y necesaria.
- C. La virtualización en sí misma es opcional para implementar DevOps, pero reportará enormes beneficios en caso de que se integre.
- D. A y C son correctas.

**9.** Termina la frase: «Las herramientas automatizadas...»

- A. Ofrecen oportunidades adicionales para mejorar la eficiencia.
- B. Ayudan a acortar los ciclos de desarrollo.
- C. Aceleran la entrega del producto.
- D. Todas las anteriores.

**10.** El término *shift left* hace alusión a...

- A. El trabajo en entornos similares a producción durante el desarrollo del software y la ejecución de pruebas.
- B. La automatización de las pruebas antes del pase a producción.
- C. El trabajo denominado *pair programming* (programación en parejas) que llevan a cabo los desarrolladores en ocasiones.
- D. Ninguna de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 2. Cultura de la colaboración

# Índice

[Esquema](#)

[Ideas clave](#)

[2.1. Introducción y objetivos](#)

[2.2. Introducción a la colaboración en DevOps](#)

[2.3. Herramientas que favorecen la colaboración](#)

[2.4. Factores que aceleran la adopción de DevOps](#)

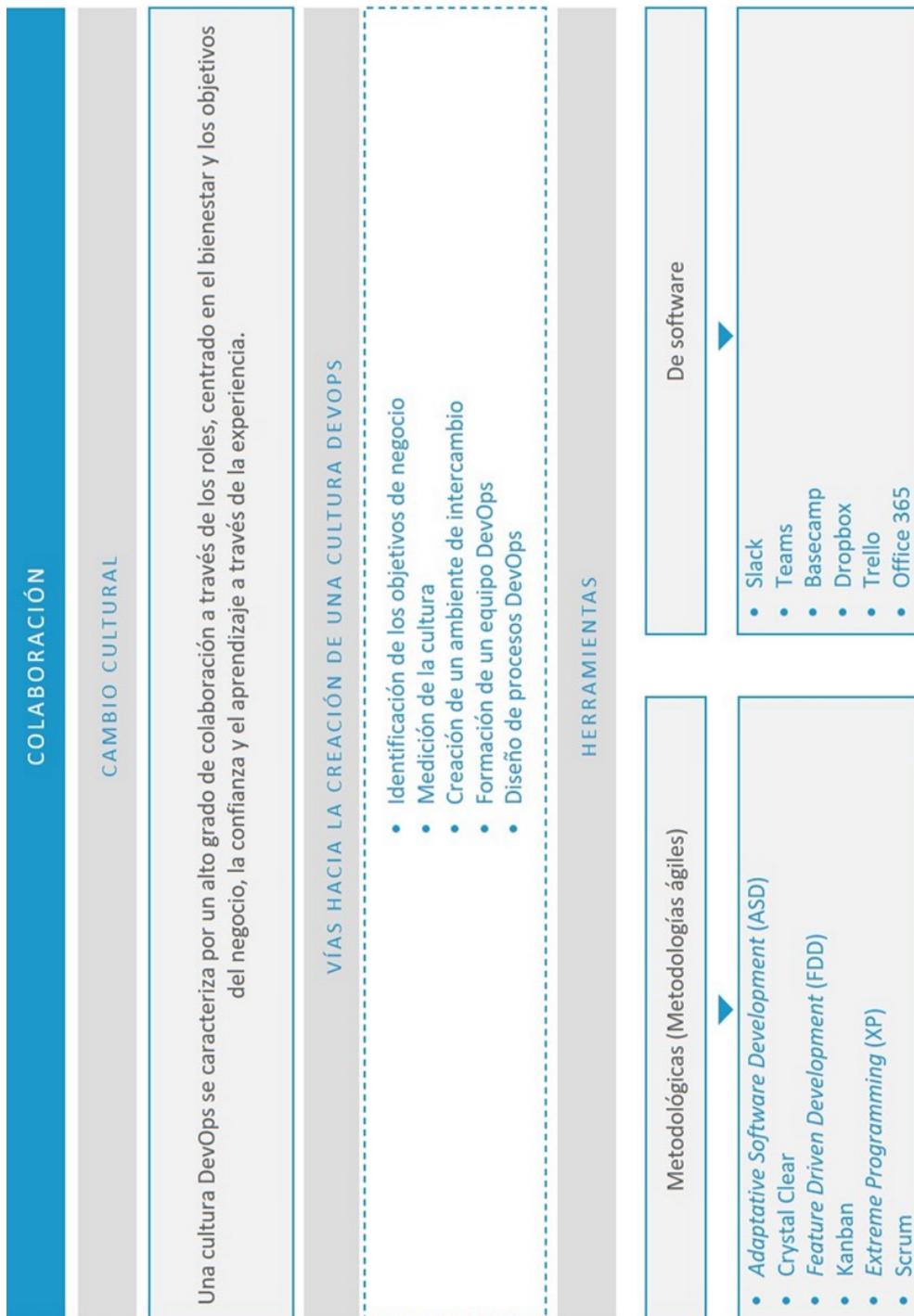
[2.5. Referencias bibliográficas](#)

[A fondo](#)

[La cultura DevOps](#)

[Google Cloud DevOps: Speed With Reliability and Security](#)

[Test](#)



## 2.1. Introducción y objetivos

Por lo general, solemos utilizar el término **cambio cultural** para implantar procesos de mejora continua, emplear metodologías ágiles y adoptar el enfoque DevOps. Esto obedece a la imposibilidad de adoptar las transformaciones de forma superficial. Tal y como hemos visto con anterioridad, esta metodología exige un cambio en la gestión de los recursos, la organización de los equipos y también en la forma de pensar. **Es necesario pasar del control al autoservicio, del miedo a la visibilidad, de los silos a la colaboración abierta**, y ninguno de estos cambios se puede imponer, de un cambio organizativo muy beneficioso a la vez que radical.

Si bien son las grandes organizaciones las que podrán beneficiarse de estos cambios, de forma más evidente, en términos cuantitativos, son las nuevas organizaciones las que tienen la posibilidad de abrazar a la cultura DevOps y de la colaboración de una forma natural. En la mayoría de las *start-ups*, la cultura de la colaboración, la visibilidad en contraposición con el control, o la automatización en oposición al trabajo manual surgen de forma natural por la necesidad de optimizar los recursos. Otro caso típico, donde podemos ver la afinidad de las nuevas organizaciones con la colaboración y la cultura DevOps, es el *Cloud First*: organizaciones donde no se plantea una transición a la nube porque nunca ha existido otra opción. Algunas organizaciones tradicionales han aceptado (mediante la resignación) que son incapaces de innovar de la forma en la que le gustaría debido a su propia inercia e intentan aprender de estas nuevas organizaciones y utilizarlas como herramientas.

En este tema se pretenden conseguir los siguientes objetivos:

- ▶ Comprender qué implica el cambio cultural de una organización para la adopción de la metodología DevOps.

- ▶ Conocer las herramientas de colaboración que ayudan a los equipos DevOps a realizar su trabajo de forma exitosa.
- ▶ Analizar los factores que promueven la adopción de DevOps.

## 2.2. Introducción a la colaboración en DevOps

Si quisieramos escribir un decálogo para DevOps, probablemente uno de los elementos más importantes sería la colaboración. Recordemos que DevOps integra dos mundos: el del desarrollo y el de las operaciones. Estos individuos adquieren conocimientos y capacidades de ambos entornos, y se abren, por tanto, canales de comunicación dentro de las organizaciones que permiten alcanzar una mayor agilidad, capacidad requerida para la Integración continua, la entrega continua y los frecuentes *releases* (lanzamientos).

**Desde sus cimientos, DevOps es un movimiento cultural: involucra a las personas.** Una organización puede adoptar los procesos más eficientes o tantas herramientas automatizadas como le sea posible, pero todas ellas serán inútiles si no están integradas con las personas que deben ejecutar esos procesos y utilizar esas herramientas. **En la construcción de una cultura DevOps, por lo tanto, está la clave de la adopción DevOps.**

Una cultura DevOps se caracteriza por un alto grado de colaboración a través de los roles, centrado en el bienestar y los objetivos del negocio (en lugar de los objetivos exclusivos de cada departamento), la confianza y el aprendizaje a través de la experiencia. La construcción de una cultura no es como la adopción de un proceso o una herramienta, ya que requiere de cierta la ingeniería social de los equipos y cada uno está compuesto por personas que tienen predisposición, experiencia y prejuicios propios. Esta diversidad puede hacer que la construcción de la cultura se convierta en un proceso extremadamente desafiante y difícil.

Las prácticas Lean y Agile son el núcleo de DevOps y, si la organización ya ha aplicado alguno de estos métodos, podrán apalancar la transición hacia una cultura DevOps.

## Identificación de los objetivos de negocio

La primera tarea en la creación de una cultura es conseguir que todos vayan en la misma dirección y trabajen por el mismo objetivo, es decir, la identificación de los objetivos de negocio comunes para el equipo y la organización en su conjunto. Cuando el equipo sabe cuál es el objetivo común por el que está trabajando y cómo se medirá el progreso hacia ese objetivo, disminuyen los riesgos y el miedo individual y colectivo.

**DevOps no es el objetivo final de una organización, sino que es una herramienta que permite alcanzar los objetivos propuestos.** Actualmente, DevOps debe abordar nuevos retos empresariales, por lo que la organización puede usar esos desafíos como punto de partida para identificar los objetivos que se desean alcanzar y, entonces, desarrollar un conjunto de hitos o peldaños que permitan alcanzar esos objetivos y que los equipos utilicen como guía.

## Medición de la cultura

La medición de la cultura es extremadamente difícil. ¿Cómo se mide con precisión la mejora de la colaboración o de la moral de los equipos? Se podría plantear una medición directa de las actitudes y valores de un equipo mediante encuestas pero las encuestas pueden tener un alto índice de error estadístico, ya que suelen ser, por lo general, pequeñas. A la inversa, se puede tomar una medida indirecta mediante el seguimiento de la frecuencia con la que un miembro del equipo de desarrollo contacta a un miembro del equipo de operaciones o de *Quality Assurance* (en siglas, QA) para colaborar en la resolución de un problema sin tener que pasar por los canales oficiales o sin que se lo ordene un superior. La colaboración y la comunicación entre las personas y los equipos es la esencia de la cultura de DevOps.

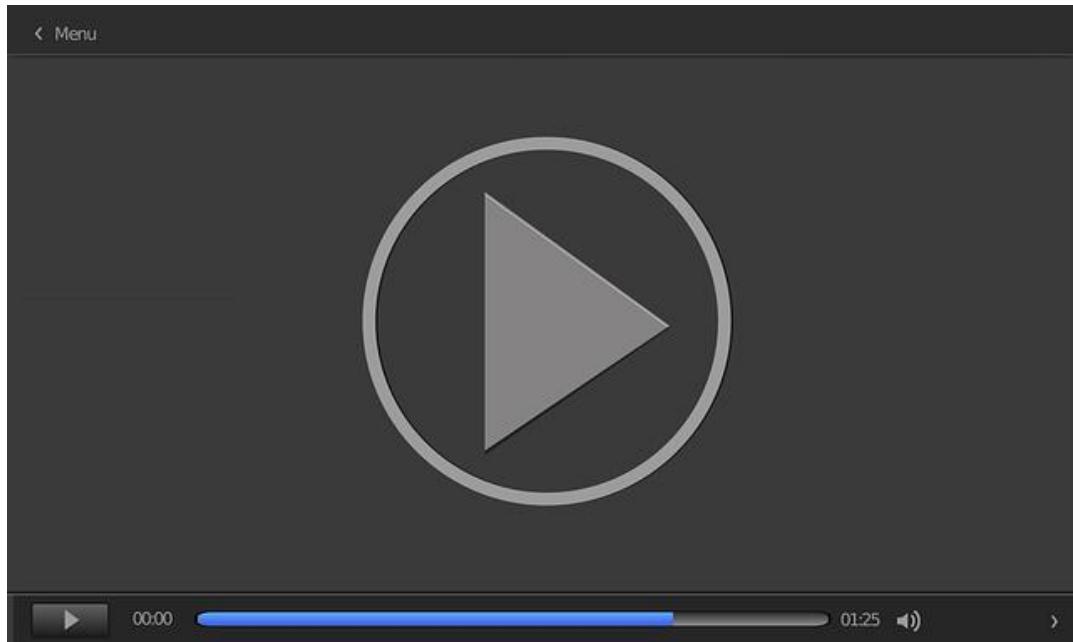
## Creación de un ambiente de intercambio

Después de identificar los objetivos comunes de negocio, el siguiente paso en la construcción de una cultura DevOps es que los líderes de la organización trabajen con sus equipos para crear un entorno de colaboración e intercambio. Los líderes deben velar por la eliminación de todas las barreras autoimpuestas para que haya cooperación. Por lo general, se suele valorar el desempeño de los equipos de operaciones a través del *uptime* (tiempo de actividad sin caída) y la estabilidad, y a los equipos de desarrollo por las nuevas características (*features*) entregadas.

La realidad es que esta práctica enfrenta a estos dos grupos: operaciones (sabe que la mejor manera de protegerse es no aceptar ningún cambio) y desarrollo (tiene poco o ningún incentivo para centrarse en QA). DevOps elimina esta fricción entre ambas áreas, valorando el éxito desde la concepción de una responsabilidad compartida en la entrega de nuevas capacidades y *features*, de forma rápida y segura.

Los líderes de la organización deben fomentar la colaboración mediante la mejora de la visibilidad. **El establecimiento de un conjunto común de herramientas de colaboración es esencial**, especialmente cuando los equipos están localizados en diferentes puntos del globo y no tienen la posibilidad de trabajar juntos en el mismo espacio físico. Resulta crucial dar **visibilidad** a todas las partes involucradas en un proyecto sobre cuáles son los **objetivos y el estado de un proyecto**. Esto facilitará la construcción de una cultura DevOps basada en la confianza y la colaboración. Este cambio en la dinámica de grupos y el traspaso de la información tendrán un impacto a nivel humano y es probable que algunas personas deban ser reasignadas a otros proyectos mientras se esté adoptando esta nueva cultura.

Antes de continuar, te invito a que veas esta píldora, *Modelos de licenciamiento Creative Commons*, donde hablaremos sobre qué es Creative Commons, y por qué es un magnífico ejemplo de colaboración.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=cbac4c5c-c005-4212-97e7-ad78013d2e6c>

---

## Creación de un equipo DevOps

Algunas organizaciones (como Netflix) no tienen equipos independientes de desarrollo y operaciones, en cambio, tienen un único equipo llamado No-Ops, que comparte ambos roles y responsabilidades. Otras organizaciones han tenido éxito con los equipos DevOps de enlace, que resuelven cualquier conflicto y fomentan la colaboración. Dicho equipo está conformado por representantes de todos los equipos y tienen una participación en la aplicación que está siendo desarrollada. Cualquiera sea la organización de equipos, si decides tener un equipo DevOps, el objetivo más importante es asegurarte de que funciona como un centro de excelencia, que facilita

la colaboración y **no añade un nuevo nivel de burocracia.**

## Procesos DevOps

Anteriormente, hemos hablado sobre el papel que tienen las personas y la cultura en la adopción de DevOps. Los procesos definen lo que esas personas deben hacer y cuándo deben hacerlo. Por tanto, la organización puede tener una gran cultura de colaboración, pero si la gente está realizando las tareas de forma errónea o, por el contrario, están trabajando eficazmente, pero en el camino equivocado, el fracaso es todavía más probable.

Como ya hemos mencionado, DevOps es una capacidad que afecta a toda la empresa, haciendo que el negocio sea más ágil y que mejore la entrega de características a los clientes. También podemos ir más allá y entenderlo como un proceso de negocio: un conjunto de actividades o tareas que produce un resultado específico (producto o servicio) para los clientes.

El proceso de negocio de DevOps consiste en llevar las capacidades fijadas normalmente por los directivos, dueños o inversores del negocio, a través del desarrollo y las pruebas, hasta que llegue a producción. Aunque este proceso de negocio no es lo suficientemente maduro para ser capturado en un flujo de procesos simples, es necesario poder capturar los procesos que la organización utiliza actualmente para producir y entregar las características. Esto servirá como punto de partida para realizar las modificaciones, cambios y vueltas de tuerca necesarias para impulsar el cambio hacia una cultura DevOps.

En la Figura 1 podemos visualizar cómo estos objetivos de negocio se integran con las tareas y capacidades del equipo de tecnología de la información (en adelante, TI). En un ciclo de vida DevOps, personas y procesos deben integrarse bajo el lema de la integración y entrega continua e incorporar las herramientas de software adecuadas para un resultado exitoso.

El resultado de dicha integración permitirá crear una sinergia que beneficiará a los clientes y usuarios finales, pero también a la organización, que ganará agilidad, flexibilidad y eficiencia.

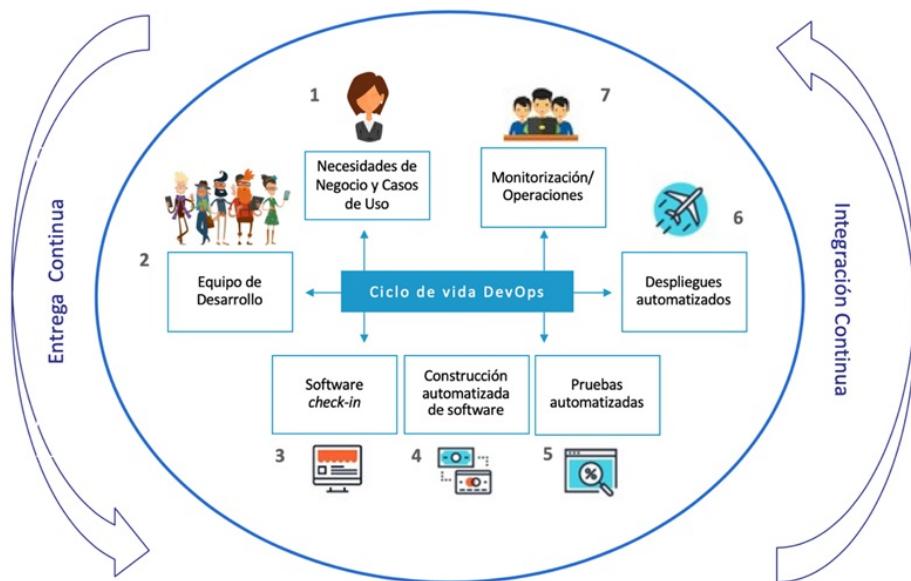


Figura 1. Ciclo de vida DevOps. Fuente: elaboración propia.

## 2.3. Herramientas que favorecen la colaboración

Cuando hablamos de herramientas de colaboración en el mundo de TI o en el ámbito de Internet, la lista podría ser casi infinita. En este caso, vamos a centrarnos en las herramientas que son utilizadas con éxito dentro de una estrategia de DevOps.

A la hora de analizar estas herramientas, podemos establecer una diferencia clara entre estos dos tipos:

- ▶ **Herramientas metodológicas.** Cuando hablamos de herramientas metodológicas, nos referimos a aquellas que favorecen la mejora de las prácticas que hacen posible la colaboración. Un ejemplo de ello podrían ser las metodologías ágiles, que fomentan la creación de un entorno innovador, de cambio y de colaboración. Algunas personas podrían pensar que estas herramientas no son tan importantes, porque pretenden encontrar en DevOps recetas mágicas que se puedan aplicar desde el día cero y buscan obtener beneficios a corto (cortísimo) plazo. Esto es imposible, ya que DevOps requiere de un cambio cultural y, como tal, las buenas prácticas y los cambios a nivel humano son fundamentales y llevan tiempo. De hecho, es perfectamente posible implementar DevOps en una organización sin la necesidad de adoptar numerosas herramientas de software, pero sí es totalmente necesario el abrazar el cambio cultural.
- ▶ **Herramientas software.** Por otro lado, cuando hablamos de herramientas software nos referimos a aplicaciones específicas que pueden habilitar nuevos canales de comunicación y colaboración dentro de la organización. Si bien hay que destacar que las herramientas de software son solo herramientas y no podemos esperar una mejora sustancial en la productividad solo por el mero hecho de usarlas, sí es cierto que son aceleradores o, incluso en algunos casos, catalizadores de la actividad y del cambio. Si bien nos referimos a estas herramientas como herramientas de colaboración, estas se centran en facilitar la comunicación y el intercambio de información.

## Ejemplos de herramientas metodológicas

A continuación, veremos una serie de herramientas que resultan sumamente eficaces por su sencillez y facilidad de uso, y por los resultados que arrojan en contraposición con los métodos o herramientas tradicionales.

### Metodologías ágiles

Se suele llamar metodologías ágiles, en comparación con la ingeniería de software tradicional, al conjunto de técnicas de desarrollo dedicadas principalmente a los sistemas complejos y al desarrollo de productos con características dinámicas (no deterministas y no lineales) donde las estimaciones precisas, los planes estables y las predicciones son, a menudo, difíciles de conseguir en etapas tempranas.

Estas metodologías se basan en la experiencia práctica de expertos de cientos de proyectos, donde las aproximaciones evolutivas han demostrado ser más eficaces que los métodos tradicionales. Muchas veces se hace referencia a los métodos ágiles como métodos de adaptación continua o métodos adaptativos. Estos se centran en aplicar rápidamente los cambios de rumbo en un proyecto, modificando estimaciones, requisitos, alcance, etc., con el objetivo de garantizar un resultado final satisfactorio.

Dentro de este grupo, podemos encontrar las siguientes metodologías:

- ▶ Adaptive Software Development (en siglas, ASD).
- ▶ Crystal Clear.
- ▶ Feature Driven Development (en siglas, FDD).
- ▶ Kanban.
- ▶ Extreme Programming (en siglas, XP).
- ▶ Scrum.

## Ejemplos de herramientas de software

A continuación, vamos a listar un conjunto de aplicaciones que ayudan a los equipos a construir un entorno colaborativo. Aquí es importante entender que, en algunos casos, se trata de aplicaciones equivalentes mientras que, en otros casos, ambas aplicaciones pueden ser perfectamente complementarias. Si hablamos en términos de funcionalidad, tampoco son iguales, ni siquiera entre aplicaciones equivalentes. Cuando se habla de colaboración, hay muchas formas de entenderla y cada organización tiene sus propias herramientas. Por ello, debemos evaluar las carencias que se necesitan cubrir, ya que no podemos hablar de agilidad e intentar aplicar las mismas soluciones de colaboración a todos los casos sin tener en cuenta las circunstancias particulares de cada uno.

### Slack



Figura 2. Logo de Slack. Fuente: [Slack](#).

Slack fue lanzado en agosto de 2013. Esta herramienta se propone reemplazar a todas las herramientas de comunicación interna de una organización, e incluso, en algunos casos, a las de relación con el cliente. Si buscásemos algún referente histórico para Slack, tendríamos que hablar del *Internet Relay Chat* (en siglas, IRC), una de las primeras herramientas de chat en línea que aún perdura.

Algunas de las características que aporta Slack son:

- ▶ Las salas de chat persistentes (llamadas canales) organizadas por tema.
- ▶ Grupos privados.
- ▶ Mensajería directa (basado en el IRC).

En Slack, todo el contenido está indexado y tiene un menú de búsqueda que permite rastrear archivos, conversaciones o personas. Uno de los puntos fuertes de Slack es que se integra con un gran número de servicios de terceros y apoya las integraciones creadas por la propia comunidad. Algunos de los principales ejemplos de integración son: Google Drive, Trello, Dropbox, Box, Heroku, Crashlytics, GitHub, Runscope y Zendesk.

Sin dudas, esta aplicación es una gran aliada en el viaje hacia la adopción DevOps, ya que su foco está puesto en **proveer agilidad a la comunicación entre equipos de trabajo dentro de una organización**. Los equipos desean que las comunidades, grupos o departamentos que se unen a través de una URL, o invitación específica enviada por un administrador o propietario del equipo, colaboren en un espacio de comunicación compartida. Aunque Slack estaba inicialmente destinada a la comunicación organizacional, se ha ido convirtiendo lentamente en una plataforma más amplia, reemplazando muchas veces la función que anteriormente cumplían las redes sociales como Facebook o LinkedIn. Además, Slack pretende (a mediano plazo) reemplazar al correo electrónico y la compartición de ficheros.

## Microsoft Teams



Figura 3. Logo de Microsoft Teams. Fuente: Microsoft Teams.

Microsoft ha ido agregando muchas características y aplicaciones nuevas a Office 365 como Planner, Shifts y Microsoft Teams. Aprovechando el éxito rotundo de Slack, Microsoft ha desarrollado Teams como un espacio de trabajo basado en chat en Office 365, que permite que los equipos trabajen juntos para fomentar la colaboración, interacción y compartición de archivos. Es importante tener en cuenta que Microsoft Teams reemplazará a Skype for Business (que se retirará del mercado el 31 de julio de 2021).

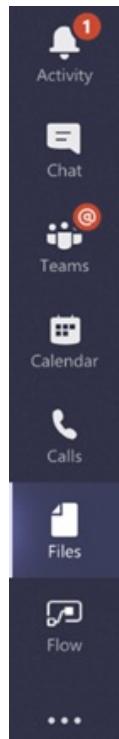


Figura 4. Menú de Microsoft Teams. Fuente: elaboración propia.

El menú de Microsoft Teams da acceso a diferentes áreas como actividad, chats, Teams, calendarios, llamadas, archivos. Veamos un resumen de algunas de estas.

- ▶ Actividad: registra la actividad que ha habido en los canales en los que el usuario está suscrito, es decir, aquellos canales de los que forma parte.
- ▶ Chat: mantiene un historial de todas las conversaciones que han tenido lugar dentro de la organización. Además, da la opción de realizar llamadas, videollamadas y compartir pantalla.
- ▶ Teams: ofrece una lista de todos los equipos a los que pertenece el usuario y permite acceder a los canales dentro de los equipos.

- ▶ Calendario: se sincroniza con el calendario de Outlook y también permite programar reuniones dentro de esta pestaña. Sin embargo, si el usuario desea calendarizar reuniones con personas ajenas a la organización, deberá utilizar Outlook (al fin y al cabo, el objetivo es la colaboración en equipo, no la gestión del calendario).
- ▶ Archivos: dentro de este menú se pueden encontrar y visualizar archivos alojados en OneNote, OneDrive y Teams (almacenados en SharePoint). También está la pestaña «Reciente» que resulta muy útil para acceder a los últimos documentos en los que se estaba trabajando, así como un acceso directo a las descargas.

## Google Docs



Figura 5. Funcionalidades de Google docs. Fuente: [Google docs](#).

Google Docs es un **Software as a Service (en siglas, SaaS)** y podríamos decir que es el equivalente de Google al Office 365 de Microsoft. Permite crear documentos de texto, hojas de cálculo y presentaciones. Podemos utilizarlo, también, como un simple repositorio de documentos y archivos (Google Drive) o incluso como un editor web. Los documentos se guardan automáticamente en la nube de Google y cuenta con una característica muy valiosa: **permite ejercer un estricto control de versiones al dar al usuario la posibilidad de ver todas las modificaciones realizadas sobre el documento desde su existencia hasta la fecha**. Además, se pueden instalar diferentes *plugins* para hacer una edición de documentos en modo *offline* y que los ficheros se descarguen en el ordenador. Los documentos también se pueden exportar en varios formatos estándar como:

- ▶ ODF

- ▶ HTML
- ▶ PDF
- ▶ RTF
- ▶ Texto plano
- ▶ Open Office
- ▶ XML

Por otra parte, los documentos pueden ser enriquecidos mediante etiquetas y metadatos. En lo referente a la compatibilidad, el servicio soporta todas las versiones recientes de Firefox, Internet Explorer, Safari y Chrome que se ejecutan en sistemas operativos Windows, OSX, y Linux.

Como hemos visto, Google Docs sirve como una herramienta de colaboración para la edición de documentos en tiempo real. Los documentos pueden ser compartidos, abiertos y editados por varios usuarios síncronamente y se pueden ver los cambios «en vivo y en directo» (carácter a carácter) mientras los colaboradores sobrescriben y editan el documento.

Asimismo, Google Docs es compatible con los formatos de documentos estándar ISO: XML de OpenDocument y Open Office. También soporta los formatos propietarios como DOC y XLS. Además, posee una herramienta portapapeles web que permite a los usuarios copiar y pegar contenido entre otras plantillas de Google: documentos, hojas de cálculo, presentaciones y dibujos. El portapapeles web también se puede utilizar para copiar y pegar contenido entre diferentes ordenadores. Los elementos copiados se almacenan en los servidores de Google durante un máximo de 30 días. Por último, vale la pena mencionar que Google Docs está integrado con Gmail y Hangouts, permitiendo también el uso del *e-mail*, chat,

calendarios compartidos y videoconferencias.

## Office 365



Figura 6. Funcionalidades de Office 365. Fuente: [Office 365](#).

Office 365 es un Software as a Service (SaaS) que se compone de una serie de productos y servicios. **Todos los componentes de Office 365 se pueden gestionar y configurar a través de un portal, los usuarios pueden añadir contenidos manualmente o importarlos de un archivo CSV, por ejemplo.** Además, puede configurarse un Active Directory local utilizando los servicios de federación de Active Directory para el inicio de sesión. Entre las herramientas que incorpora Office 365 podemos nombrar:

- ▶ Servicio de correo electrónico.
- ▶ Gestor de tareas.
- ▶ Aplicación de calendario.
- ▶ Gestor de contactos.
- ▶ Aplicaciones de ofimática.

Una de las características más interesantes que presenta es su capacidad de funcionar en un entorno de nube donde todos los archivos residen en Internet de forma segura. La lista de productos ofertados ha ido creciendo rápidamente, incluyendo Exchange, Skype para empresas (Skype for Business), SharePoint,

Yammer, Flow OneNote, Dynamics 365 y la suite de Office Web Apps basada en navegador y el servicio OneDrive. Como novedad importante, existe la opción de acceso gratuito a las aplicaciones de Office Mobile para Android y dispositivos iOS (incluyendo tanto los teléfonos inteligentes como las tabletas) para la edición y la creación de documentos. Por supuesto, otra de las grandes ventajas que tiene es que no es necesario preocuparse por las actualizaciones.

## Dropbox



Figura 7. Logo de Dropbox. Fuente: [Dropbox](#).

Dropbox es una herramienta que permite sincronizar archivos a través de un directorio virtual o disco duro virtual en la red. Es decir, **esta aplicación nos permite disponer de un disco duro o carpeta virtual de forma remota y accesible desde cualquier ordenador, estemos donde estemos (en cualquier lugar del planeta)**. Podríamos decir que es como tener un *pen drive* USB, pero alojado en Internet, de tal forma que nos permite almacenar toda la información que deseemos en la red y tener esa información disponible desde cualquier PC.

Cuando instalamos Dropbox, se crea un directorio en nuestro ordenador y, cada vez que guardemos los datos en ese directorio, realmente lo estaremos guardando en nuestro «espacio» en Internet. Eso nos permite tener una copia de nuestros archivos (siempre sincronizada) alojada en Internet de forma que, cada vez que hagamos cambios en nuestros documentos, estos se guardarán sincronizados en la red y, además, se conservará la versión antigua en caso de que la necesitemos en el futuro.

Dropbox es compatible con el control de versiones para varios usuarios, lo que

permite que varios usuarios puedan editar y volver a colocar los archivos sin sobrescribir versiones.

En nuestro contexto, la funcionalidad clave es la de compartir, y DropBox permite a sus usuarios compartir archivos con diferentes niveles de acceso, tanto con usuarios de Dropbox como con usuarios externos, e incluso admite la compartición pública.

DropBox es extremadamente popular y extensible, por lo que los usuarios han creado una serie *mash-ups* que amplían la funcionalidad de Dropbox. Estos incluyen:

- ▶ El envío de archivos a un Dropbox a través de Gmail.
- ▶ La sincronización de los registros de chat de mensajería instantánea.
- ▶ BitTorrent.
- ▶ Gestión de contraseñas.
- ▶ El inicio de aplicaciones en remoto.
- ▶ La monitorización del sistema.
- ▶ Un servicio de alojamiento web gratuito.

## Trello



Figura 8. Logo de Trello. Fuente: [Trello](#).

Trello es una **aplicación de gestión de proyectos basada en web**. Esta aplicación

utiliza el paradigma de Kanban para la gestión de proyectos, popularizado originalmente por Toyota en la década de 1980 para la gestión de la cadena de suministro. Los proyectos están representados por tableros, que contienen listas (listas de tareas) y estas, a su vez, contienen tarjetas (correspondientes a las tareas). Las tarjetas deben moverse de una lista a la siguiente (a través de «arrastrar y soltar»), a fin de reflejar el flujo de una característica desde la idea o concepción hasta la ejecución y finalización. Cada tarjeta puede ser asignada a uno o varios usuarios, a fin de que esa persona o ese equipo se encargue de llevar a cabo esa tarea. Por último, cabe mencionar que es compatible con iPhone, Android y Windows 10.

## BaseCamp



Figura 9. Logo de Basecamp. Fuente: [Basecamp](#).

Basecamp es una herramienta de gestión de proyectos basada en web y desarrollada por la empresa, del mismo nombre, en el año 2004. Se caracteriza, fundamentalmente, por la **simplificación de los métodos de trabajo y por favorecer una circulación fluida de información entre los usuarios**. Algunas de las funcionalidades de Basecamp son:

- ▶ La compartición de archivos entre los miembros de un equipo de trabajo.
- ▶ Asignación de tareas.
- ▶ Planificación de calendarios.
- ▶ Generación de informes.

- ▶ Establecimiento de fechas de entrega.

Todas estas acciones se hacen de manera sencilla e intuitiva, desde cualquier ordenador con acceso a Internet.

## Otras herramientas

Resultaría imposible realizar aquí un análisis completo de todas las herramientas de colaboración que existen en la actualidad, y no solo eso, sino que también se escapa del alcance de la asignatura, ya que constantemente surgen nuevas opciones.

Por eso, hemos realizado un pequeño resumen de aquellas que resultan más relevantes y, como añadido, se incluye debajo una lista de otras herramientas que pueden resultar de útiles para facilitar la implantación de DevOps:

- ▶ [Asana](#).
- ▶ [Onehub](#).
- ▶ [PBWorks](#).
- ▶ [Zoho Projects](#).

## 2.4. Factores que aceleran la adopción de DevOps

Para que la adopción de DevOps sea exitosa, es necesario que la organización tenga el objetivo de innovar. Sin embargo, también es frecuente que la decisión de implantar este enfoque se produzca por la aparición y repetición de problemas recurrentes:



Figura 10. Problemas recurrentes en las organizaciones. Fuente: elaboración propia.

### Procesos pensados para controlar y no para habilitar

Imaginemos el siguiente escenario: un equipo de 100 desarrolladores necesita tener acceso a los recursos de pruebas para realizar sus tareas habituales y cotidianas (desarrollo, demos, etc.) y la empresa desea que estos recursos se consuman en la nube con el objetivo de que sean más ágiles y evitar así una inversión en hardware. En una organización tradicional, podrían surgir muchos inconvenientes si se diese a los desarrolladores acceso a esos recursos (gasto descontrolado, poca visibilidad, imposibilidad de prever, etc.). A continuación, probablemente se definiría un proceso para recountar y contabilizar los recursos que se están consumiendo y, de alguna forma, aprobar este consumo. Por último, es posible que se generase un *workflow* (ciclo) de aprobaciones que, si bien funcionaría, dejaría mucho que desear en términos de eficiencia y agilidad.

**Desde el enfoque DevOps, debemos responder al interrogante de cómo se puede implementar un sistema para que los desarrolladores puedan acceder de forma eficiente y rápida a los recursos, con control y visibilidad.** La respuesta, como es de esperar, será diferente a la de una empresa tradicional. Se definirán las políticas que permitan a los desarrolladores autoabastecerse de los recursos necesarios, mediante unas limitaciones definidas a priori. Un ejemplo real puede ser el de una empresa que cuente con un portal autoservicio (*self-service*) que tenga un presupuesto ya aprobado para permitir al equipo que invierta en herramientas y recursos, de acuerdo con las políticas que ha dictado la empresa. De esta manera, se elimina la burocracia de los ciclos de aprobación. Recordemos que este enfoque favorece la independencia, la responsabilidad, la eficiencia y, por supuesto, la colaboración.

## Falta de visibilidad

El hecho de establecer múltiples puntos de control es una forma de asegurar que las tareas, las personas, los recursos y demás elementos que componen la organización, sean gestionados de forma correcta. Sin embargo, también es sabido que todos estos puntos de control y procesos burocráticos pueden generar cuellos de botella o incluso generar una pérdida de tiempo y dinero por la monitorización de aspectos que no generan ningún beneficio a la compañía.

En línea con lo que hemos comentado en el apartado anterior, el hecho de tener una buena visibilidad reporta muchas más ventajas que ejercer un control excesivo. Por decirlo de otra forma, **una buena visibilidad en conjunto, con altos niveles de responsabilidad y compromiso, permitirá al equipo reaccionar ante cualquier situación y cambiar de rumbo, mientras que un control férreo ralentizará a todo el equipo y no garantiza un resultado exitoso.**

Otro caso de falta de visibilidad, lo podemos encontrar en las organizaciones que utilizan cuentas Amazon Web Services (en siglas, AWS) no asociadas, lo que eleva

el gasto a nivel corporativo porque, al no haber una asociación de cuentas, no se pueden aprovechar de los descuentos corporativos por volumen que ofrece el proveedor de nube.

## Poca capacidad de innovación

La innovación en sí misma implica agilidad. Resulta imposible realizar cambios realmente significativos e innovadores en una organización encorsetada por una maraña de reglas. Debe haber un cambio de filosofía en las empresas que aporte más flexibilidad a sus procesos.

Algunas veces es fácil identificar los puntos de mejora, sin embargo, resulta difícil encontrar la mejor solución posible a un determinado *issue*, así que aquí es donde la innovación entra en escena. Hablando desde un punto de vista meramente técnico, además de un buen análisis de la situación y de las posibles soluciones, **es importante que los equipos tengan la libertad para probar, cometer errores y aprender de ellos.** En esto se basa, principalmente, la innovación, así que trataremos a este proceso como un proceso de definición/prueba/error/exito/aprendizaje, un ciclo con tantas iteraciones como sean necesarias.

## Dificultades técnicas evitables

DevOps aboga por la eliminación de la complejidad, ya que esta va en detrimento de la productividad y la eficiencia. Seguramente conozcas aquel dicho que reza: «Lo bueno, si simple, dos veces bueno». Pues bien, en muchos entornos las dificultades técnicas que se perciben como «técnicas» son en realidad dificultades humanas y, a menudo, están causadas porque se trabaja con procesos extremadamente complejos, que no hacen más que incrementar los puntos de fallo.

En las organizaciones tradicionales, existen ciclos de aprobación de cambios de código, comités de aprobación de pase a producción y, por supuesto, esto no hace más que añadir complicaciones a la hora de revertir el pase y corregir el código. **En**

**una organización DevOps, la gobernanza está en manos del equipo DevOps,** y ellos pueden acceder al código y subsanar cualquier problema que surja en un breve plazo de tiempo, sin la necesidad de que un comité o departamento les autorice (esta es la verdadera agilidad a la que hacemos referencia cuando hablamos de DevOps).

## Miedo y resistencia al cambio

El miedo es endémico a la vida y al ser humano, de la misma forma que es el peor enemigo de la innovación y la colaboración. Nos encontraremos con el simple miedo al cambio, que incluso es un motivador positivo en las proporciones adecuadas, o el miedo a la pérdida de control, que en el caso de DevOps debería entenderse como un paso hacia la independencia y la visibilidad. Pero el peor, sin dudas, es el miedo a la pérdida del *status quo* por parte de aquellas personas que quieren ser imprescindibles.

En un equipo DevOps, las personas pueden jugar diferentes roles en el transcurso de un proyecto, pero también comparten información sobre los avances, estados y alcance, entre otras cosas. Aquellos miembros que se guardan información, conocimiento y herramientas, creyendo que así serán imprescindibles, tienen cada vez menos cabida en las organizaciones que desean crecer.

Como indica Cariñena (2016), **las personas imprescindibles no atesoran los conocimientos como propios, sino que los comparten, están siempre predispuestas a ayudar a los demás y a emprender nuevos retos.** Estas personas son imprescindibles porque sin ellas las cosas se podrían hacer (no vamos a decir que no), pero con más esfuerzo o incluso con menos calidad. Pero lo más importante de todo, es que el mundo gira cuando no están, porque consiguen «enseñar a pescar» y no solo dan pescado cuando se necesita.

**Ese es la fórmula que describe el espíritu de un DevOps: colaborar + compartir + ayudar + producir.**

## Shadow IT

El *shadow IT* (o TI oculta), engloba a todas las tecnologías y elementos dentro de TI que están fuera de control o simplemente son desconocidas. Siguiendo nuestro enfoque de colaboración, innovación y agilidad, la falta de control no es el problema, sino que lo es **la no estandarización y la falta de visibilidad derivada de ello**. Es importante comprender que, en muchos casos, la necesidad imperiosa de ejercer control es lo que ha causado el *shadow IT*, es decir, la imposibilidad de acceder a los recursos en tiempo y forma. Numerosas organizaciones, grandes o medianas, se han visto en la tesitura de estar usando recursos en la nube antes de poder siquiera diseñar una estrategia de nube.

Según Génesis (2018), entre los ejemplos más frecuentes de *shadow IT* podemos destacar el uso de:

- ▶ Aplicaciones de almacenamiento en la nube, como Google Docs o Dropbox.
- ▶ Redes no permitidas (como la red pública de Internet).
- ▶ *Bring your own device* (en siglas, BYOD) no autorizado.
- ▶ Aplicaciones de terceros como las de sistema en la nube tipo *Software as a Service* (SaaS).

Una de las principales razones que motivan a los empleados a usar aplicaciones no autorizadas, además de su bajo coste, es la facilidad que estas ofrecen en cuanto a acceso, mantenimiento y despliegue.

Entre las principales consecuencias que acarrea el *shadow IT* podemos encontrar (Rivas, 2018):

- ▶ **Fuga de datos sensibles:** a través de aplicaciones de terceros que no estén debidamente supervisadas por el equipo de seguridad de TI.

- ▶ **Procesos lentos:** el despliegue de ciertas aplicaciones puede ralentizar u obstaculizar el desempeño de los procesos de la organización.
- ▶ **Ineficiencia en la resolución de problemas:** en caso de que algún dispositivo o aplicación de *shadow IT* genere un incidente, puede resultar complicado para los profesionales de TI encontrar la fuente del problema.

## 2.5. Referencias bibliográficas

Cariñena, P. (2016, enero 28). *Personas imprescindibles... Las que se lo creen y las que de verdad lo son.* <https://www.linkedin.com/pulse/personas-imprescindibles-las-que-se-lo-creen-y-de-son-paz/>

Rivas, G. (2018, agosto 17). *Shadow IT: ¿Cómo hacer de la TI invisible una oportunidad para tu organización?* <https://www.gb-advisors.com/es/shadow-it/>

Microsoft. (s.f.). *Videoconferencia, reuniones, llamadas / Microsoft Teams.* <https://www.microsoft.com/es-es/microsoft-365/microsoft-teams/group-chat-software>

Dropbox. (s. f.). *¿Qué es?: Resumen de funciones .* [https://www.dropbox.com/es\\_ES/features](https://www.dropbox.com/es_ES/features)

## La cultura DevOps

Exceltic. (2018, octubre 26). *La cultura DevOps* [Vídeo]. YouTube.

<https://www.youtube.com/watch?v=Te47jTCDfVM>

Este recurso te ayudará a comprender un poco más sobre la cultura DevOps dentro de las empresas y sus beneficios.

## Google Cloud DevOps: Speed With Reliability and Security

Google Cloud Platform. (2019, abril 11). *Google Cloud DevOps: Speed With Reliability and Security (Cloud Next'19)* [Vídeo]. YouTube. <https://www.youtube.com/watch?v=cXXZ-AOCALU>

En este vídeo aprenderás más acerca de cómo funciona la implementación de DevOps dentro de Google.

1. Indica cuáles de estas afirmaciones son correctas acerca de la colaboración en DevOps:

  - A. Los resultados de la transformación de la cultura organizacional son inmediatos.
  - B. La cultura de la colaboración involucra únicamente a las personas.
  - C. La colaboración solo puede ser abordada de la mano de la tecnología, por lo tanto, sin herramientas tecnológicas, no habrá un entorno de trabajo colaborativo.
  - D. Ninguna de las anteriores.
2. Indica cuáles de estas afirmaciones son correctas acerca de las herramientas metodológicas:

  - A. Estas herramientas fomentan la creación de un entorno innovador, de cambio y de colaboración.
  - B. Se ha demostrado que estas herramientas son menos eficaces que los métodos tradicionales.
  - C. Estas herramientas incluyen a las metodologías ágiles.
  - D. A y C son correctas.
3. El *shadow IT* acarrea importantes consecuencias. Indica cuál de las siguientes opciones es la más adecuada para describir algunas de ellas:

  - A. Fuga de información y datos sensibles, e ineficiencia en la resolución de problemas.
  - B. Ralentización de procesos, y rotación del personal.
  - C. Fuga de talentos, información y datos sensibles.
  - D. Fuga de información y datos sensibles, ralentización de procesos e ineficiencia en la resolución de problemas.

4. ¿Cuáles suelen ser los problemas que actúan como potenciadores de la adopción de DevOps? Elige la opción más acertada:
- A. Falta de visibilidad, miedo y procesos burocráticos que van en detrimento de la agilidad.
  - B. Pérdida de la capacidad de innovación, dificultades técnicas y *shadow IT*.
  - C. Personal poco cualificado.
  - D. A y B son correctas.
5. DevOps puede ser entendido como una capacidad de negocio porque:
- A. Es un conjunto de actividades o tareas que produce un resultado específico (producto o servicio) para los clientes.
  - B. Consiste en llevar las capacidades fijadas por las áreas de negocio, a través de todo el ciclo de desarrollo, desde su inicio y hasta su fin (*end-to-end*).
  - C. Toda organización debe adoptar DevOps si desea competir en el mercado de forma exitosa.
  - D. A y B son correctas.

6. Termina la frase: «Para la conformación de un equipo DevOps debemos procurar...»:

- A. Que funcione como una perfecta maquinaria con diversos engranajes; que prime la colaboración como valor fundamental; y que no se añadan niveles de burocracia en los procesos internos.
- B. Que cada individuo vele por sus propios objetivos de eficiencia y productividad, a fin de que su evaluación de desempeño tenga un excelente resultado.
- C. Que los desarrolladores puedan trabajar en parejas, bajo la práctica del *pair programming*.
- D. Ninguna de las anteriores.

7. Relaciona los elementos interviniéntes en la cultura de la colaboración DevOps en contraposición con aquellos de las metodologías tradicionales (en cascada):

Visibilidad	1	A	Trabajo manual
Automatización	2	B	Burocracia
Agilidad	3	C	Control
Ciclos iterativos	4	D	Modelo secuencial

- 8.** Termina la frase: «Para garantizar el éxito en la adopción de DevOps, los líderes deben...»:
- A. Establecer un conjunto común de herramientas a toda la organización para que los trabajadores conozcan los diferentes entornos y trabajen de la forma más ágil que les sea posible.
  - B. Lograr que todos los trabajadores estén alineados con los objetivos de negocio.
  - C. Crear procesos que eliminen la burocracia y proporcionen flexibilidad, a fin de que los equipos puedan solventar los fallos cuando sea necesario.
  - D. Todas las anteriores.
- 9.** Elige la opción correcta para la siguiente afirmación: «En un ciclo de vida DevOps, intervienen los siguientes procesos»:
- A. El establecimiento de los objetivos de negocio y casos de uso, la comunicación de los objetivos y requisitos al equipo de desarrollo, el *check-in* de software y el desarrollo automatizado.
  - B. La integración y entrega continuas.
  - C. Las pruebas y despliegues automatizados, y la monitorización continua.
  - D. Todas las anteriores.
- 10.** ¿Cuál de las siguientes afirmaciones es correcta acerca de DevOps?:
- A. Este enfoque resulta apropiado para las organizaciones que pueden invertir grandes cantidades de dinero en software y hardware.
  - B. DevOps no es el objetivo final de una organización, sino que es una herramienta que permite alcanzar los objetivos propuestos.
  - C. DevOps no es compatible con la virtualización.
  - D. Ninguna de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 3. DevOps Dojos

# Índice

[Esquema](#)

[Ideas clave](#)

[3.1. Introducción y objetivos](#)

[3.2. Dojos tecnológicos](#)

[3.3. Cómo crear un Dojo](#)

[3.4. Selección e ingesta de nuevos equipos](#)

[3.5. Medición del éxito del Dojo](#)

[A fondo](#)

[DevOps Dojo: A Massive Internal Coaching Program – DXC Technology](#)

[Test](#)



## 3.1. Introducción y objetivos

Estamos en un momento de transformación a nivel humano, organizacional y tecnológico. Estas transformaciones giran en torno a cuatro resultados clave:

- ▶ Transformación del modelo operativo: cambio de paradigma en el pensamiento y desarrollo, desde proyectos a productos.
- ▶ Transformación cultural: habilitar a trabajadores altamente capacitados, comprometidos y eficientes.
- ▶ Transformación hacia la tecnología: aplicación intensiva de Agile y DevOps.
- ▶ Transformación de la arquitectura tecnológica: abrazar el uso de las plataformas y tecnologías modernas.

Los objetivos que se pretenden conseguir en este tema son:

- ▶ Conocer el concepto de Dojo.
- ▶ Aprender cómo iniciar la revolución cultural de DevOps gracias a los Dojos.
- ▶ Aprender a diseñar y mantener un Dojo exitoso.

## 3.2. Dojos tecnológicos

Los Dojos pueden ser un vehículo singularmente poderoso para acelerar la transformación a través de estos cuatro pilares de cambio que ya mencionamos. «Dojo» significa ‘lugar del camino’ en japonés. El Dojo, en nuestro contexto, es un **espacio diseñado para albergar una experiencia de aprendizaje inmersiva donde equipos completos vienen a aprender ingeniería moderna, productos y prácticas ágiles.**

Los equipos participan en experiencias de Dojo, a las que típicamente se hace referencia como un «desafío» o «reto». En estos desafíos, el equipo construye productos reales y trabaja en problemas reales, utilizando herramientas populares y conocidas en el mercado, mientras es dirigido por un entrenador del Dojo. El proceso de trabajo dentro del Dojo está típicamente compuesto por: *hiper-sprints*, un patrón común de *sprints* de dos días y medio durante doce ciclos de *sprints* (es decir, seis semanas). Durante el tiempo en que un equipo está en el Dojo, completa varios *sprints*, siguiendo este patrón de trabajo que representa una estructura cíclica.

En el transcurso del Dojo, el equipo construye una nueva experiencia y desarrolla nuevas habilidades. Uno de los elementos clave es el hecho de que el equipo interdisciplinar trabaje en un problema real y haga uso intensivo de las nuevas tecnologías, arquitecturas y enfoque de desarrollo de productos. Todo esto permite al equipo dominar rápidamente las habilidades y las prácticas de trabajo que son deseables para un equipo DevOps moderno.

## ¿Qué no es un Dojo?

- ▶ **Un Dojo no es una forma de acelerar la entrega de los proyectos existentes.** Cuando un equipo entra en el Dojo, su prioridad es aprender juntos y aprovechar las nuevas tecnologías y formas de trabajo. Desconfía de los líderes que quieren poner a sus equipos en un Dojo simplemente porque tienen fechas de entrega agresivas y quieren «acelerar sus equipos». La verdad es que hay que estar dispuesto a reducir la velocidad a corto plazo para acelerarla a largo plazo. Además, los equipos deben tener el espacio físico y mental adecuado que propicie una experiencia de aprendizaje. Las organizaciones se beneficiarán a largo plazo a medida que los equipos se vuelvan más productivos y puedan cumplir con tareas cada vez más complejas.
- ▶ **Un Dojo no es una práctica de entrenamiento puramente Ágil ni Lean.** El enfoque debe ser la creación de equipos más efectivos, y eso requiere reunir múltiples áreas de conocimiento. Un Dojo funciona mejor cuando los equipos tienen una responsabilidad *end to end* (de principio a fin) y están enfocados en el desarrollo de productos.

## Resultados esperables de un Dojo

Para poder competir en «la era del software», las empresas deben contar con un entorno de aprendizaje continuo, que incluya no solo el aprendizaje de la tecnología sino, lo que es quizás más importante, mejores formas de trabajo, patrones y prácticas. También, deben ser eficaces en la colaboración entre equipos y áreas de conocimiento. Esta colaboración es vital, ya que los equipos necesitan dominar las nuevas tecnologías, integrarlas y aportar valor al negocio a un ritmo cada vez más acelerado.

Como ejemplo, considera la posibilidad de desarrollar una nueva aplicación móvil en la que los servicios en línea existentes tienen que ampliarse a un número más grande de clientes sin que ello suponga un tiempo de inactividad y, además, añadiremos que también necesitan ser más rápidos y eficientes. El hecho de que los

equipos trabajen juntos en un espacio de inmersión, diseñado específicamente para el aprendizaje y la mejora, crea un entorno seguro y estructurado. Este **entorno seguro y estructurado permite a los equipos superar la resistencia cultural al cambio.**

Los graduados del Dojo se convierten en evangelistas de las buenas prácticas y los conceptos del Dojo, de tal forma que pueden ayudar a mejorar a otros equipos con los que colaboran. A medida que otros equipos ven los buenos ejemplos y formas de trabajar, comienzan a cambiar también, acelerando la transformación en toda la empresa. **¡Los casos de éxito son poderosos agentes de cambio!**

Con el tiempo, la mejora de las capacidades de estos equipos dispara la aceleración de la creación de valor, lo que también mejora la moral del equipo. De hecho, según numerosos estudios científicos, la felicidad y la **moral de un equipo es un indicador clave en la productividad.** En última instancia, los Dojos se instituyen con la expectativa de que proporcionen un retorno de la inversión (en siglas, ROI) suficiente: acelerando o acortando el tiempo de transformación, incluida la ampliación de la transformación a toda la organización, y mejorando el aprendizaje y el cambio organizativo, que es un determinante clave a los esfuerzos de transformación que no cumplen las expectativas comerciales.

**Los Dojos optimizan la velocidad y la eficiencia en lugar de optimizar los costes. De ello se deduce que la producción rápida de valor para el negocio es lo que hace que compense la inversión.**

A través de los Dojos, los equipos aprenden no solo a centrarse en construir software de manera correcta, a través de las prácticas ágiles, sino que también aprenden a construir software de forma correcta, a través de la orientación a crear valor. En este escenario, la velocidad proviene del hecho de que se reducen significativamente los elementos superficiales de los procesos. **La mayor parte del esfuerzo desperdiciado en la entrega de TI proviene del hecho de que los equipos pasan tiempo construyendo cosas que no añaden valor.**

## Maximización del poder de un Dojo

El máximo poder del Dojo se produce cuando se convierte en un nexo de aprendizaje e inmersión de los principios ágiles, el desarrollo de productos y la adopción de nuevas tecnologías. Construir software de forma correcta es lo que consigue crear mayor valor para una organización. Si el Dojo puede reunir estos tres elementos, los equipos maximizarán su potencial para cambiar y transformarse de verdad.

## Ejecución Ágil

Ágil es el proceso, el método y el conjunto de principios que forman la base del pensamiento y el enfoque de la entrega. Los principios clave de Agile son:

- ▶ Entrega incremental.
- ▶ Retroalimentación y mejora continua.
- ▶ El trabajo en elementos de alto valor.

Estos principios crean un enfoque de la entrega de software que maximiza el tiempo de obtención de valor. Muchos equipos no se dan cuenta del impacto total de una transformación de DevOps porque han implementado las prácticas ágiles de forma deficiente desde el principio. En realidad, en muchos casos se trata de un proceso en cascada envuelto en terminología ágil. Frases como «ScrumBut», «Water-Scrum» o «Scrum-Fall» son reflejo de estas implementaciones ágiles deficientes. El Dojo

debería ser un modelo y una implementación sobresaliente de Agile, debería ayudar a los equipos a hacer los ajustes necesarios para su propia implementación de agile y así maximizar la entrega de valor.

## Orientación al producto

Hay dos componentes que deben abordarse en el Dojo en relación con el término «producto»: **orientación al producto y gestión del producto**.

La **orientación al producto** se refiere a la necesidad de que un equipo que acude al Dojo adopte, si no lo ha hecho ya, una orientación al producto en lugar de una orientación al proyecto. Esta orientación implica que el equipo esté preparado para resolver, crear y construir soluciones que puedan satisfacer una necesidad de negocio.

La **gestión del producto** es la práctica y disciplina de gestionar y traducir las necesidades de la empresa en aquello que puede ser desarrollado y entregado por el equipo de producto. Dentro del Dojo, ambos aspectos deben reflejarse en la experiencia de inmersión. Los equipos deben experimentar el Dojo de tal manera que emergan con:

- ▶ Una perspectiva de enfoque de trabajo a largo plazo.
- ▶ Una fuerte identificación con sus objetivos.
- ▶ Sentido de la responsabilidad sobre aquello que entregan.

## Excelencia tecnológica

Las transformaciones de DevOps, generalmente, ocurren al mismo tiempo que la adopción de nuevas tecnologías. Por ejemplo, muchas compañías emparejarán la adopción de DevOps con sus esfuerzos de adopción de nube. La agilidad de la infraestructura de la nube probablemente reduzca los obstáculos tecnológicos en el sistema y las herramientas de DevOps. La gestión automatizada de las entregas, la

gestión del ciclo de vida de las aplicaciones y la monitorización son ejemplos de nuevas herramientas o tecnologías que los equipos deben aprender a utilizar.

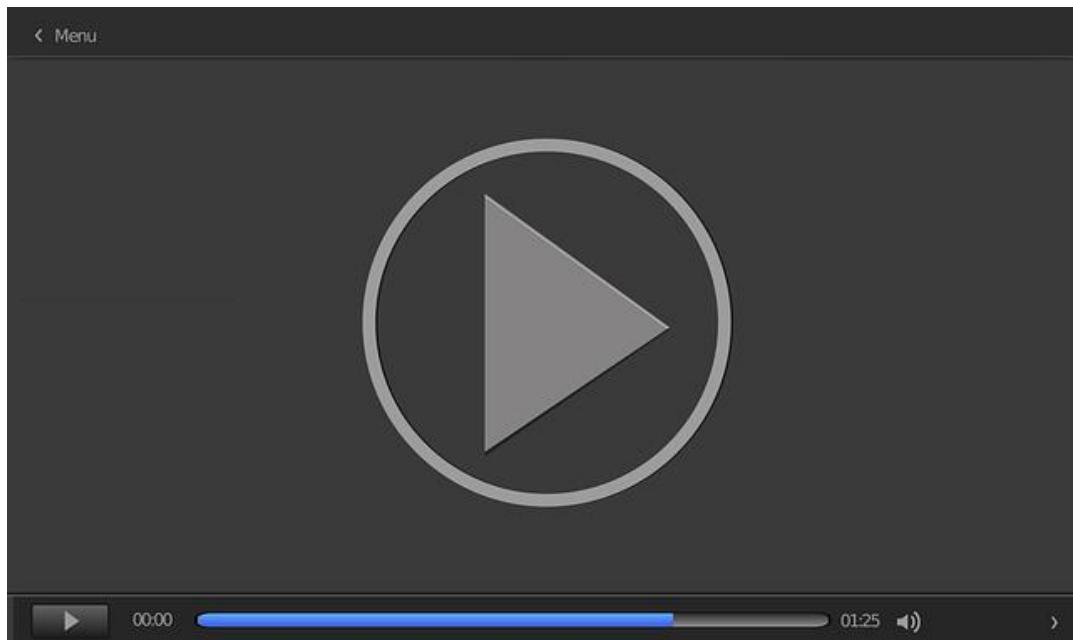
El Dojo debe proporcionar apoyo y propiciar el aprendizaje de cualquier tecnología clave en la adopción de DevOps.

## 3.3. Cómo crear un Dojo

Piensa en los Dojos como tu centro de inmersión de transformación, como si fuese un gimnasio donde entrenar los músculos de tecnología de información (en siglas, TI). Para construir un caso de uso e invertir en un Dojo, enfócate en cómo lo usarías para acelerar tu transformación tecnológica en la organización. El Dojo está pensado como un recurso compartido y de aprendizaje, y es por ello por lo que, en la práctica, la financiación suele provenir del departamento de desarrollo estratégico o innovación de las empresas, ya que el impacto positivo que tendrá abarcará a todas las áreas.

Los creadores del Dojo coinciden en la idea de que se debe invertir en el desarrollo del personal y el espacio de trabajo, a fin de **fomentar el entusiasmo en el aprendizaje de técnicas nuevas**. Asimismo, es importante destacar que el éxito del Dojo se hace más evidente cuando participan ingenieros populares y reconocidos por los miembros del equipo, es decir, líderes naturales o instituidos. Esto ayudará a que el Dojo tenga la dirección estratégica inicial adecuada, y la credibilidad necesaria para construir un plan de adopción a largo plazo. Otra opción para la creación de Dojos, especialmente si la organización no tiene los recursos necesarios en cuanto a espacio, personas o conocimientos técnicos, es colaborar con un servicio externo o una empresa consultora que tenga experiencia en la creación y gestión de Dojos.

Los equipos interdisciplinares han demostrado su alto valor en equipos ágiles y en, prácticamente, cualquier ámbito de la empresa moderna. Te invito a que veas el vídeo *Grupos interdisciplinares y autogestionados* para que descubras donde radica su fortaleza e importancia.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=d24b9864-1abc-4277-b9a5-ad7900c7a8e0>

---

## Creación de un espacio para el aprendizaje y la diversión

Un espacio adecuado es un poderoso facilitador para el cambio de cultura y el aprendizaje. Cuando des las primeras pinceladas en la organización del Dojo, necesitarás identificar cuál será el espacio para albergarlo. Si el Dojo se desarrollará en una sala o espacio dentro de la empresa, procura que sea una zona de alto tránsito, a fin de dar una mayor visibilidad a la experiencia que están viviendo los equipos. Como puedes imaginar, se disparará una gran curiosidad dentro de la organización y es probable que otros equipos deseen participar de un Dojo en el

futuro. Veamos algunos **componentes clave para la configuración del espacio trabajo:**

## **Colocación de mesas de trabajo que optimizan la colaboración**

Hay que tomar una decisión sobre la forma de colocar las mesas de trabajo. Una opción puede ser la de instalar mesas individuales agrupadas que incluyan monitores dobles en cada estación de trabajo. Sin embargo, también es una opción válida colocar una mesa «familiar» sin monitores, para facilitar la comunicación.

Sea cual sea la disposición que se decida, el factor más importante a considerar es que todos los participantes se sienten cerca unos de otros, con la esperanza de que la colaboración en persona sea inevitable. Una práctica común es tener una estación de tecnología, o monitor, en un extremo de cada mesa de equipo, para compartir más rápidamente las pantallas y la posible conexión de vídeo con los miembros del equipo que estén a distancia (colocalizados).

## **Pizarras para fomentar la colaboración y la visualización de ideas**

Las pizarras móviles se pueden utilizar como separadores del espacio para **delimitar espacios mientras los equipos trabajan**. Por supuesto, esta no es la función principal de una pizarra, estas se utilizan para anotar información esencial sobre el equipo: los nombres de los participantes, los objetivos de aprendizaje, los miedos que deben ser vencidos o resueltos, etc. Como puedes ver, estas anotaciones sirven como un recordatorio para el equipo de por qué están allí y qué pretenden conseguir en el Dojo. También ayuda a orientar a otros equipos y a los visitantes ocasionales para que comprendan en qué esfuerzo está centrado el equipo.

## **Espacio abierto para permitir la colaboración entre equipos**

Los equipos que aprenden de los demás en el Dojo son el ejemplo de la mejor forma de eliminar los silos. Parte de este aprendizaje ocurrirá de forma natural cuando los equipos escuchen otras conversaciones, experiencias y casos de uso que puedan

ser relevantes para su propio progreso. Además, parte del aprendizaje ocurrirá a medida que las personas empiecen a relacionarse, apoyarse y ayudarse mutuamente a desbloquear problemas.

## Centro de comunicación y colaboración

Estos espacios son típicamente referidos como salas de demos y son el punto focal de las actividades en el Dojo, a fin de que **todos los equipos del Dojo se reúnan y compartan su trabajo**. Esta es buena forma de conseguir que los interesados que no están en el Dojo, como la dirección y los socios comerciales, entren y participen en el aprendizaje y el progreso del equipo. Para montar una sala de demos, normalmente se necesita un proyector o una pantalla grande, asientos cómodos e informales, buena calidad de sonido y algún dispositivo que permita realizar videoconferencias con otros Dojos u organizaciones. Las demos que se llevan a cabo en la sala son una forma fantástica de reunir a diferentes equipos para compartir información y obtener retroalimentación, lo que ayudará a construir mejores productos. Estas salas también se pueden aprovechar para impulsar eventos más amplios, organizar reuniones externas y fomentar una comunidad interna mediante actividades divertidas, como días de cine, días de cómic o cualquier otra actividad que sea de interés.

## Diseminadores de información

La transparencia del Dojo es fundamental y se considera primordial para fomentar una experiencia cultural a nivel de la organización. El uso de medios para diseminar la información (tales como *dashboards*) que reflejen los principios del Dojo son de gran ayuda para diseminar la cultura. Aunque parezca que carece de importancia, **es imprescindible que los principios del Dojo se hagan visibles en el espacio**. El recordatorio constante de cómo debemos trabajar es importante, ya que el equipo lleva mucho tiempo trabajando de otra forma y nunca está de más reforzar la información que les ayude a enderezar el camino. Es hora de adoptar nuevos principios, experimentar, fallar de forma temprana, disminuir la velocidad para acelerar, etc.

## Espacio que fomente la diversión y la creatividad

El hecho de contar con un espacio que tenga juegos recreativos ayuda a los equipos a **desconectar, refrescar la mente y pensar con más claridad**. Será útil tener estos elementos a disposición del equipo durante el Dojo para cuando los participantes necesiten recargar energías o tomarse un descanso para reforzar la productividad. Como añadido, participar de actividades lúdicas y recreativas fomentará la socialización y las relaciones interpersonales, que resultan fundamentales para el funcionamiento del equipo.

## Personal facilitador del Dojo

### Entrenadores

Similar a un Dojo de artes marciales que es supervisado por un maestro, un Dojo de transformación es dirigido por entrenadores (*coaches*) que conocen cuál es el punto de partida y ayudan a los equipos a visualizar el camino. Estos entrenadores del Dojo pueden contar con la ayuda de otros expertos en la materia, dependiendo de las necesidades del equipo y del objetivo de aprendizaje propuesto. Por ejemplo,

diseñadores y especialistas en experiencia de usuario (en siglas, UX) son un buen ejemplo de expertos externos que podríamos invitar al Dojo si el equipo está diseñando un nuevo portal orientado al cliente.

**El número de coaches variará de acuerdo con la cantidad de participantes, aunque el factor determinante será la diversidad o la práctica en la que se esté trabajando.** Por lo general, suele haber un entrenador que está más enfocado en las prácticas de producto y agilidad y otro que se centra en las prácticas de tecnología, como las prácticas combinadas de integración y entrega continuas (en siglas, CI/CD) y, también, suele haber otro coach especializado en DevOps y desarrollo nativo de la nube.

Los entrenadores pueden ser asignados a múltiples equipos, especialmente cuando estos se vuelven más autosuficientes con el tiempo. Una práctica seguida por algunas empresas que dirigen Dojos es tener a un entrenador asignado a dos equipos a la vez: un equipo que esté en las tres primeras semanas de su desafío y otro que esté en las tres semanas finales. Esto funciona bien, ya que la demanda de tiempo de un entrenador es mucho mayor al principio, cuando los equipos están aprendiendo las nuevas dinámicas de trabajo.

Para la creación de un Dojo, es recomendable **seleccionar entrenadores que tengan una sólida base de conocimiento en ingeniería del software**. El motivo radica en que los equipos trabajarán con problemas específicos y necesitan un entrenador que pueda aportar ideas prácticas, que sea capaz de llevarlos en la dirección técnica correcta para desbloquearlos. Además, es común que los equipos aprovechen su experiencia en el Dojo para avanzar hacia nuevas plataformas, herramientas y pilas de tecnología y, en este escenario, necesitan un entrenador que pueda explicarles cómo hacer la transición desde la tecnología existente hacia las nuevas arquitecturas.

Veamos, a continuación, algunos de los requisitos en los perfiles más comunes de los entrenadores de Dojos:

### **Entrenador ágil (*Agile Coach*)**

Un entrenador ágil debe ser hábil en múltiples prácticas: Ágiles, Scrum y Kanban. Estos entrenadores no deben ser dogmáticos acerca de un marco Ágil, sino que deben aprovechar las metodologías y prácticas para ayudar a los equipos a definir el proceso que mejor se adapte a sus necesidades.

### **Entrenador de producto (*Product Coach*)**

Un entrenador de producto debe ser experto en **diversas prácticas y técnicas de descubrimiento de productos**, así como en comprender cómo validar las ideas con los clientes para determinar el valor de estas. Las aptitudes de gestión de productos también son fundamentales para los instructores, ya que orientarán a esos equipos sobre cómo negociar, establecer prioridades e influir en la dirección de los productos que se construyen. Esto incluye comprender y valorar los diferentes tipos de trabajo que se realizan en la construcción y el funcionamiento de un producto, como las características, los defectos y la deuda técnica.

### **Entrenador de tecnología (*Technology Coach*)**

En todo equipo de Dojo deberá haber un entrenador que posea un amplio conjunto de habilidades tecnológicas que incluyan: el desarrollo *front-end*, el desarrollo *back-end*, la integración, la infraestructura y las plataformas. El entrenador de tecnología debe tener conocimiento de **un amplio conjunto de lenguajes de desarrollo**, especialmente los que se utilizan comúnmente en la empresa en la que se lleve a cabo dicho evento. Existen, también, prácticas tecnológicas comunes que querrás tener en cuenta para enseñar en el Dojo que lleves a cabo en un futuro, como, por ejemplo:

- ▶ **DevOps:** CI/CD, configuración como código, automatización de pruebas, métodos de telemetría, etc.
- ▶ **Seguridad de aplicaciones (SecDevOps):** práctica que consiste en construir código seguro, automatizar la validación de este código e incorporar la responsabilidad sobre la seguridad dentro de los equipos.
- ▶ **Desarrollo nativo de la nube:** práctica de construir aplicaciones independientes, dinámicas y robustas, aprovechando y poniendo en práctica microservicios, contenedores, etc.
- ▶ **Nube pública:** aprovechamiento de los servicios de nube pública para configurar, construir y ejecutar entornos de aplicaciones.
- ▶ **Integración:** creación de *Application Programming Interface* (en siglas, API), servicios empresariales e integraciones, así como la definición de patrones, técnicas y tecnologías de integración.

## Entrenador experto (opcional)

Pueden ser expertos en un conjunto de herramientas o plataformas específicas. Un buen ejemplo son los arquitectos, ingenieros de plataformas de CI/CD concretas, expertos en Kubernetes, ingenieros de fiabilidad de sitios o similares, etc. Si bien los expertos no tienen por qué estar presentes constantemente en el Dojo, es importante que sean accesibles, ya sea por su proximidad física al Dojo o por los canales de comunicación apropiados. Por lo general, los entrenadores solo integran a expertos cuando hay cuestiones profundas con las que los equipos tienen que lidiar y que están más allá de los conocimientos del equipo.

## Gerente o director de operaciones del Dojo (opcional)

Es importante tener a alguien con una amplia visibilidad de todas las necesidades operacionales. Algunas de las responsabilidades habituales de un director de operaciones de un Dojo son:

- ▶ Planificación y mantenimiento del espacio, así como la organización de cualquier arreglo o reconfiguración que pueda ser necesaria.
- ▶ Planificación de las sesiones de entrenamiento y comprobación de que el personal adecuado está disponible para hacer frente a la demanda del Dojo.
- ▶ Organización de entrevistas con los equipos interesados en participar en el Dojo.
- ▶ Gestión de la logística de los equipos que participarán en el Dojo, incluyendo su incorporación, su desarrollo dentro del Dojo y su salida.
- ▶ Gestión de la logística de otros interesados que visiten el Dojo, así como también visitas a clientes potenciales o a empresas externas interesadas en conocer el Dojo.
- ▶ Gestión de la comunicación alrededor del Dojo, incluyendo la señalización del Dojo y, potencialmente, la comercialización.

## Entrenador principal

La principal responsabilidad de este puesto radica en asegurar que el personal **que lleva a cabo el entrenamiento esté enfocado a la enseñanza que permite satisfacer las demandas de habilidades de los equipos de ingeniería**. Estos entrenadores son polifacéticos en términos de experiencia y tienen un conocimiento significativo en la formación de equipos. Además de entrenar a los equipos más complejos, los entrenadores principales evaluarán continuamente a los otros entrenadores y proporcionarán retroalimentación e ideas, o planes de acción, para mitigar cualquier carencia de habilidades. Este papel es más importante cuando se inicia un Dojo y se incrementa el personal de entrenamiento. Es común que,

inicialmente, se obtenga este puesto de un consultor externo más capacitado en los Dojos.

## **Gerente del Dojo (opcional)**

Este puesto de gerente se responsabiliza de la definición del caso de negocio para el Dojo, alineado con la estrategia empresarial más amplia y la agenda de la organización, a nivel de tecnológico. A través de una fuerte colaboración con las partes interesadas y los entrenadores del Dojo, el gerente trabaja para alcanzar la continua generación de valor por parte de los equipos participantes. Este puesto es, generalmente, responsable de establecer los mejores servicios, ofertas y prácticas que deben aplicarse en el Dojo, tomando como foco principal las necesidades concretas de la empresa.

## 3.4. Selección e ingesta de nuevos equipos

Basándose en los objetivos del Dojo, hay **que definir los criterios para la selección de los equipos** que serán buenos candidatos para este tipo de entrenamiento. Los equipos que, generalmente, se benefician más de la experiencia del Dojo tienen estas características:

- ▶ Un equipo multidisciplinar completo, un equipo de «dos pizzas» (ocho personas).
- ▶ Equipos que trabajan físicamente juntos (más adelante, en el tema, se analizan otras opciones para cuando esto no sea factible).
- ▶ Equipos que sean capaces de comprometerse, al menos, seis horas al día por un período de cuatro a ocho semanas consecutivas.
- ▶ Individuos que puedan comprometerse con el aprendizaje y que combinen la vocación Ágil con el foco en el producto y la tecnología.

Cuando se inicia un programa de Dojo, primero es recomendable trabajar con unos pocos equipos piloto que estén muy motivados con el objetivo de mejorar sus prácticas. Esto hará posible alcanzar el éxito desde una fase temprana y mostrarlo a toda la organización, lo que incrementará la confianza y el entusiasmo de muchos individuos.

Es necesario saber aprovechar cada oportunidad que se presenta **para detectar y «reclutar» agentes de cambio que se apasionen por los caminos del Dojo**. Estos individuos promoverán, aún más, el concepto de Dojo a través de la organización, incluso después de que termine la experiencia, lo que impulsará la futura demanda de los servicios de Dojo en otros equipos. Muchas empresas empiezan sus programas de Dojo usando estrategias «de arriba abajo» y «de abajo arriba» para construir un plan de ingesta de nuevos equipos para el Dojo.

Las organizaciones también pueden fijarse en las métricas de rendimiento para decidir que un equipo está listo para una experiencia de Dojo:

- ▶ Velocidad del equipo.
- ▶ Frecuencia de despliegue.
- ▶ Tiempo medio de recuperación (en siglas, MTTR).

Resulta útil tener una métrica que sirva de base para medir el desempeño de los equipos antes de demostrar cómo evolucionan esas métricas durante la experiencia en el Dojo y una vez que este acaba. A medida que los equipos avanzan, es recomendable incorporar métricas orientadas al valor de los productos que están construyendo, de modo que puedan asegurar que el mejor rendimiento se canaliza hacia los mejores negocios y clientes.

Los responsables deben tener en cuenta, sin embargo, que el uso de los Dojos como «palo» para corregir el mal rendimiento no dará resultados positivos. **Los Dojos deben ser vistos como un lugar inspirador para aprender las prácticas modernas de productos e ingeniería**, no como una fuente para que los equipos aceleren un objetivo de entrega a corto plazo.

Los Dojos no son iguales a las salas de guerra (*war rooms*). Si bien los primeros suelen contar con un fuerte apoyo de los ejecutivos, a veces hay dificultades para obtener la alineación de los mandos intermedios. Esto ocurre porque, generalmente, los mandos intermedios no saben priorizar el aprendizaje sobre la ejecución. Si se analiza el motivo que impulsa a este comportamiento, es probable que se deba a que la estructura de incentivos para un gerente se basa en la entrega: el Dojo requiere que se disminuya la velocidad a corto plazo para que, finalmente, a través del aprendizaje, los equipos puedan construir mejores productos y más rápidamente, a largo plazo.

Es por ello por lo que la empresa debe hacer una inversión específica (no estamos hablando de dinero exclusivamente, sino de tiempo para que haya un cambio de mentalidad) para ayudar a los líderes a comprender los cambios que están experimentando los equipos a medida que aprenden nuevas formas de organización del trabajo. Para conseguir este apoyo de los mandos intermedios, resulta útil proporcionar servicios centrados en el liderazgo dentro del Dojo, a fin de que también tengan ellos una experiencia de inmersión.

## Ciclo de vida y formato del Dojo

### Ciclo de vida del Dojo

El ciclo de vida cuenta con las siguientes etapas:

- ▶ **Consulta:** consiste en una breve reunión con el equipo interesado a fin de describir qué es un Dojo y cómo podría ayudar a ese equipo en concreto. El objetivo de esta reunión es el de dar a los equipos interesados suficiente información para decidir si el Dojo es una buena solución para ellos y para que los entrenadores evalúen si pueden cumplir con todos los requisitos necesarios de formación para el Dojo.
- ▶ **Acuerdo:** se trata de una reunión, de aproximadamente media jornada, en la que participa el equipo que desea entrar en el Dojo, así como cualquier otra parte interesada que resulte crítica. El equipo se alinea en sus resultados para el Dojo a través de la articulación de su campo de interés, metas, métricas clave, acuerdos de trabajo y una evaluación completa de las habilidades del equipo.
- ▶ **Experiencia del Dojo:** esta es la etapa más comprometida del ciclo de vida, ya que aquí se lleva a cabo la experiencia del Dojo, donde se trabaja directamente con los equipos, los entrenadores y otros interesados durante un período prolongado para alcanzar los objetivos de aprendizaje establecidos por la empresa.

- ▶ **De vuelta a la «naturaleza»:** el equipo retorna al entorno habitual de trabajo y es encuestado sobre su experiencia en el Dojo. Se espera que comparten sus vivencias y aquello que han aprendido con otras personas ajenas al Dojo. Los entrenadores hacen un seguimiento en las siguientes cuatro a seis semanas para comprobar que están aplicando lo aprendido durante el Dojo, solventar lagunas que puedan haber quedado de las sesiones e identificar nuevas oportunidades en otros equipos.

## Formatos de Dojo

Los Dojos pueden ofrecer variados formatos que se adaptan a los diferentes tipos de resultados de aprendizaje y limitaciones de los equipos. Algunos formatos comunes son:

- ▶ **Desafío:** los equipos entran en el Dojo con un reto propio diseñado a medida por ellos mismos. Estos Dojos deben incluir a todo el equipo: *product owner*, ingenieros, maestros de Scrum y diseñadores. El enfoque del desafío es una transformación holística de las prácticas y formas de trabajo de un equipo. Como resultado, un desafío es una actividad de larga duración (típicamente seis semanas) donde el equipo corre en *hiper-sprints* (típicamente dos días y medio cada uno). Al final del desafío, generalmente hay un evento con un grupo más amplio de líderes y partes interesadas, para exponer sus resultados.
- ▶ **FlashBuild:** este formato es más pequeño, suele durar menos de una semana y está pensado para equipos que intentan resolver un problema técnico muy específico, cómo establecer un pipeline de CI/CD, aplicar un *framework* de desarrollo o de pruebas, configurar servicios basados en nube o contenedores, etc. Los entrenadores expertos trabajan junto a los equipos para cocrear la solución, transfiriendo el conocimiento a lo largo del proceso de trabajo. Dada su corta duración, hay menos enfoque en el aprendizaje de las prácticas que determinan cómo los equipos deben trabajar con las nuevas tecnologías.

- ▶ **Talleres:** se trata de eventos, de una o varias horas, diseñados para educar a ingenieros específicos. Ejemplos de estos pueden ser eventos enfocados en migrar infraestructuras a la nube de forma eficiente, implantar Kubernetes, securizar el código, implementar el desarrollo basado en pruebas, etc.
- ▶ **Talleres de líderes:** es un taller de inmersión que no suele durar más que un par de días, que combina experiencias prácticas junto con una conferencia sobre los conceptos y prácticas fundamentales que los equipos están adoptando. El objetivo de estas sesiones no es que los líderes construyan productos completos, sino que obtengan la información necesaria para comprender los cambios por los que están atravesando sus equipos y empatizar con ellos. El foco de estas jornadas debe estar puesto en la enseñanza de conductas de liderazgo que permitan capacitar a los líderes de equipos, a fin de que este modelo sea exitoso.

## Ofertas básicas

Las empresas que organizan los Dojos suelen tener algunas ofertas básicas, públicamente accesibles para sus clientes, pero que además tengan la capacidad de personalizarse según las necesidades de cada equipo concreto. Algunas de estas ofertas podrían basarse en temas como: seguridad informática, CI/CD, DevOps, Cloud, o Agile. Estas ofertas básicas también pueden nombrarse con el objetivo o los resultados de aprendizaje del Dojo. Los ejemplos pueden ser: «Romper el monolito», «Responer a los incidentes de seguridad», etc.

Un Dojo efectivo es capaz de encontrar los puntos débiles de cada equipo, evaluar sus fortalezas y diseñar los objetivos de aprendizaje. La experiencia del Dojo es altamente personalizable y adaptable y, por tanto, **no es una buena práctica orientar a los Dojos en torno a un plan de estudios establecido, ya que resultaría demasiado rígido.** Al fin y al cabo, debemos recordar que el fin último del Dojo es el de cambiar la forma de trabajar de las personas, y no certificarlas en un conjunto de habilidades que no saben aprovechar eficazmente en la práctica.

## Variaciones del Dojo estándar

Numerosas investigaciones han demostrado **que la comunicación entre las personas funciona mejor cuando se lleva a cabo de forma presencial** y podríamos agregar, también, que el aprendizaje es más efectivo cuando la formación se lleva a cabo en persona. El hecho de que el Dojo se desarrolle en un espacio físico compartido permite proporcionar un ambiente seguro y controlado para el aprendizaje, facilita la construcción de una comunidad y conexiones, creando energía y sinergia. Sin embargo, dada la naturaleza global de los negocios, a veces no es posible que todos los miembros de un equipo puedan participar de una experiencia de Dojo. A continuación, hemos esbozado algunas variaciones para desarrollar el Dojo en otros formatos.

### Dojo virtual: Miembros de equipos remotos o no localizados

El mayor obstáculo o impedimento para el éxito en un formato de Dojo virtual es mantener a los participantes ocupados durante seis o más horas al día por teléfono o por teleconferencia. Como puedes imaginar, resulta un extremo complicado mantener al grupo concentrado y enfocado en una conversación a distancia cuando en su vida diaria desarrollan múltiples tareas a distancia, sin compartir con otros miembros la mayor parte del tiempo. El coach o entrenador deberá emplear técnicas especializadas que le permitan mantener el *momentum* a fin de que la atención del grupo no se disperse.

A nivel técnico, para garantizar el éxito del Dojo, será necesario **invertir en herramientas que faciliten altos niveles de colaboración y comunicación a distancia**. Algunas herramientas útiles en este contexto son Slack, Screenhero, Google Jamboard y Sococo. Además, algunas de las plataformas comunes de control código, como GitHub y GitLab, tienen capacidades de colaboración bastante avanzadas.

## Falta de espacio destinado a la formación

Como indica el encabezado, en estos casos la empresa no dispone de un espacio físico que permita llevar a cabo el Dojo. Esta realidad es común a muchas empresas cuando comienzan a trabajar con los Dojos. Resulta sencillo, al principio, organizar todo aquello que tiene que ver con los entrenadores, pero lo cierto es que lleva un poco más de tiempo asegurar y construir un espacio permanente. Para dar una solución a este problema, la mejor apuesta es **optimizar directamente el espacio de trabajo para que apalanque el cambio cultural**, con el objetivo de que los participantes estén rodeados por los principios del Dojo, como ocurriría en circunstancias normales.

## Dojo-in-a-Box

Este formato es similar a un desafío estándar de Dojo, pero en este caso, **los entrenadores del Dojo acuden al entorno de trabajo habitual de los equipos y no a un espacio dedicado a la formación** (es similar al caso de los entrenadores o coaches tradicionales de Agile). Aun así, los organizadores y entrenadores del Dojo aportarán materiales específicos, prácticas y enfoques que serán implantados en este entorno. Si bien esto puede funcionar cuando existen algunos lugares remotos que no se pueden gestionar fácilmente desde el Dojo, el hecho de que los equipos permanezcan en su entorno de trabajo habitual dificulta conducir una experiencia verdaderamente transformadora, ya que no están saliendo de su zona de confort.

## Dos por cuatro

Para los equipos que no están colocalizados, pero que quieren algunos de los beneficios de una experiencia de Dojo en persona, el «dos por cuatro» suele ser la mejor opción. Este formato es el mismo que el de un desafío de Dojo estándar, pero varía la cantidad de tiempo que el equipo comparte de forma conjunta. **Después de las primeras dos semanas permaneciendo juntos, todos los miembros del equipo regresan a su lugar de origen para atravesar las cuatro semanas**

**restantes del desafío.** Esta opción puede combinarse con los otros formatos alternativos descritos anteriormente.

Las empresas que han experimentado con este formato han visto muchos beneficios al reunir al equipo, incluso aunque solo se trate de las dos primeras semanas. Su naturaleza mixta permite que todos los miembros estén «en la misma página» en todas las fases de la experiencia y fortalece las relaciones interpersonales, algo que será fundamental a la hora de regresar a trabajar de forma remota.

## 3.5. Medición del éxito del Dojo

Los Dojos se establecen para ayudar a los equipos a abordar las nuevas tecnologías y procesos a través de una experiencia directa. A medida que los equipos trabajan con los problemas, adquieren nuevas habilidades y demuestran la capacidad de aplicarlas a sus propios proyectos y productos. A través de [una serie de entrevistas a varias empresas en la revista Fortune 500](#), se ha develado que estas utilizan una variedad de métodos para evaluar el éxito de esta experiencia.

### Delta AirLines

Por ejemplo, Delta Airlines usa, como su principal herramienta de evaluación, una lista de treinta preguntas (cuestionario) con respuestas «sí» o «no». El cuestionario abarca las capacidades que necesita una organización de software moderna para liberar nuevas versiones frecuentemente con bajas tasas de fallos. El cuestionario está disponible en línea y un facilitador se encarga de ayudar a quienes no estén familiarizados con la terminología. Comparando las respuestas antes y después del Dojo, la organización puede evaluar los progresos realizados.

Además del cuestionario, Delta AirLines adapta los objetivos del Dojo a cada equipo. Antes de que los equipos entren a un Dojo, completan una «fase de descubrimiento» donde identifican los objetivos y metas que pretenden alcanzar en el Dojo. Los equipos evalúan su propio progreso con respecto a los objetivos de la fase de descubrimiento.

### Corporación Target

En Target, los Dojos son una experiencia de aprendizaje opcional y los equipos deben inscribirse para participar. Los equipos definen sus propios objetivos y medidas de éxito en un acuerdo, incluyendo las metas que se pretenden completar en un solo Dojo. Los estatutos tienen cuatro secciones:

- ▶ Resumen: nombre del Dojo, duración y breve explicación de por qué es importante.
- ▶ Objetivos y métricas: los resultados y métricas de éxito deseados.
- ▶ Matriz de habilidades: habilidades explícitas que los individuos obtienen durante su participación.
- ▶ Acuerdos de trabajo: normas de comportamiento durante el Dojo.

La sección de «Metas y medidas» de uno de los acuerdos debe tener, por lo menos, una meta de aprendizaje y una entrega de trabajo real. El hecho de tener entregables de trabajo real significa que los equipos están haciendo cambios tangibles en los servicios y aplicaciones de producción. Esto convierte al Dojo en una experiencia productiva, alinea el aprendizaje en el Dojo con el trabajo diario y fomenta cambios significativos al tener un impacto directo.

Los participantes del Dojo pueden desempeñar diferentes funciones, desde el desarrollo de software hasta la gestión de productos o el control de calidad. Cada rol funcional tendrá, por tanto, diferentes necesidades y habilidades que adquirir. El mapa de habilidades indica explícitamente lo que cada participante obtendría. Por otra parte, los participantes del Dojo proceden de toda la organización y pueden provenir de diferentes equipos. Tener un conjunto de reglas explícitas establecerá normas de comportamiento y reunirá a los participantes en una mini comunidad. Los participantes del Dojo deben establecer sus propias reglas básicas y, a medida que progresan en el Dojo, serán libres de cambiarlas y actualizarlas.

Los equipos necesitan medio día para crear su acuerdo y estos suelen completarse unas semanas antes del Dojo. Los entrenadores dan apoyo y ayudan a los equipos a alcanzar los objetivos que han definido en sus estatutos, estos deben completar con éxito el 70 % de sus objetivos o más. Si un equipo no está en camino de completar sus metas, un entrenador intervendrá e iniciará un debate. En este, se propondrán medidas realistas que permitan desbloquear obstáculos y restablecer las expectativas.

## Mantener/sostener el Dojo

Es habitual lograr buenos resultados al principio y que, a pesar de tener esas mejoras que son inicialmente exitosas, el efecto termine por desaparecer o incluso fracasar debido a una inversión insuficiente o a una estructura que no es la apropiada para escalar y mantener los resultados en el tiempo. Aquellos que experimentan el Dojo necesitan tomarse el tiempo para **explicar el valor que crean los Dojos y cómo puede ayudar a la organización en conjunto, desde un punto de vista global, a alcanzar sus objetivos**. Muchas de las organizaciones comienzan de a poco, con dos o tres Dojos de equipos de vanguardia y el patrocinio de un solo ejecutivo. Estas primeras oportunidades deben aprovecharse para generar casos de éxito y testimonios que ayuden a impulsar la iniciativa. De esta manera, la financiación y los recursos serán más fáciles de conseguir.

La dotación de personal también puede ser un desafío. Los roles del Dojo deben ser rotativos, para que la gente pueda volver al trabajo y también para que aquellos que lo han experimentado pasen a ser entrenadores. Esto no es atípico ni incompatible con la forma en que los roles de entrenamiento se manejan en Agile o DevOps.

## DevOps Dojo: A Massive Internal Coaching Program – DXC Technology

IT Revolution. (2017, junio 22). *DevOps Dojo: A Massive Internal Coaching Program – DXC Technology* [Vídeo]. YouTube. [https://www.youtube.com/watch?v=75aZRf0\\_wSo](https://www.youtube.com/watch?v=75aZRf0_wSo)

En este vídeo podrás explorar la experiencia de la implantación real de una comunidad masiva de entrenadores y Dojos.

1. Selecciona la afirmación que mejor se corresponda con la definición de Dojo.

  - A. Es un espacio diseñado para albergar una experiencia de aprendizaje inmersiva donde los equipos acuden para aprender ingeniería moderna, productos y prácticas ágiles.
  - B. Es una herramienta de HashiCorp.
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.
2. Selecciona la afirmación que mejor se corresponda con el rol de entrenador de Dojo.

  - A. Es el encargado del primer equipo.
  - B. Es el coach que entiende el panorama general del estado inicial de los equipos y les ayuda a atravesar las etapas de aprendizaje «mostrándoles el camino».
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.
3. Selecciona la afirmación que mejor se corresponda con el concepto de Dojo in a box.

  - A. Es un proceso de Dojo normal, pero los entrenadores del Dojo acuden al entorno de trabajo habitual de los equipos.
  - B. Se refiere al espacio de reuniones recomendado para las demos y reuniones.
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.

- 4.** Selecciona la afirmación que mejor se corresponda con el reto de realizar un Dojo con equipos deslocalizados.
- A. No presenta ningún reto, es exactamente igual al Dojo presencial.
  - B. No presenta ningún reto si se tienen las herramientas de colaboración correctas.
  - C. El mayor reto viene dado por el hecho de tener seis horas diarias de interacción virtual con altos grados de atención y resultados.
  - D. Ninguna de las anteriores es correcta.
- 5.** Selecciona la afirmación que mejor se corresponda con la definición de oferta básica en el contexto de los Dojos.
- A. Es el catálogo estable de servicios de entrenamiento que ofrece la empresa organizadora del Dojo.
  - B. Es el servicio a coste reducido que ofrecen las empresas que forman en Dojo.
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.
- 6.** Selecciona la afirmación que mejor se corresponda con la definición de un taller de líderes en el contexto de los Dojos.
- A. Son talleres que enseñan a los miembros de los equipos a adquirir competencias de liderazgo.
  - B. Son talleres que pretenden enseñar a los mandos medios el proceso de transformación que están viviendo sus equipos.
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.

- 7.** Selecciona la afirmación que mejor se corresponda con la definición de un formato de Dojo basado en un desafío.
- A. Se le da este nombre al Dojo cuando es casi imposible que los equipos mejoren.
  - B. DevOps es siempre un desafío.
  - C. Ninguna de las afirmaciones es correcta.
  - D. Es el formato de Dojo donde el equipo trae sus propios objetivos y retos/reto.
- 8.** Selecciona la afirmación que mejor se corresponda con la descripción de los métodos de medición del éxito de los Dojos.
- A. La medición se realiza teniendo en cuenta el número de los equipos entrenados.
  - B. Es Agile, no hay métricas.
  - C. La medición se realiza teniendo en cuenta la cantidad de entrenadores que trabajan en la experiencia.
  - D. Cada organización suele usar sus propias métricas, pero casi todas suelen centrarse en el aumento de la eficiencia y la entrega de valor.
- 9.** Selecciona la afirmación que mejor se corresponda con la descripción ideal de un equipo para ser seleccionado para un Dojo.
- A. Comprometido, enfocado en producto, localizado en el mismo entorno y multidisciplinar.
  - B. Pequeño y tradicional.
  - C. A y B son correctas.
  - D. Ninguna de las anteriores es correcta.

**10.** Selecciona la afirmación que mejor se corresponda con la descripción del formato dos por cuatro.

- A. Ocho.
- B. Es el formato del Dojo que permite su implementación con las dos primeras semanas trabajando *in situ* y otras cuatro en el puesto de trabajo habitual, incluso si este es distinto para cada trabajador.
- C. Es una métrica de eficiencia de los Dojos.
- D. Ninguna de las anteriores es correcta.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 4. Tecnologías de virtualización

# Índice

[Esquema](#)

[Ideas clave](#)

[4.1. Introducción y objetivos](#)

[4.2. Virtualización de servidores](#)

[4.3. Tipos de hipervisores y productos](#)

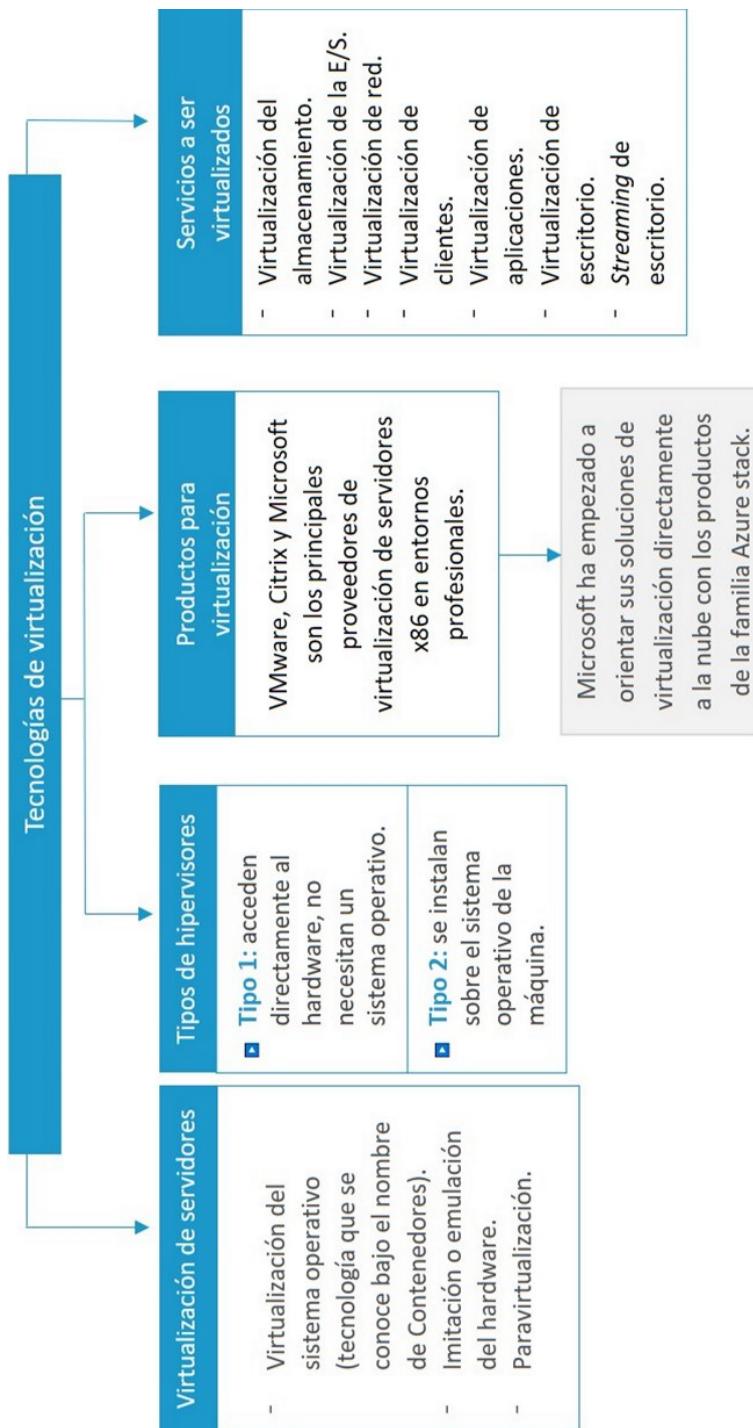
[4.4. Componentes y servicios susceptibles a ser virtualizados](#)

[4.5. Referencias bibliográficas](#)

[A fondo](#)

[Instalar y Configurar ESXi 6.7, 6.5 y 6 \(VMware vSphere\)](#)

[Test](#)



## 4.1. Introducción y objetivos

La virtualización tiene varios usos comunes y, lo cierto, es que todos ellos giran en torno al concepto de que su tecnología representa una abstracción de los recursos físicos. De hecho, existen demasiados tipos de virtualización y, es por ello por lo que, resulta un tanto confuso tratar de determinar cuál es la mejor solución que se puede aplicar en cada organización. Los dos tipos más comunes de virtualización aplicados a los centros de datos son: la virtualización de servidores y la virtualización de almacenamiento. Dentro de cada uno de estos, existen, además, diferentes enfoques o «sabores», cada uno con sus beneficios y desventajas. A continuación, vamos a explorar estos «sabores» básicos de las tecnologías de virtualización para que puedas descubrir las particularidades de cada uno y tomes decisiones estratégicas en la organización donde desarrolles tu actividad profesional.

Los objetivos de este tema son:

- ▶ Conocer las diferentes tecnologías existentes referentes a la virtualización.
- ▶ Conocer las diferentes herramientas que existen.
- ▶ Conocer los diferentes elementos susceptibles de ser virtualizados.

## 4.2. Virtualización de servidores

Existen tres tipos principales de virtualización de servidores que son: virtualización del sistema operativo (tecnología que se conoce bajo el nombre de Contenedores), imitación o emulación del hardware y paravirtualización. En este tema desarrollaremos los dos últimos.

### Imitación o emulación del hardware

En este caso, el **software de virtualización (denominado hipervisor)** crea una **máquina virtual que imita todo el entorno de hardware**. El sistema operativo, que está cargado en una máquina virtual, es un producto estándar no modificado. Cuando realiza llamadas para recursos del sistema, el software de emulación de hardware captura la llamada del sistema y la redirige para que pueda gestionar estructuras de datos proporcionadas por el hipervisor. Es el propio hipervisor el que realiza las llamadas al hardware físico real, subyacente a toda la aglomeración de software.

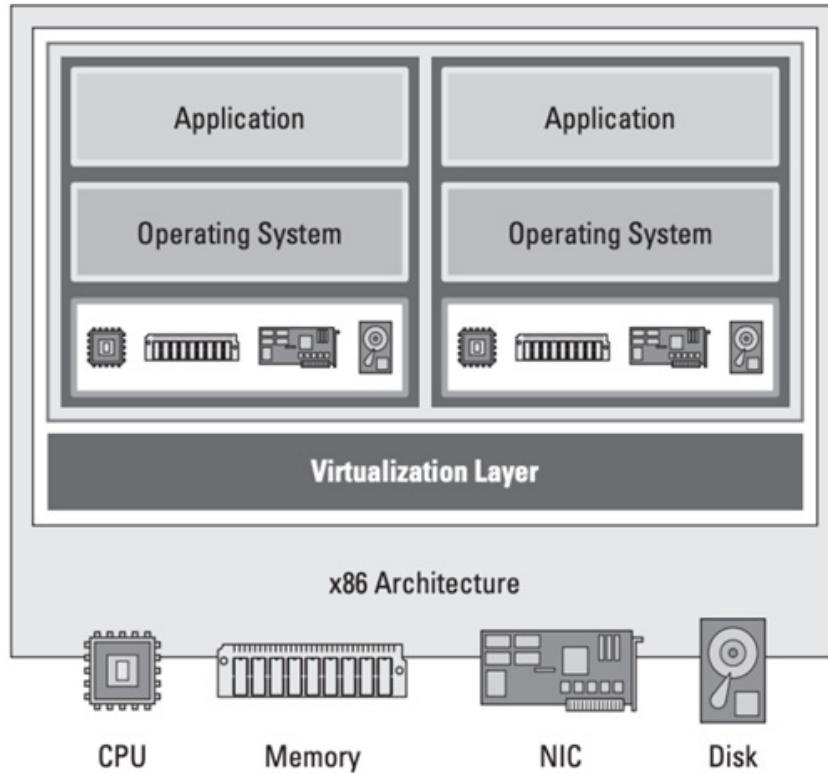


Figura 1. Virtualización por emulación de hardware. Fuente: Hewlett Packard Enterprise, s.f.

La emulación o imitación de hardware también es conocida como virtualización de metal desnudo (del inglés *bare metal virtualization*), para simbolizar el hecho de que ningún software se encuentra entre hipervisor y el «metal» del servidor. Como hemos mencionado, el hipervisor intercepta las llamadas del sistema desde la máquina virtual huésped y coordina el acceso al hardware subyacente directamente.

## Paravirtualización

La paravirtualización no intenta emular un entorno de hardware en software, sino que un hipervisor de paravirtualización coordina (o multiplexa) el acceso a los recursos de hardware subyacentes del servidor. La arquitectura de la paravirtualización se puede ver en la Figura 2, que representa a Xen.

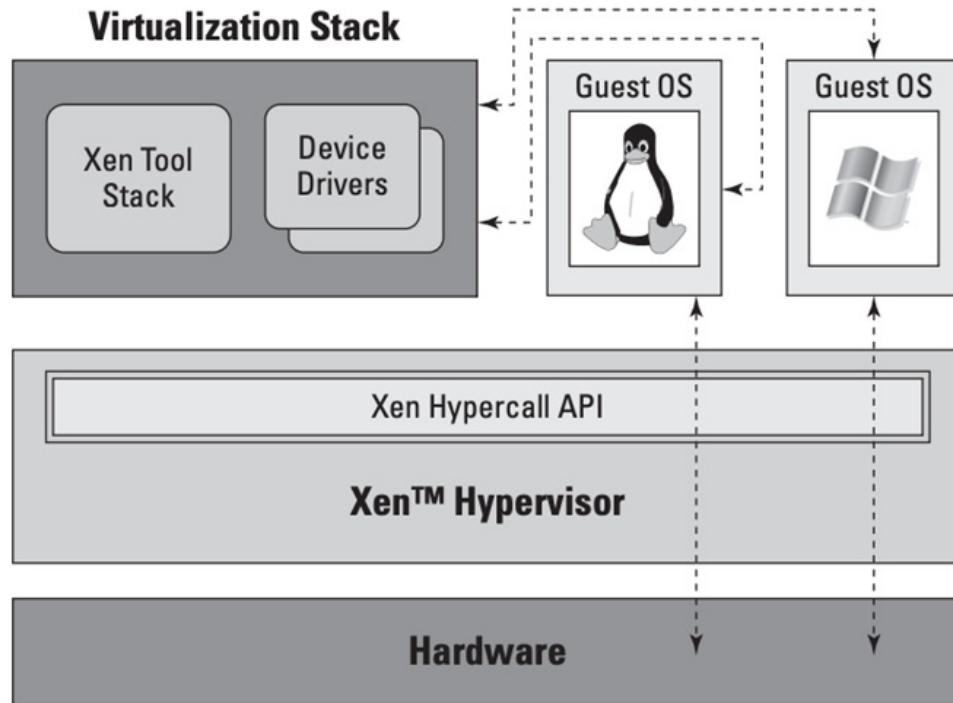


Figura 2. Estructura de la paravirtualización. Fuente: Hewlett Packard Enterprise, s.f.

**En la paravirtualización, el hipervisor reside en el hardware y, por lo tanto, esta se puede concebir como una arquitectura de virtualización de metal desnudo.** Uno o más sistemas operativos huésped (equivalente a máquinas virtuales en virtualización de emulación del hardware) se ejecutan sobre el hipervisor. Un huésped privilegiado se ejecuta como una máquina virtual huésped, pero tiene privilegios que le permiten acceder directamente a ciertos recursos en el hardware subyacente.

## 4.3. Tipos de hipervisores y productos

VMware, Citrix y Microsoft son los principales proveedores de virtualización de servidores x86 en entornos profesionales, más adelante veremos otras opciones susceptibles de ser usadas también en entornos personales y de prueba.

- ▶ **VMware:** es el proveedor de virtualización de servidores más extendido y afianzado en el mercado. La plataforma insignia de VMware, vSphere, utiliza la tecnología de emulación de hardware.
- ▶ **Citrix:** ofrece un producto de virtualización de servidor, llamado XenServer, basado en paravirtualización. El huésped privilegiado (llamado *control domain* en lenguaje Xen) y el hipervisor Xen trabajan en equipo para permitir que las máquinas virtuales huésped interactúen con el hardware subyacente.
- ▶ **Microsoft:** Hyper-V, el producto de virtualización del servidor de Microsoft, tiene una arquitectura muy similar a la de Xen. En lugar de usar el término *control domain* para referirse a las máquinas virtuales huésped, Hyper-V se refiere a ellas como particiones y a la contraparte del *control domain* de Xen se la denomina partición principal.

Una definición sencilla de hipervisor podría ser: la parte de la nube privada que gestiona las máquinas virtuales, es decir, es la parte (programa) que permite que múltiples sistemas operativos comparten el mismo hardware. Cada sistema operativo podría usar todo el hardware (procesador, memoria) si no hay otro sistema operativo encendido. Ese es el hardware máximo disponible para un sistema operativo en la nube. Sin embargo, **el hipervisor es lo que controla y asigna qué parte de los recursos de hardware debe obtener cada sistema operativo, para que cada uno obtenga lo que necesita y no se interrumpa entre sí.** Hay dos tipos de hipervisores:

- ▶ Hipervisor de tipo 1. Los hipervisores se ejecutan directamente en el hardware del sistema: un hipervisor integrado «básico».
- ▶ Hipervisor tipo 2. Los hipervisores se ejecutan en un sistema operativo *host* que proporciona servicios de virtualización, como soporte de dispositivos de E/S y administración de memoria.

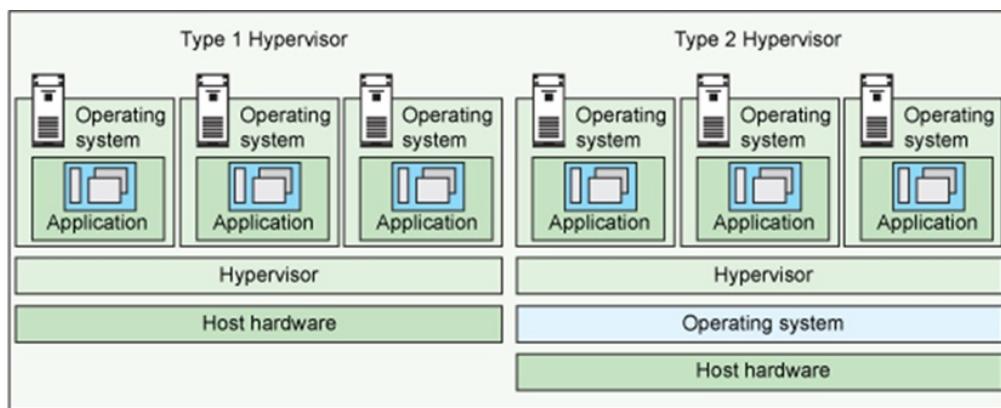


Figura 3. Diferencias entre los tipos de hipervisores. Fuente: Projekt Cloud Computing, s.f.

## Hipervisores tipo 1

### VMware ESX y ESXi

Estos hipervisores ofrecen funciones avanzadas y escalabilidad, pero requieren licencia, por lo que los costos son más altos. VMware ofrece algunos paquetes de menor costo y pueden hacer que la tecnología de hipervisor sea más asequible para infraestructuras pequeñas. VMware es el líder en los hipervisores tipo 1. Su producto vSphere ESXi está disponible en una edición gratuita y en cinco ediciones comerciales.

### Microsoft Hyper-V

El hipervisor de Microsoft, Hyper-V, no ofrece muchas de las funciones avanzadas que ofrecen los productos de VMware. Sin embargo, con XenServer y vSphere, Hyper V es uno de los tres principales hipervisores tipo 1. Se lanzó por primera vez

con Windows Server, pero ahora, Hyper-V se ha mejorado enormemente con Windows Server 2012 Hyper-V. Hyper-V está disponible tanto en una edición gratuita (sin GUI y sin derechos de virtualización) como en cuatro ediciones comerciales: Fundamentos (solo OEM), *Essentials*, *Standard* y *Datacenter Hyper-V*. Actualmente, las nuevas versiones de supervisores de Microsoft están íntimamente relacionadas a sus productos de *Cloud Platform* y se les conoce como *Azure Stack*.

## Citrix XenServer

Comenzó como un proyecto de código abierto. La tecnología principal del hipervisor es gratuita, pero al igual que ESXi gratuito de VMware, casi no tiene características avanzadas. Xen es un hipervisor de tipo desnudo de tipo 1 y, así como Red Hat Enterprise Virtualization usa KVM, Citrix usa Xen en el XenServer comercial.

Hoy, los proyectos y la comunidad de código abierto de Xen están en [Xen.org](http://Xen.org). Hoy, XenServer es una solución comercial de hipervisor tipo 1 de Citrix, que se ofrece en cuatro ediciones. Confusamente, Citrix también ha calificado sus otras soluciones propietarias como XenApp y XenDesktop con el nombre Xen.

## Oracle VM

El hipervisor Oracle se basa en el código abierto Xen. Sin embargo, si necesita soporte de hipervisor y actualizaciones de productos, le costará. Oracle VM carece de muchas de las características avanzadas que se encuentran en otros hipervisores de virtualización de metal desnudo.

## Hipervisores tipo 2

### VMware Workstation/Fusion/Player

VMware Player es un hipervisor de virtualización gratuito. Está destinado a ejecutar solo una máquina virtual (en siglas, VM) y no permite crear máquinas virtuales. VMware Workstation es un hipervisor más robusto con algunas características

avanzadas, como grabación y reproducción y compatibilidad con instantáneas de VM.

VMware Workstation tiene tres casos de uso principales:

- ▶ Para ejecutar múltiples sistemas operativos.
- ▶ Para ejecutar versiones diferentes de un sistema operativo en un escritorio.
- ▶ Para desarrolladores que necesitan entornos de sandbox e instantáneas, o para laboratorios y con fines de demostración.

## Servidor VMware

VMware Server es un hipervisor de virtualización alojado gratuito que es muy similar a VMware Workstation. VMware ha detenido el desarrollo en el servidor desde 2009.

## Microsoft Virtual PC

Esta es la última versión de Microsoft de esta tecnología de hipervisor, Windows Virtual PC y solo se ejecuta en Windows 7 y solo es compatible con los sistemas operativos Windows que se ejecutan en él.

## Oracle VM VirtualBox

La tecnología de hipervisor VirtualBox proporciona un rendimiento y características razonables si desea virtualizar con un presupuesto limitado. A pesar de ser un producto alojado gratuito con una huella muy pequeña, VirtualBox comparte muchas características con VMware vSphere y Microsoft Hyper-V.

## **Red Hat Enterprise Virtualization**

La máquina virtual basada en el kernel (en siglas, KVM) de Red Hat tiene cualidades tanto de un hipervisor de virtualización alojado como virtual. Puede convertir el núcleo de Linux en un hipervisor para que las máquinas virtuales tengan acceso directo al hardware físico.

## **KVM**

Esta es una infraestructura de virtualización para el kernel de Linux. Admite la virtualización nativa en procesadores con extensiones de virtualización de hardware. El KVM de código abierto (o máquina virtual basada en el núcleo) es un hipervisor tipo 1 basado en Linux que se puede agregar a la mayoría de los sistemas operativos Linux (incluidos Ubuntu, Debian, SUSE y Red Hat Enterprise Linux) pero también Solaris y Windows.

## 4.4. Componentes y servicios susceptibles a ser virtualizados

### Virtualización de almacenamiento

La cantidad de datos que las organizaciones están creando y almacenando ha alcanzado cifras récord. Este repentino crecimiento en los requisitos de almacenamiento ha hecho que la virtualización del almacenamiento sea cada vez más importante.

La virtualización del almacenamiento puede ser entendida como el proceso de abstracción lógica del almacenamiento físico. Los recursos de almacenamiento físico (como las unidades de disco) se agregan en agrupaciones de almacenamiento, desde donde el almacenamiento lógico se crea y se presenta al entorno de aplicación. Esta se puede implementar dentro del almacenamiento de matrices en sí misma (virtualización basada en matrices) o bien en el nivel de red, donde múltiples matrices de discos o almacenamiento en red de sistemas de diferentes proveedores, dispersos por la red, se pueden agrupar en un único dispositivo de almacenamiento monolítico. Esto permite que las matrices múltiples se administren de manera uniforme como si se tratase de un solo grupo.

La virtualización basada en matrices ofrece más flexibilidad, simplifica la gestión y mejora el rendimiento y la utilización de la capacidad en comparación con las matrices de discos tradicionales.

Hay dos tipos principales de sistemas de almacenamiento en red con virtualización de almacenamiento: los sistemas *Network Attached Storage* (en siglas, NAS) y *Storage Area Network* (en siglas, SAN). Estos sistemas de almacenamiento compartido (SAN y NAS) permiten aprovechar las capacidades avanzadas de virtualización del servidor como la migración de máquinas virtuales en vivo, la alta disponibilidad, la tolerancia a fallos y la recuperación ante siniestros.

## Almacenamiento conectado a la red (NAS)

El almacenamiento conectado a la red, o NAS, es un **dispositivo de almacenamiento que se encuentra en la propia red y ofrece almacenamiento a los servidores en la red**. Permite que múltiples clientes, como usuarios de PC y servidores, compartan archivos a través de una red de área local (en siglas, LAN). NAS utiliza archivos basados en protocolos como *Network File System* (en siglas, NFS), *Server Message Block* (en siglas, SMB) o *Common Internet File System* (en siglas, CIFS), donde el almacenamiento es remoto y las ordenadoras solicitan un archivo en lugar de un bloque de disco. El hecho de tener todos los archivos movidos a una única ubicación central hace que se simplifique la administración de estos. Es decir que, en lugar de tener que hacer un seguimiento de todos los archivos repartidos entre docenas, cientos, o incluso miles de máquinas, todos los datos se encuentran en un lugar, lo que permite realizar una mejor copia de seguridad, archivado, etc. Una de las ventajas más acusadas del NAS es que está basado en *Internet Protocol* (en siglas, IP) y es fácil de usar, desplegar y gestionar. Los usos más comunes incluyen archivos de rápido almacenamiento para *rich media*, documentos y archivos de *back-up* y correo electrónico.

## Redes de área de almacenamiento (SAN)

Una red de área de almacenamiento, o SAN, es un **dispositivo de almacenamiento (como una matriz de discos o una biblioteca de cintas) accesible a los servidores para que los dispositivos puedan permanecer localmente conectados al sistema operativo**. La SAN, normalmente, tiene su propia red de dispositivos de almacenamiento a los que usualmente no se puede acceder a través de una red ordinaria de dispositivos. Una SAN sola no proporciona la abstracción del «archivo» como en el caso de NAS, sino que solo provee operaciones a nivel de bloque.

La mayoría de las SAN usan la conectividad del canal de fibra, del inglés *Fibre Channel Connectivity* (en siglas, FC), una tecnología de red especialmente diseñada para gestionar comunicaciones de almacenamiento, o Internet SCSI (en siglas, iSCSI), que es un estándar de red basado en IP para vincular dispositivos de almacenamiento.

Las empresas utilizan almacenamiento este tipo de almacenamiento para centralizar la gestión de los datos corporativos. Los usos comunes de una SAN incluyen el aprovisionamiento de datos de acceso transaccional que requieren acceso a nivel de bloque de alta velocidad a los discos duros de almacenamiento, tales como servidores de correo electrónico, bases de datos y servidores de archivos de acusado uso.

## Virtualización de E/S

La virtualización de servidores involucra a máquinas que funcionan en un servidor físico, lo que hace posible la ejecución de múltiples máquinas virtuales en un único sistema físico. **La virtualización del almacenamiento permite que los datos se transfieran a un almacenamiento centralizado y compartido, lo que permite que se gestione de manera eficiente y rentable.** Sin embargo, obtener datos de la máquina requiere pasar por *endpoints* de red y almacenamiento en el servidor, lo que puede traer aparejado otro conjunto de problemas.

¿De qué sirve tener máquinas virtuales y almacenamiento (que se pueda virtualizar y migrar, según sea necesario) cuando los dispositivos físicos de *endpoint* de entrada y salida (en adelante, E/S) que residen en el servidor son no son ágiles? La administración manual de un recurso clave en un entorno virtualizado significa que la eficiencia está reñida con la capacidad que tenga esa organización de TI para administrar manualmente estos dispositivos de E/S.

## Virtualización de red

Si todo lo demás está virtualizado, está claro que la red también debe volverse más ágil y debe ser susceptible de ser administrada como un recurso virtual, en lugar de requerir actividad manual para responder a las cargas de trabajo o a las condiciones cambiantes de negocio. Como es de esperar, **la virtualización también se ha trasladado a la red y, en lugar de realizar cambios en la red moviendo cables entre diferentes recursos físicos, la tecnología de virtualización se aplica a la red en sí misma.**

La virtualización de red permite que la red se configure sobre la marcha, sin necesidad de tocar ni un solo cable o dispositivo. Por tanto, los dispositivos de red con capacidad de virtualización se administran de forma remota y se pueden reconfigurar lógicamente.

Esta capacidad de realizar modificaciones en la red de forma remota (y lógica) completa la virtualización del centro de datos. Cada tipo de recurso, desde el servidor hasta el almacenamiento y todo lo demás, ya no está físicamente vinculado a piezas específicas de hardware. Por el contrario, cada tipo de recurso puede ser abordado lógicamente y reconfigurado sin necesidad de configurarlo físicamente.

## Virtualización de clientes

La virtualización no resuelve todos los problemas de las organizaciones de TI. Actualmente, se utilizan una gran cantidad de dispositivos cliente en las organizaciones. En muchas empresas, casi todos los empleados tienen su propia PC (ya sea un dispositivo de escritorio o un ordenador portátil), además de teléfonos inteligentes y blocs de notas.

Mantener todos esos dispositivos actualizados con parches en el sistema operativo, actualizaciones de aplicaciones, antivirus y *spyware*, etc., es una tarea prácticamente interminable. Se hace aún más difícil por el hecho de que estas máquinas están localizadas con el usuario, en oficinas y hasta en lugares de trabajo temporal como

Starbucks. Hacer un seguimiento de los dispositivos y garantizar que estén seguros ha impulsado el movimiento hacia la virtualización de clientes.

## Virtualización de aplicaciones

**La virtualización de aplicaciones es una separación de la ejecución del programa y de la visualización de este.** En otras palabras, un programa como Microsoft Word se ejecuta en un servidor ubicado en el centro de datos, pero la salida gráfica se envía a un dispositivo cliente remoto. El usuario final ve la pantalla gráfica completa del programa y puede interactuar con él a través del teclado y el ratón.

Una variante de la virtualización de aplicaciones es aquella en la que la aplicación no se ejecuta en un servidor en el centro de datos, sino en el dispositivo del cliente. La diferencia con el modo tradicional de uso de la aplicación radica en la forma en la que se administra la aplicación: en lugar de instalarse permanentemente en el dispositivo del cliente, se envía (en modo *streaming* o transmisión) al dispositivo del cliente cada vez que se ejecuta. Este modo de «instalación en cada uso» puede parecer repetitivo, pero permite que una organización de TI controle mejor la aplicación, para garantizar que se mantenga actualizada con versiones, parches y todo lo demás.

## Virtualización de escritorio

A diferencia de la virtualización de aplicaciones, donde una o más aplicaciones se muestran o se transmiten desde un servidor central, **en la virtualización de escritorio el ordenador del usuario se ejecuta en un servidor central, con la salida gráfica de la pantalla a un dispositivo cliente.** Este tipo de virtualización se conoce como VDI (sigla del inglés, *Virtual Desktop Infrastructure*) que significa infraestructura de escritorio virtual. La ventaja de este enfoque es que es más fácil mantener actualizados los sistemas del cliente con parches, etc. Esto se debe a que, en lugar de administrar sistemas individuales que se encuentran por aquí y por allá,

los equipos de TI pueden administrarlos en una ubicación centralizada.

En los últimos años, la virtualización de escritorios ha evolucionado con creces. En lugar de tener que almacenar una imagen de escritorio para cada usuario, usa una sola imagen que se clona según sea necesario. Supongamos que para 4000 usuarios se necesitan 4000 imágenes de disco, por lo que podemos deducir que haría falta una gran capacidad de almacenamiento, a pesar de que gran parte de cada imagen es idéntica a la otra. Por tanto, deducimos que esta eficiente clonación reduce enormemente la necesidad de almacenamiento y hace que la virtualización de escritorio sea muy atractiva desde un punto de vista «económico».

Pues bien... podríamos preguntarnos qué sucede con nuestra aplicación favorita que ejecutamos a diario en nuestro ordenador. ¿Desaparecería con la virtualización de escritorio? La respuesta es no. **Los datos y configuraciones se pueden guardar por separado y se aplican a la imagen clonada, para garantizar que cada usuario tenga sus aplicaciones y datos listos para cuando abra su escritorio.**

La virtualización de escritorio, a menudo, utiliza un dispositivo cliente económico para la visualización e interacción con el usuario final. Estos «clientes ligeros», como se les suele llamar, pueden ser dispositivos baratos con poca potencia, desde un punto de vista informático, y sin almacenamiento en disco local. Esto puede reducir significativamente el coste por dispositivo de usuario final: el hardware es menos costoso, pero generalmente usan menos energía, ocupan menos espacio y requieren menos soporte.

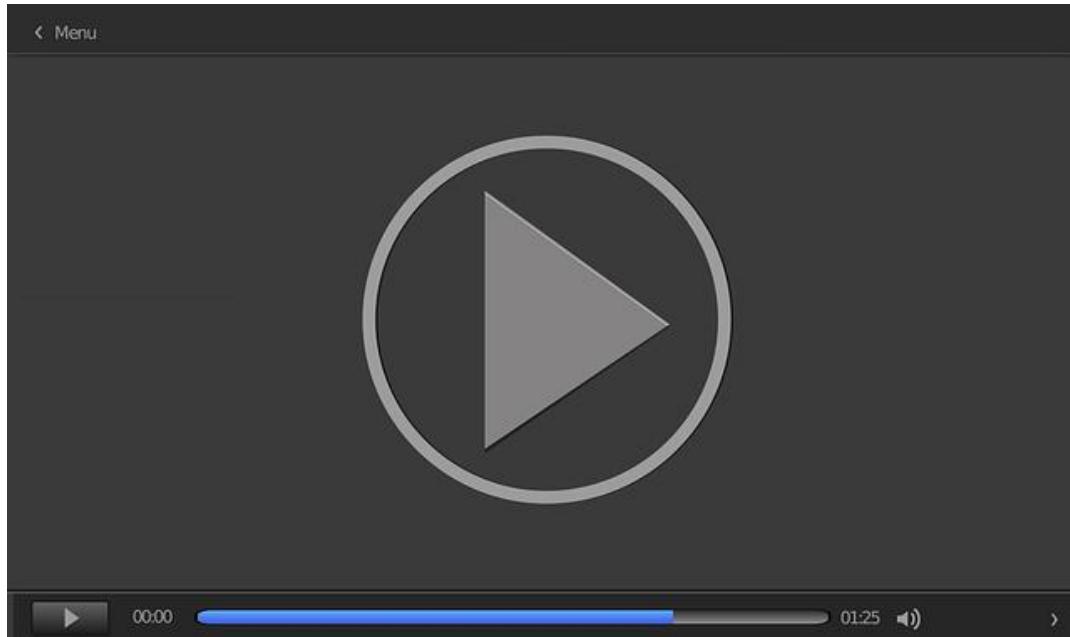
## Streaming de escritorio

Una variante de la virtualización de escritorio es el *check-in, check-out* (entrada y salida) cliente. De esta forma, el almacenamiento continuo del sistema cliente está centralizado, pero cuando el usuario final está listo para comenzar a trabajar, el sistema del cliente se transfiere al dispositivo cliente y se utiliza como un ordenador tradicional. **Cuando finaliza la sesión, el usuario apaga el sistema y la imagen**

**del ordenador se escribe en el repositorio central y no queda nada en el hardware del usuario.**

Esta forma de entrada y salida de virtualización cliente está comenzando a explorarse, pero es muy prometedora para los entornos donde la disponibilidad de conectividad de red de alta velocidad es incierta. Por ejemplo, cuando alguien trabaja de forma remota desde casa, puede que no esté claro si su conexión es lo suficientemente sólida como para permitir la virtualización de aplicaciones o la virtualización de escritorio tradicional. En estos casos, una única descarga del escritorio a un dispositivo cliente puede ser una buena opción.

La virtualización ha dado paso a las tecnologías basadas en la nube, posteriormente a los contenedores y, en los últimos años, el concepto *serverless* se ha instaurado como una nueva forma de construir aplicaciones de forma más eficiente, económica y efectiva. Ingresá al video *¿Qué es Serverless?* para desarrollar más este concepto.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=b7297eb1-360a-4627-b93e-ad78013d2d4b>

---

## 4.5. Referencias bibliográficas

Projekt Cloud Computing. (s.f.). *Two types of hypervisors.* <https://oze.pwr.edu.pl//kursy/introcloud/content/start/K-04-03.html>

Hewlett Packard Enterprise. (s.f.). *Introducción a la virtualización de HP.* <https://www.hpe.com/>

### Instalar y Configurar ESXi 6.7, 6.5 y 6 (VMware vSphere)

TKapacito TIC. (2018, septiembre 24). *Instalar y Configurar ESXi 6.7, 6.5 y 6 (VMware vSphere)* [Vídeo]. YouTube. <https://www.youtube.com/watch?v=WARIBT97Z0w>

Este vídeo ayuda a instalar una solución de virtualización de VMware, que puede utilizarse para profundizar en los conocimientos prácticos de virtualización.

1. Selecciona la afirmación que mejor se corresponde con la definición de virtualización de almacenamiento.

  - A. Es el proceso de abstracción lógica del almacenamiento físico.
  - B. Son los ficheros de las máquinas virtuales.
  - C. A y B son correctas.
  - D. Ninguna es correcta.
  
2. Selecciona la afirmación que mejor se corresponde con definición de virtualización de escritorio.

  - A. Es la posibilidad de tener sistemas de virtualización en un PC de escritorio.
  - B. Es el proceso de convertir los PC de escritorio en máquinas virtuales.
  - C. A y B son correctas.
  - D. Ninguna es correcta.
  
3. Selecciona la afirmación que mejor se corresponde con virtualización del sistema operativo.

  - A. Este tipo de virtualización se corresponde a lo que hoy en día llamamos Contenedores (Containers).
  - B. Es la virtualización de máquinas virtuales.
  - C. A y B son correctas.
  - D. Ninguna es correcta.

4. Selecciona la afirmación que mejor se corresponde con la definición de «paravirtualización».

  - A. Se llama así a la virtualización implementada sobre una segunda capa de virtualización.
  - B. Se llama así a la virtualización anidada.
  - C. La paravirtualización no intenta emular un entorno de hardware en software.
  - D. Ninguna es correcta.
5. Selecciona la afirmación que mejor se corresponde con la virtualización de red.

  - A. Es el proceso de abstracción lógica de la capa de red.
  - B. Es el servicio que facilita la transferencia de máquinas virtuales por la red.
  - C. A y B son correctas.
  - D. Ninguna es correcta.
6. Selecciona la afirmación que mejor se corresponde con la virtualización de aplicaciones.

  - A. Son máquinas virtuales especializadas en la ejecución de aplicaciones.
  - B. Es una separación de la ejecución del programa y de la visualización de este.
  - C. A y B son correctas.
  - D. Ninguna es correcta.

- 7.** Selecciona la afirmación que mejor se corresponde con la definición de hipervisor de tipo 1.
- A. Son los hipervisores que funcionan sobre un sistema operativo.
  - B. Son el tipo de hipervisor más antiguo.
  - C. Ninguna es correcta.
  - D. Son los hipervisores que pueden ejecutarse directamente sobre el hardware sin la necesidad de un sistema operativo.
- 8.** Selecciona la afirmación que mejor se corresponde con la descripción de VirtualBox.
- A. Es un software de virtualización propiedad de Oracle.
  - B. Es un software de virtualización de aplicaciones.
  - C. Es un hipervisor de tipo 2.
  - D. A y C son verdaderas.
- 9.** Selecciona la afirmación que mejor se corresponde con la definición de *streaming* de escritorio.
- A. Es proceso de capturar la entrada y salida de un escritorio (virtual o físico).
  - B. Utilizar servicios de streaming desde el escritorio.
  - C. A y B son correctas.
  - D. Ninguna es correcta.

10. Selecciona la afirmación que mejor se corresponde con la descripción de KVM.
- A. Es el *Keyboard Virtual Module* que permite el uso de teclados en máquinas virtuales.
  - B. Es el sistema de virtualización incorporado en la mayoría de los sistemas Linux.
  - C. Es un producto de virtualización de VMware.
  - D. Ninguna es correcta.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 5. Casos de uso en virtualización

# Índice

[Esquema](#)

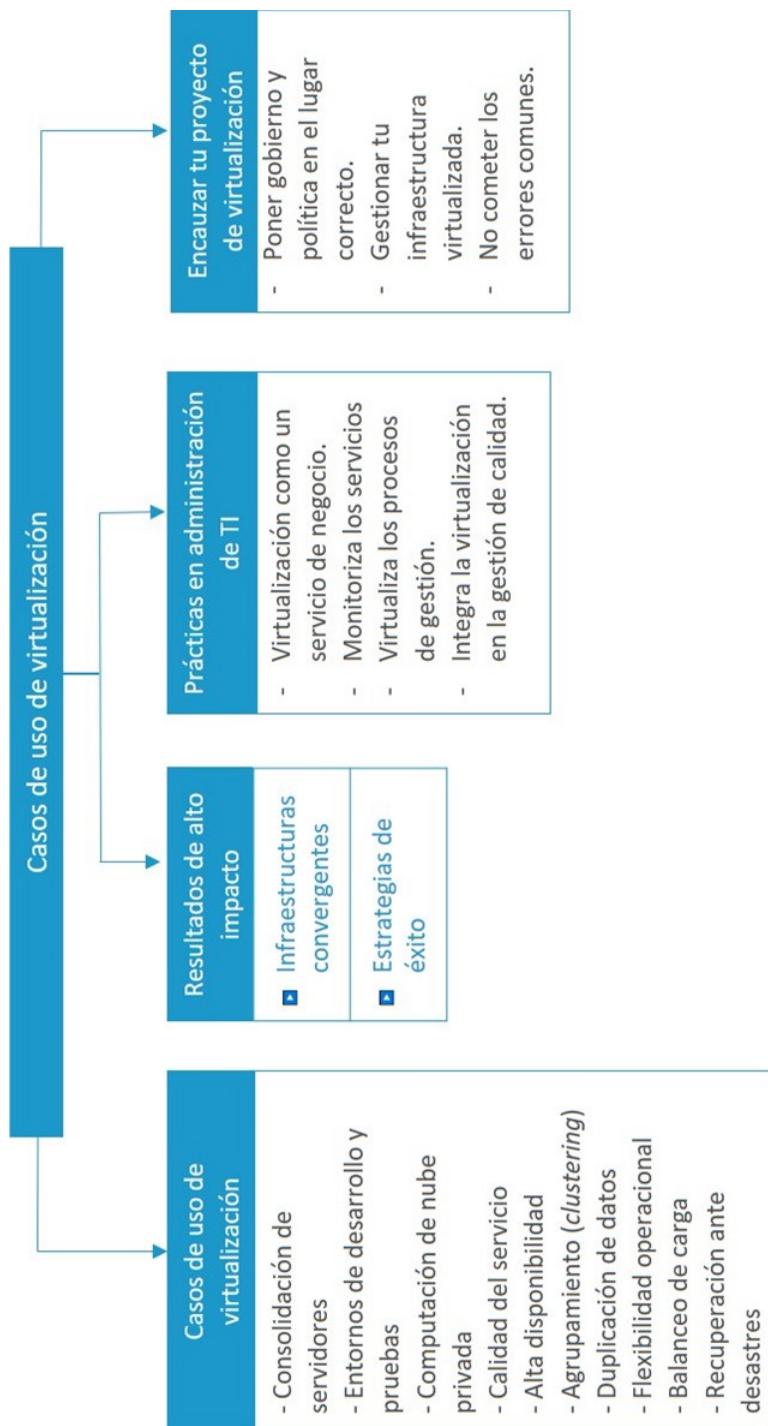
[Ideas clave](#)

- [5.1. Introducción y objetivos](#)
- [5.2. Casos de uso de virtualización](#)
- [5.3. Resultados de alto impacto](#)
- [5.4. Buenas prácticas en administración de TI](#)
- [5.5. Cómo encauzar tu proyecto de virtualización](#)

[A fondo](#)

[Caso éxito virtualización - MAPFRE](#)

[Test](#)



## 5.1. Introducción y objetivos

Resulta importante mostrar ejemplos de aplicación de la virtualización para complementar los conocimientos técnicos y teóricos de su funcionamiento y ver las formas de aplicarla de manera exitosa.

Siguiendo con esta idea, los objetivos que pretendemos alcanzar con este tema son:

- ▶ Entender las áreas más comunes donde la virtualización es beneficiosa.
- ▶ Conocer ejemplos exitosos de virtualización.
- ▶ Analizar la importancia de la gobernanza de tecnología de información (en adelante, TI) en conjunción con la virtualización.

## 5.2. Casos de uso de virtualización

La virtualización es útil en varios escenarios, que pueden ser simples o extremadamente complejos. Lo que resulta importante es comprender cómo se desea utilizar la virtualización, ya que esto determinará qué solución es la más adecuada para cada circunstancia u organización. Para ello, vamos a repasar algunos de sus principales usos.

### Consolidación de servidores

El primer caso de uso de la virtualización suele ser la consolidación de servidores. Esta se refiere a **tomar instancias separadas del servidor y migrarlas a máquinas virtuales que se ejecutan en un único servidor**. Pero además de ello, la consolidación de servidores también puede ser entendida como el acto de coger numerosos servidores separados y migrarlos a menos servidores, con múltiples máquinas virtuales ejecutadas en cada servidor.

Las empresas que implementan la consolidación de servidores experimentan una profunda transformación: pasan de ejecutar 150 servidores físicos a ejecutar 150 máquinas virtuales en solo quince servidores, lo que redundá en una enorme reducción de costes e inversión de hardware, energía y refrigeración, empleados, tiempo y, en muchos casos, costes de licencias de software. Por tanto, **podemos afirmar que la consolidación de servidores puede llegar a ofrecer un 60 % u 80 % de mejora de utilización de recursos**.

### Entornos de desarrollo y pruebas

Vamos a imaginar que un ingeniero construye un software capaz de implementar una determinada funcionalidad. Antes de pasar a la siguiente fase, el software se ha de ejecutar y probar en una variedad de sistemas (por ejemplo, Windows y Linux), así como varias versiones de esos productos. Entonces, un departamento de calidad, o *quality assurance* (en siglas, QA), realiza todas las pruebas necesarias para

garantizar que cumple con los requisitos aplicables de funcionalidad, escalabilidad, robustez, y detectar errores.

Gracias a la implementación de la virtualización, un desarrollador o *tester* puede replicar un entorno distribuido con varios sistemas en una sola pieza de hardware. Esto permite librarse de la necesidad de tener un montón de servidores aparcados, esperando que los *testers* o desarrolladores necesiten utilizarlo.

A su vez, la virtualización también es útil en entornos de prueba y desarrollo por otro motivo. Uno de los efectos secundarios del software de prueba es que las primeras versiones a menudo se bloquean y dañan no solo a la aplicación, sino también al funcionamiento del sistema subyacente, así como a otras aplicaciones en la pila de software y, para poder iniciar la recuperación, es necesario reinstalar todo el software. Resulta evidente que esto representa un verdadero impedimento para la productividad. Pues bien, a través de la virtualización esto puede evitarse ya que solo la aplicación de prueba, la máquina virtual y el software relacionado con esta se ven afectados.

## Computación en la nube privada

En pocas palabras, la computación en la nube es un medio para proporcionar servicios de tecnología bajo demanda a través de Internet. **Una nube privada es un entorno donde estos servicios operan para una única organización.** Numerosas organizaciones de TI están explorando cómo construir sus propias nubes privadas para aumentar la agilidad y reducir sus costes. La virtualización es un componente clave de las nubes privadas porque permite un rápido aprovisionamiento y desaprovisionamiento de servicios bajo demanda.

## Calidad de servicio

Las organizaciones de TI deben centrarse en la calidad de los servicios que brindan, es decir, **el buen mantenimiento de las aplicaciones y la infraestructura**

**subyacente.** Cuando la virtualización se gestiona adecuadamente, puede ayudar a mejorar la calidad del servicio porque elimina la dependencia del hardware. Además, la virtualización de los sistemas permite responder con más rapidez a todo tipo de fallos (sean estos de hardware, de red), incluso si son causados por el software de virtualización. También, se puede implementar de forma preventiva para evitar fallos al mover cargas de trabajo de un sistema que presenta signos de problemas inminentes (memoria, disco, etc.).

## Comutación por error

El hipervisor monitoriza constantemente el estatus de cada máquina virtual, por lo que es relativamente sencillo configurarlo para iniciar una nueva instancia de una máquina virtual, en caso de que se observe que una que se estaba ejecutando anteriormente ya no está presente. El hipervisor, simplemente, tiene que iniciar una máquina virtual nueva basada en la imagen de la máquina virtual anterior. Este proceso puede durar apenas unos segundos y, obviamente, representa una importante mejora sobre la duración típica de restauraciones de sistemas no virtualizados.

## Alta disponibilidad

La alta disponibilidad o *high availability* (en siglas, HA) extiende el concepto de comutación por error **al incorporar un servidor de hardware adicional**. Es decir que, en el caso de que la máquina virtual fallase, esta no se iniciaría en la misma pieza de hardware, sino que se iniciaría en un servidor diferente, evitando así el problema de comutación por error de virtualización por adición del hardware.

### ¿Cómo funciona la alta disponibilidad?

¿Cómo es posible que un hipervisor, en un servidor físico, inicie una máquina virtual en otro hipervisor? La respuesta es clara: no es posible. No puede. La alta disponibilidad se basa en un software de virtualización global que coordina los esfuerzos de múltiples hipervisores. Cuando una máquina virtual en un servidor de

hardware falla, el software de coordinación inicia una nueva máquina virtual en un servidor de hardware separado.

En realidad, es un poco más complejo que eso. El software de coordinación de virtualización monitoriza constantemente todos los hipervisores y sus máquinas virtuales. Si el software de coordinación ve que el hipervisor en un servidor ya no responde, reinicia las máquinas virtuales del hardware que ha fallado en otro hardware. Por lo tanto, **la alta disponibilidad aborda el problema del fallo del hardware mediante el uso de un software de virtualización de mayor nivel para coordinar los hipervisores en dos o más máquinas, a través de la monitorización continua y el reinicio de máquinas virtuales en otras máquinas si es necesario.**

Como se puede apreciar, esto ciertamente aborda el problema de fallo del hardware y hace que la conmutación por error sea más robusta. Parte del estado de una máquina virtual es su dirección de red y sus recursos de almacenamiento. Si se mueve una máquina virtual a otra pieza de hardware, estos *bits* de su estado necesitan moverse también, ya que, en caso contrario, la nueva máquina virtual no podrá ubicar el almacenamiento ni conectarse a las redes. Por lo tanto, la alta disponibilidad requiere que el software de virtualización pueda migrar estas partes (estados) de la máquina virtual a otro servidor físico y configurar el hipervisor de ese servidor para usar el estado de la máquina virtual del hardware original que ha fallado. Algunos programas (software) de alta disponibilidad incluso tienen la capacidad de ver lo que sucede dentro de una máquina virtual y reiniciar la aplicación que se estaba ejecutando en otra máquina virtual distinta.

**La HA proporciona una capa adicional de protección contra la conmutación por error a través del software de virtualización.** Sin embargo, la HA no proporciona la capacidad para migrar el estado actual de la memoria de la máquina virtual a la segunda máquina. Dicho de otra manera, aunque se pueda ejecutar una máquina

virtual en una segunda máquina, los usuarios que trabajan en la máquina virtual original perderán el estado de su trabajo. Dependiendo de qué tipo de aplicación se trate, esa pérdida podría ser inaceptable. Por ejemplo, si la solicitud está procesando transacciones multimillonarias en dólares estadounidenses, una pérdida mínima puede impactar significativamente a la organización.

## Agrupamiento o *clustering*

**El agrupamiento está diseñado para garantizar que no se pierdan datos en caso de que haya un fallo de software o de hardware.** El *clustering* ha sido ofrecido, históricamente, por los proveedores de aplicaciones como complemento de otros productos, pero con algunos inconvenientes relacionados: gastos adicionales, soluciones redundantes e infraestructura compleja. El gasto extra, o adicional, se produce por la necesidad de contar con hardware adicional, con el sistema en espejo en espera (en modo *standby*), listo para asumir el control si fallase el sistema primario. Como verán, no hace falta ser un experto para reconocer que la compra de un segundo conjunto de hardware hace que el agrupamiento represente un gasto significativo. Sin embargo, si en el sistema se operan transacciones de millones de euros (o cualquier otra moneda), mantener un servidor redundante que esté listo para operar cuando haya un fallo, puede ser una inversión que valga la pena.

## ¿Cómo funciona el agrupamiento?

Esencialmente, **el software de coordinación de virtualización ejecuta dos máquinas virtuales en máquinas separadas.** Las máquinas virtuales son idénticas en cuanto al sistema operativo y la configuración de la aplicación, pero difieren, naturalmente, en los detalles de sus conexiones de red y hardware local. El supervisor de virtualización se comunica constantemente con las máquinas virtuales en el clúster para confirmar que están trabajando (esto generalmente se conoce como *heartbeat* o latido del corazón). Una máquina virtual (en siglas, VM) es el servidor primario y es el sistema con el que los usuarios interactúan. La segunda VM

sirve como copia de seguridad, lista para actuar en caso de que el servidor primario se caiga.

El servidor primario envía constantemente cualquier cambio al servidor secundario para que su estado refleje el de la VM primaria en todo momento. Si la VM principal falla, el supervisor de virtualización detecta que no está disponible y cambia a los usuarios al servidor de respaldo. Los usuarios que se conecten después del cambio no notarán nada y no serán conscientes de que están conectados a una VM diferente. Por su parte, los usuarios que se han conectado a la VM original, que ya no está disponible, tampoco serán conscientes del cambio, porque el software de virtualización ha ido enviando el estado de estos usuarios a la máquina secundaria. A lo sumo, podrán notar un descenso en la capacidad de respuesta mientras se realizaba el cambio, pero generalmente es tan rápido que es difícil que lo detecten.

Ahora bien, como podemos ver, hay un recurso poco aprovechado en el clúster de virtualización: hay una VM que actúa como copia de seguridad y que se mantiene actualizada, pero que no realiza ningún trabajo. Aunque ejecutar una VM en un servidor virtualizado es ciertamente menos costoso que dedicar un servidor completo para actuar, como una copia de seguridad activa, esto genera costes.

## Duplicación de datos (data mirroring)

Hasta aquí hemos abordado los mecanismos involucrados en mantener a las máquinas virtuales en funcionamiento, pero ¿qué hay de los datos? Después de todo, las aplicaciones dentro de las máquinas virtuales son inútiles sin datos, por lo que es claramente importante garantizar la disponibilidad de los datos como parte de una estrategia general de calidad de servicio.

Una forma de mantener los datos disponibles es a través de la duplicación. Como el nombre implica, esta duplicación o reflejo significa que los datos existentes en un sitio son reflejados en otro, y ambas contienen la misma información. La duplicación permite la consistencia en tiempo real entre dos fuentes de datos. Esto posibilita el

cambio inmediato entre un sistema y otro, es decir, conectando el segundo sistema a la duplicación o el reflejo de los datos del sistema primario.

## Réplica de datos

La replicación es otro servicio orientado a mejorar la calidad del servicio de datos. A diferencia de la duplicación (*data mirroring*) que se enfoca en cómo mantener copias de datos consistentes en tiempo real, **la replicación aborda la necesidad de mantener copias completas de los datos para que puedan ser utilizados en la reconstrucción del sistema**. Esto se logra enviando copias de datos a un almacenamiento centralizado, lo que permite a una organización tener la seguridad de que en caso de que necesite acceder a los datos críticos por algún motivo, estos están almacenados de forma segura y disponibles en caso de ser necesarios.

La eficiencia es vital para la replicación, es decir que, no debemos pensar que porque los datos se están moviendo a una ubicación de almacenamiento hay que olvidarse de que los datos deben fluir correctamente. Un software de replicación inteligente mantiene los cambios, minuto a minuto, fluyendo a la ubicación central, asegurando así que una organización de TI pueda localizar rápidamente los datos y usarlos para reconstruir el sistema en caso de fallos.

## Flexibilidad operacional de TI

Las organizaciones de TI deben estar preparadas para responder ante los cambios y las condiciones de negocio. Es por ello por lo que no hay mejor palabra para definir esta cualidad que el adjetivo «ágil». Cuando pensamos en un atleta ágil, probablemente imaginamos a alguien que puede cambiar de rumbo rápidamente, detenerse o cambiar la dirección para adaptarse a lo que sucede en el entorno que le rodea.

Como es sabido, estos son los requisitos para las organizaciones de TI en la actualidad:

- ▶ Deben ser lo suficientemente flexibles como para aumentar o reducir los recursos informáticos que dedican al desarrollo de sus aplicaciones.
- ▶ Deben implementar infraestructura y procesos que reduzcan la cantidad de trabajo manual para cambiar los recursos informáticos que utilizan.

En resumen, **la organización de TI actual debe estar lista para moverse rápidamente en cualquier dirección a fin de responder a los cambios internos o externos del entorno.**

## Balanceo de carga (*load balancing*)

El equilibrio o balanceo de carga protege a un sistema de la vulnerabilidad contra cualquier condición de error dada al implementar la denominada redundancia. Esta se logra a través de **la ejecución de una o más copias de una máquina virtual en servidores separados**. Cuando se ejecutan dos instancias de una máquina virtual y una de ellas se bloquea, la otra continúa funcionando. Si el hardware que da soporte a una de las máquinas virtuales falla, la otra máquina sigue funcionando. De esta manera, se evita que la aplicación sufra una interrupción.

El balanceo de carga también hace un mejor uso de los recursos de la máquina. Esto es así porque, en lugar de que la segunda máquina virtual esté inactiva y no realice ningún trabajo útil (aunque esté siendo actualizada por la máquina principal), la segunda VM lleva la mitad de la carga y esto hace que al menos la mitad de sus recursos se utilicen. El uso de recursos duplicados puede extenderse más allá de las máquinas virtuales en sí mismas. Las organizaciones que luchan por alcanzar altos niveles de disponibilidad a menudo implementan redes duplicadas con cada servidor físico con conexión cruzada con el resto de la red, lo que garantiza que las máquinas virtuales continuarán siendo capaces de comunicarse, incluso si parte de la red se cae.

La migración hacia un almacenamiento virtualizado puede ayudar con el balance de carga, la combinación de almacenamiento y virtualización en el servidor proporciona:

- ▶ Máxima flexibilidad.
- ▶ Mayor aprovechamiento.
- ▶ Administración más sencilla.

Es necesario el almacenamiento en la red si las máquinas virtuales se ejecutan en servidores diferentes. Las máquinas virtuales con carga balanceada también se pueden configurar para que funcionen como un agrupamiento o clúster y que comparten el estado entre ellas. De esa manera, si una máquina virtual falla, su trabajo puede ser retomado por la otra máquina virtual.

## Agrupación de servidores (*server pooling*)

Con la agrupación de servidores, el software de virtualización gestiona un grupo (*pool*) de servidores virtualizados. **En lugar de instalar una VM en un servidor en particular, simplemente se apunta el software de virtualización a la imagen de la VM** y este software determina qué servidor físico es el más adecuado para ejecutar la VM. El software del pool de servidores también realiza un seguimiento de cada VM y servidor para determinar cómo se asignan los recursos. Si una VM necesita ser reubicada para utilizar mejor los recursos disponibles, el software de virtualización automáticamente migra la VM a un servidor más adecuado.

El *pool* se gestiona a través de una consola de administración y, si se advierte que este está alcanzando su tasa de utilización máxima, se puede agregar de forma transparente otro servidor adicional al grupo. A continuación, el software de virtualización reequilibra las cargas para que el uso de todos los recursos de los servidores sea lo más efectivo posible. Debido a que no se puede saber qué servidor físico ejecutará la máquina virtual, el almacenamiento debe estar conectado en red para que una VM en cualquier servidor pueda acceder a los datos.

Sin lugar a duda, **aquí está la semilla del futuro de la virtualización**. En un futuro próximo, los departamentos de TI dejarán a un lado la instalación manual sistemas operativos en servidores individuales, o incluso en la gestión de clústers de máquinas virtuales en servidores individuales, debido a que son prácticas ineficientes.

## Gestión de recuperación ante desastres

La recuperación ante desastres engloba a productos y procesos que ayudan a las organizaciones de TI para que puedan responder ante situaciones catastróficas, mucho peores que aquellas que se desatan cuando una VM falla o aquellas en las que falla una pieza de hardware. **La recuperación ante desastres entra en juego cuando todo el centro de datos se pierde, ya sea de forma temporal o permanente.**

En estos casos, las organizaciones de TI necesitan luchar para mantener la infraestructura informática de toda la empresa en funcionamiento. Pensemos en el huracán Katrina: cuando se desató la catástrofe muchas empresas de TI perdieron toda la capacidad de procesamiento porque sus centros de datos quedaron completamente inundados. Como si eso fuese poco, también se perdió la conectividad a Internet debido a que los centros de telecomunicaciones también estaban inundados. En estos casos, la capacidad de reserva en el centro de datos no tiene ninguna relevancia. Ante siniestros de esta magnitud como tormentas, terremotos o desastres provocados por el hombre, **se necesita contar con la capacidad de recuperación ante desastres**.

Es un requisito indispensable contar con capacidad adicional en el centro de datos, facilidad para recuperar los sistemas operativos y las aplicaciones, y una ágil administración de la infraestructura migrada. Además, se necesitará contar con un plan de contingencia para el proceso de recuperación de desastres, de modo que, si este ocurriera, el personal de TI pueda ejecutar el plan documentado y ensayado.

La virtualización será útil para la recuperación de la aplicación y las tareas de gestión: la tolerancia a fallos, la alta disponibilidad, el agrupamiento o *clustering* y las capacidades de virtualización de balance de carga (o agrupación de servidores) se pueden aplicar en un escenario de recuperación ante desastres. Todo dependerá de cuánto se desee gestionar físicamente durante el proceso de recuperación ante desastres.

Debido a que las imágenes de la VM pueden capturarse en archivos y luego iniciarse a través del hipervisor, la virtualización es una tecnología ideal para escenarios de recuperación ante desastres. Como se puede imaginar, en un momento crítico, localizar servidores físicos, configurarlos, instalar y configurar aplicaciones, hacer una copia de seguridad y actualizar el sistema puede ser una verdadera pesadilla. Además, mantener una capacidad informática adicional en un centro de datos remoto, que refleje completamente la infraestructura informática primaria, es extremadamente caro.

A través de la virtualización, un conjunto mucho más pequeño de máquinas puede mantenerse disponible en un centro de datos remoto, con un software de virtualización preinstalado y listo para aceptar imágenes de una VM. En el caso de que se produzca un siniestro de enormes proporciones, las imágenes de la VM se pueden transferir desde el centro de datos de producción hacia el centro de datos de *backup*. Estas imágenes de la VM pueden iniciarse con el software de virtualización preinstalado y ponerse en marcha en solo unos minutos.

En caso de dudas, si existe un riesgo inminente de que algunas transacciones se perdieran sin que hubiese tiempo para migrar imágenes de una VM, se puede ejecutar el clúster o el balanceo de carga, a fin de que los dos centros de datos permanezcan actualizados. Esta infraestructura asegura el acceso al más valioso activo de toda organización: los datos, y, además, ayuda a las empresas a ser resilientes en caso de que se pierda el acceso a los datos en una de las ubicaciones.

## Ejemplos concretos

Sector y objetivos	Quién (empresas)	Qué	Ventajas
Educación – VDI	The University of Toledo	Recursos computacionales a estudiantes	Disponibilidad y eficiencia
Educación – Migración	Indiana University	Abandonar AIX hacia plataformas sostenibles	Reducción de costes y mantenimiento
Green IT	Varios	Responsabilidad para el medioambiente	Reducción de costes y mejor imagen
Salud – Homogeneidad	Consorci Sanitari del Garraf	Simplificar infraestructuras	Menor gestión
<i>Hosting</i> – Servicios de terceros	Consona	Cambio de negocio hacia el <i>hosting</i> virtual	Reducción de costes
Consolidación	Avanade	Reducción de <i>legacy</i> y homogeneización	Reducción de costes
Continuidad	Bouygues	Garantizar la continuidad de servicio	Mantenimiento del negocio
Gestión – VDI	Aspentech	Teletrabajo y virtualización de escritorio	Disponibilidad y eficacia
Migración y automatización de pruebas	INA Industria Nafte	Eliminar <i>legacy</i> y automatizar test	Reducción de defectos y eficiencias

Tabla 1. Ejemplos reales de virtualización. Fuente: elaboración propia.

## 5.3. Resultados de alto impacto

Los centros de datos tradicionales, aunque ofrecen buenas capacidades de procesamiento, tienen sus limitaciones, y está claro que no podrán satisfacer las necesidades de TI del mañana. Están formados por sistemas aislados, cada uno de los cuales proporciona funcionalidades específicas, pero que no pueden trabajar de forma cooperativa con otros sistemas. El modelo tradicional «una aplicación, un servidor», tan típico de muchos centros de datos: aplicaciones en servidores físicos incapaces de compartir datos, incapaces de aprovechar eficientemente el hardware y poco ágiles para responder rápidamente a las condiciones cambiantes (y a veces caprichosas) de negocio. **La virtualización proporciona beneficios reales en términos de rendimiento empresarial: mejor utilización del hardware e infraestructuras de TI más robustas y ahorro de costes de energía eléctrica.**

### Infraestructura convergente

La virtualización está cambiando el paradigma de los negocios de TI y es por ello por lo que es necesario examinar todos los supuestos y prácticas preexistentes para alinearlos con la nueva realidad.

Implementar la virtualización a una determinada infraestructura, sin hacer un análisis previo, puede generar fricciones y opacar una visión objetiva sobre todos los beneficios que puede ofrecer la tecnología. Por el contrario, un enfoque más eficiente consiste en repensar cómo está montada la infraestructura y qué cambios se pueden aplicar en pos de la virtualización.

### Estrategias de éxito

#### Análisis de la infraestructura

En lugar de enfocar a la virtualización como un esfuerzo fragmentario, es necesario tener **una visión integral de la infraestructura** y planificar con detenimiento cómo

se podría aplicar la virtualización en todo el centro de datos. Es necesario hacerse estas preguntas: ¿qué servidores se pueden virtualizar?, ¿qué aplicaciones deben permanecer en sistemas independientes?, ¿qué tipo de redes y almacenamiento necesitas para dar soporte a un entorno altamente virtualizado, que permita un rápido movimiento entre las VM y las cargas de trabajo de las aplicaciones?

## **Uso y preferencia de tecnología diseñada y optimizada para la virtualización**

Cuando llega el momento de reemplazar el hardware existente o agregar capacidad, asegúrate de que los nuevos productos sean compatibles con la virtualización. Actualmente, se están diseñando servidores, almacenamiento y sistemas Blade compatibles con la virtualización y es por ello por lo que incluyen memoria adicional, balance de carga automatizado, más conexiones de red, e incluso software de virtualización integrado. Asegúrate de que el nuevo equipo de infraestructura puede soportar los planes de virtualización que tienes en mente.

## **Crea un entorno de almacenamiento en red o compartido**

Si bien el concepto de «una aplicación, un servidor» era suficiente para las organizaciones TI del pasado, en la actualidad **es un modelo que ha quedado desfasado**. Sabemos que el almacenamiento que no se puede compartir representa una gran desventaja, porque mantiene los sistemas aislados e incapaces de responder a las condiciones cambiantes de mercado. Se requiere una red de área de almacenamiento compartido, del inglés *Network Attached Storage*, (en siglas, SAN o NAS) para aprovechar las capacidades de virtualización del servidor como: la migración en tiempo real de las máquinas virtuales, la alta disponibilidad, la tolerancia a fallos y la recuperación ante desastres.

## **Virtualización de las conexiones de red**

Las conexiones que vinculan los servidores a las redes de almacenamiento y a las **redes de área local (en siglas, LAN)** pueden representar una importante barrera.

Cada cambio que requiera intervención manual ralentizará a la organización de TI y la alejará de la agilidad tan característica de DevOps. Las conexiones de red virtualizadas permiten que estos posibles puntos de fricción se gestionen de forma remota, eliminando así los cuellos de botella.

## **Gestiona los recursos virtuales y físicos con las mismas herramientas**

Muchas organizaciones de TI instalan una solución de gestión de la virtualización junto a los recursos de gestión existentes y luego se preguntan por qué la carga de trabajo ha aumentado en vez de disminuir. El enfoque más recomendado es aquel que te permita gestionar los dispositivos físicos y virtuales desde la misma interfaz.

## 5.4. Buenas prácticas en administración de TI

**La clave para hacer coincidir los requisitos de negocio con las operaciones de TI consiste en una alineación adecuada y decisiones acertadas a la hora de elegir los recursos físicos y virtuales necesarios para brindar los servicios que permitirán alcanzar los objetivos comerciales.** Además, dichos recursos deben administrarse en tiempo real para garantizar que las aplicaciones y la infraestructura estén siempre listas para cumplir con las demandas del servicio.



Figura 1. El Gobierno de TI se aplica también a la virtualización. Fuente: elaboración propia.

A continuación, veremos las áreas más importantes en las que debemos focalizarnos para que esta alineación sea eficiente y productiva:

- ▶ **Considera a la virtualización como un servicio de negocio.** Al tener esta perspectiva, estarás motivado bajo la premisa de que la virtualización forma parte de un esfuerzo conjunto de la organización para cumplir con los requisitos de negocio.

- ▶ **Monitoriza los servicios en las infraestructuras físicas y de virtualización.** Los complejos procesos empresariales actuales están controlados por múltiples aplicaciones, que pueden ubicarse en infraestructuras físicas o virtuales. La administración desde una perspectiva de negocio te permitirá realizar un seguimiento de las operaciones de TI en todo el espectro de recursos de TI y optimizarlos de manera integral. El uso de un enfoque unificado hacia la administración física y virtual evitara que haya grietas (y, por tanto, fallos y pérdidas de información) entre estos dos bloques.
- ▶ **Virtualiza los procesos de gestión de los servicios.** Las soluciones de gestión proporcionan procesos como la gestión de incidentes y fallos. También aportan soluciones consistentes para gestionar el cumplimiento de las licencias en entornos virtuales y físicos. Además, favorecen la alineación de todos los recursos y procesos para las iniciativas de optimización de servicios.
- ▶ **Integra la virtualización en la gestión de calidad.** Las áreas que se benefician en mayor medida de la virtualización son aquellas relacionadas con las pruebas y la calidad, ya que se pueden eliminar las intervenciones manuales. Como beneficio aparejado, veremos un aumento de la coherencia, porque se podrán utilizar recursos idénticos en todo el ciclo de vida de QA y se evitarán errores derivados de una configuración incorrecta o una instalación inconsistente.

## Virtualización del cliente desde el escritorio hasta el centro de datos

La **virtualización del cliente** es la última frontera en la tecnología de virtualización y es sabido que brinda enormes beneficios a las organizaciones. Aunque la virtualización del servidor mejora las operaciones del centro de datos, la virtualización del cliente puede mejorar los resultados para cualquier otra parte de la organización, en otras palabras, para la gran mayoría de empleados.

Existen numerosos enfoques para implementar la virtualización del cliente y cada uno de ellos presenta beneficios. Sin embargo, es importante que las circunstancias sean las adecuadas para asegurar que se obtenga el máximo valor de su implementación.

Estas son algunas de las consideraciones más importantes que hay que analizar antes de proceder hacia la virtualización del cliente:

### **No pierdas de vista los requisitos de negocio**

El ordenador tradicional proporciona un buen nivel de productividad a los trabajadores, pero tal vez no sea suficiente para cumplir con todos los objetivos de negocio. Por ejemplo, las necesidades de recuperación ante desastres pueden establecer que las aplicaciones y los datos se mantengan dentro de los centros datos donde se puedan ser migrados rápidamente, a través de la virtualización, hacia otros centros de datos.

### **Asegúrate de que la tecnología que elijas es adecuada para cada tipo de usuario**

Dependiendo del tipo de empleado del que se trate, puedes elegir una variedad de virtualización del cliente u otra. Las soluciones informáticas basadas en el servidor proporcionan acceso remoto a la aplicación, a un escritorio en un entorno operativo compartido. Ciertamente, desde un punto de vista financiero, esto puede representar cierta rentabilidad, ya que aumenta la seguridad del cliente y mejora la protección de datos y la capacidad de administración para aquellos trabajadores que desempeñan sus funciones resolviendo tiques o tareas concretas.

Por su parte, la infraestructura de escritorio virtual, del inglés *Virtual Desktop Infrastructure*, (en siglas, VDI) proporciona escritorios completamente funcionales y personalizados, que se entregan a través de la red desde un servidor compartido. Cada escritorio virtual está aislado y seguro en el centro de datos, compartiendo

recursos del centro de datos físico, para garantizar un uso óptimo de sus recursos a través de la virtualización. Esta opción resulta ideal para las aplicaciones básicas de oficina y permite que TI maximice la utilización, reduzca costes y aumente la fiabilidad.

Los Blade resultan óptimos para las aplicaciones especializadas que operan, por ejemplo, con gráficos pesados, ya que ofrecen un rendimiento muy fluido y sin interrupciones (generando una excelente experiencia de usuario), junto con un control centralizado y seguridad en el centro de datos.

### **Determina cuáles son tus necesidades de almacenamiento**

La gran mayoría de las soluciones de virtualización del cliente requieren que se muevan datos fuera del dispositivo local hacia otra solución de almacenamiento centralizada. Asegúrate de que en tu planificación de virtualización de cliente incluyes requisitos y opciones de almacenamiento que puedan dar soporte de forma eficiente.

### **No olvides ser coherente en tus esfuerzos de virtualización de cliente**

Cuando implementas varias tecnologías diferentes de virtualización del cliente, hay que asegurar que todas ellas funcionan bien juntas. Por ello, obtener soluciones de diferentes proveedores puede complicar aún más las cosas y perjudicar el esfuerzo global de virtualización, y, ciertamente, obstaculizará la premisa de mantenerse alineado con los requisitos de negocio. Dicho esto, es probable que la opción más coherente sea la de contar con un único proveedor que proporcione una cartera completa de soluciones y servicios de cliente.

## 5.5. Cómo encauzar tu proyecto de virtualización

A continuación, descubriremos cuáles son las premisas fundamentales que debes tener en mente si deseas emprender el viaje hacia la virtualización y quieres trabajar de forma metódica y ordenada.

Para comenzar, lo más importante que debes saber es que **la virtualización es un camino para emprender, no un producto**. Cualquier proyecto importante requiere una planificación cuidadosa y, por supuesto, una monitorización en cada una de sus etapas. Deberás pasar por la planificación, la implementación y tareas ligadas al área de operaciones, y luego dar seguimiento al proyecto y evaluar los hitos. La virtualización hace que el proceso de planificación de proyectos sea aún más importante porque, después de que las organizaciones de TI comienzan a ver los beneficios tangibles de la migración a entornos virtualizados, comienzan a cuestionarse en qué otras áreas podrían aplicar la virtualización.

A continuación, **es recomendable que realices una evaluación de los casos de uso**. En pocas palabras, cómo aplicarás, o cómo aplicarías en un futuro próximo la virtualización. Un caso de uso es sumamente útil para identificar las capacidades de virtualización que quizás no necesites hoy en día, pero que podrían ser relevantes en el futuro. Debido a que hay varias maneras diferentes de usar la tecnología de virtualización, debes pensar en cómo quieres o puedes aplicarla. La gama de productos que se pueden utilizar para la virtualización tiende a reducirse a medida que aumenta la complejidad de las aplicaciones. En estos casos, acudir a un taller o workshop organizado por un proveedor puede ayudarte a esclarecer qué producto es más conveniente para tu caso particular. Otra herramienta valiosa es realizar una prueba de concepto, del inglés *proof of concept*, (en siglas, POC) de virtualización.

El siguiente paso en este plan metódico es **analizar la estructura del equipo de operaciones de la organización**. Es tentador creer que la virtualización es un

asunto puramente técnico, pero eso sería muy simplista. Lo que sí es un hecho contrastado es que los humanos somos animales políticos (o, al menos, sociales). Eso significa que las decisiones, incluso las tecnológicas, confrontan con los prejuicios emocionales de las personas que forman parte de la organización y afectan la aceptación de nuevas iniciativas (resistencia al cambio).

Muchos equipos de operaciones de TI están organizados de forma funcional. En otras palabras, un grupo administra servidores, otro grupo administra la red y otro administra los recursos de almacenamiento. Sin embargo, la virtualización integra todas estas funciones en una unidad, por lo que estos grupos autónomos necesitarán colaborar y cooperar. Por lo tanto, **es necesario examinar el «lado humano» de los equipos de operaciones de TI para garantizar que la virtualización se llevará a cabo de manera fluida.**

Para continuar avanzando en el viaje hacia la virtualización, es necesario que definas tu arquitectura. Esto es una forma elegante de decir que debes **recoger la información que has recabado sobre los casos de uso, así como de la estructura organizativa de operaciones y definir cuál será tu diseño de infraestructura de virtualización.** Este diseño es absolutamente crítico y es necesario que sea revisado por todas las partes interesadas e involucradas. Este proceso de revisión tiene dos propósitos fundamentales: por un lado, garantiza que se han recogido las necesidades de todas las partes y, por otro lado, tiende a generar conciencia y compromiso en los diferentes grupos involucrados en la virtualización.

Una vez que hayas definido tu arquitectura de virtualización, ya puedes **seleccionar el producto o los productos (software) que usarás en tu proyecto de virtualización.** Esta selección debería ser un ejercicio relativamente sencillo, ya que previamente has identificado qué funcionalidad debe estar presente en tu arquitectura de virtualización. En cuanto al servidor, los productos de VMware, Citrix y Microsoft son opciones más que consolidadas. A su vez, numerosos productos de

virtualización de almacenamiento están en el mercado, y esta se está extendiendo a nuevas áreas como las redes. Es una cartera mucho más rica, pero con una gran cantidad soluciones, solo debes seleccionar cuidadosamente aquellos productos que cumplan con los requisitos de los casos de uso que has detectado.

Y como es de esperar, lo siguiente que debes tener en cuenta para tu proyecto es seleccionar el hardware de virtualización. **Un punto clave es que te asegures de que el hardware es compatible con la virtualización.** Muchas organizaciones intentan reutilizar hardware del que ya disponen y descubren que no es escalable ni es adecuado para los casos de uso que tienen en mente.

Además, cada vez que virtualizas servidores, también debes evaluar:

- ▶ La infraestructura de almacenamiento para que satisfaga las demandas del servidor.
- ▶ Las implementaciones de virtualización del cliente y que, a su vez, estén estrechamente integradas con la plataforma del software de virtualización.

Llegados a este punto, **es necesario que realices una prueba piloto para confirmar que todo funciona, como es debido, en producción.** Realiza una prueba de tu solución de virtualización en un entorno de prueba controlado para asegurarte de que todo funciona correctamente. Ya tienes tu software de virtualización, el hardware que te acompañará en este viaje hacia la virtualización y cualquier otro dispositivo necesario para que el centro de datos funcione adecuadamente. Este es el momento de confirmar que tienes todo lo que necesitas para seguir adelante con la migración hacia tu nueva arquitectura. Este paso tiene que ejecutarse con cautela y sin errores, porque si la infraestructura no está bien montada, los sistemas de producción no trabajarán correctamente. Así que, pon a punto las configuraciones y asegúrate de que se inicia correctamente.

## Poner gobierno y política en el lugar correcto

La virtualización reduce, en gran medida, el esfuerzo para implementar nuevos sistemas. Eso puede causar problemas, ya que, los nuevos sistemas se crean con poca supervisión. Puedes evitar esta expansión descontrolada poniendo en práctica procesos y políticas. Esto se conoce como el término «gobierno» que, como es de esperar, implica estructura y reglas formales por las que se rigen el uso de los recursos de TI.

## Gestiona tu infraestructura virtualizada

Asumamos que finalmente has implementado tu infraestructura de virtualización y conseguiste que sus servidores físicos sean migrados a ella. Quizás crees que has terminado, pero de ninguna manera. Necesitas administrar el nuevo entorno. Además, para obtener el máximo beneficio de tus nuevas capacidades de virtualización, necesitas integrarlos en tu proceso de gestión general, para que puedas administrar sistemas físicos y virtuales con las mismas herramientas, los mismos procesos y las mismas personas. Esto es particularmente importante porque las organizaciones están cambiando cómo implementan los sistemas hoy en día: puedes mantenerlos virtuales a lo largo de las fases de desarrollo y despliegue o puedes desarrollarlos virtualmente y desplegar físicamente. Por esta razón, se utiliza un proceso de gestión y un conjunto de productos que pueden fácilmente gestionar tanto lo físico como lo virtual es importante.

## Diez errores de virtualización que debes evitar

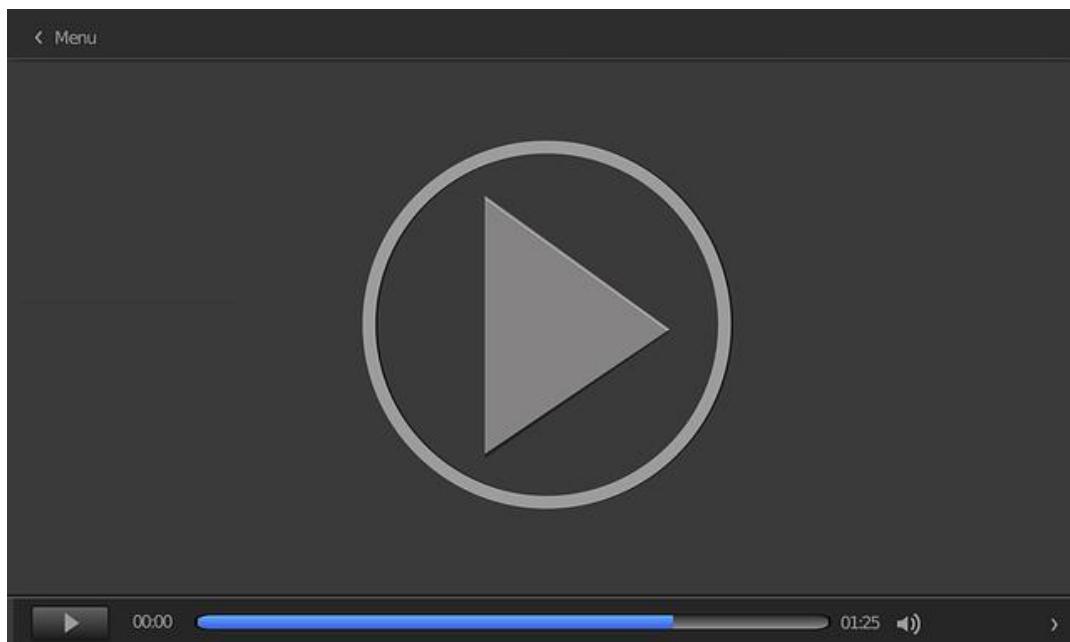
- ▶ **No esperes la perfección.** El campo de virtualización está en un gran flujo porque así están sucediendo muchos cambios, a veces parece un hay nuevos productos o servicios disponibles todos los días. Algunas personas deciden abrazar un campo que cambia rápidamente porque esperan que ofrezca tremendo potencial. Otros se quedan atrás, con el pensamiento de que preferirían diferir la acción hasta que las cosas estén más estable.

- ▶ **No escatimes en el entrenamiento.** Una de las cosas más desconcertantes sobre las organizaciones de TI es que invertirán grandes sumas en nuevo hardware y software, pero escatiman en asegurar que sus empleados sepan cómo usar sus nuevos sistemas.
- ▶ **No te expongas a problemas legales:** *compliance*. Otra área potencial para considerar en un entorno virtualizado es el licenciamiento de software. Cuando sistemas físicos individuales se aprovisionan y presentan como sistemas múltiples, y el uso de sistemas virtuales es variable, los acuerdos de licencia históricos y los escenarios pueden cambiar. Además, con el auge de los dispositivos virtuales y la gran cantidad de nuevos sistemas posibles (debido a la virtualización), hace muy complicado, y más importante, el seguimiento de las licencias de software. Así que es necesario asegurarse de poner los procesos para garantizar que realiza un seguimiento de licencias y mantener el cumplimiento de todas sus responsabilidades.
- ▶ **No pienses que la virtualización es estática.** Las condiciones de su negocio no solo dictarán que evalúes continuamente qué tan bien tu infraestructura de virtualización cumple con las realidades comerciales actuales, pero la virtualización en sí está cambiando constantemente. Esto significa que tu solución de virtualización de vanguardia, implementada hace dieciocho meses, puede necesitar ser renovada a la luz de los nuevos desarrollos de virtualización.
- ▶ **No evites hacer el trabajo que consideras aburrido.** Es divertido instalar software y ver surgir cosas nuevas. No es tan divertido utilizar entrevistas de casos o revisiones técnicas. Pero ten en cuenta que esas tareas «aburridas» hacen posibles a las cosas divertidas. De hecho, a menos que completes estas tareas aburridas, probablemente no obtendrás el visto bueno para avanzar con el proyecto y divertirte haciendo las cosas interesantes.

- ▶ **Presta atención a los casos de negocio.** En estos tiempos de presupuestos ajustados para las organizaciones de TI, hoy no hay forma más segura de que tu proyecto sea derribado que la falta de casos negociales para él. Por otro lado, no hay manera 100 % segura de asegurar que tu proyecto obtenga apoyo ejecutivo y navegue a través del proceso de aprobación, más que demostrando impresionantes beneficios financieros generados al avanzar con el proyecto. Esto significa que te corresponde: evaluar la situación financiera, el impacto de pasar a la virtualización y presentar esa información como parte del proceso de aprobación del proyecto.
- ▶ **Presta atención a la organización.** Debido a que la virtualización afecta a tantos grupos, es importante que trabajes con cada uno de ellos y los convenzas de que la virtualización hará que su trabajo sea mejor y más fácil. Esto incluye no solo grupos técnicos, sino patrocinadores comerciales, así como la alta gerencia. Cuando cambias la infraestructura afecta a muchos grupos, así que asegúrate de incluirlos a todos en la planificación de tu proyecto. La clave es hacer que sus flujos y modelos de trabajo organizacionales explícitos.
- ▶ **Presta atención al hardware.** La virtualización es un software que habilita a otros recursos de software para aprovechar mejor el hardware subyacente. Pero no imagines que el hardware en sí no tiene ningún efecto en la virtualización. Lejos de eso, el tipo y la capacidad del hardware que utilizas para alojar su solución de virtualización puede afectar drásticamente la densidad de virtualización que logres, así como el rendimiento y los niveles de disponibilidad para tus máquinas virtuales o clientes. Así que, idealmente, tu hardware debe estar alineado a tus requisitos y expectativas.

- ▶ **Presta atención a la administración de servicios.** La mayoría de las plataformas de virtualización vienen con su propio conjunto de herramientas de administración. Aunque muchas de estas herramientas funcionan bien dentro de sus dominios propios, no son lo suficientemente maduros o robustos para administrar más allá de sus plataformas originales. También tienden a requerir capacitación adicional para ejecutar, mantener y producir gestión de información que reside fuera de los procedimientos de operaciones estándar aceptados. En lugar de adoptar un conjunto de herramientas y procedimientos para la virtualización, y otro para entornos físicos, buscar soluciones de gestión que administren tanto entornos físicos como virtuales en común facilitara el éxito.
- ▶ **Prepárate para un futuro aún más virtualizado.** La virtualización facilita la creación de nuevos sistemas, reduciendo el esfuerzo de días o semanas a meros minutos. Asimismo, el coste puede reducirse en un orden de magnitud. Dada esta facilidad de creación de sistemas, algunas organizaciones descubren que de repente tienen un exceso de sistemas: están expandiendo todo y, aunque el coste y el esfuerzo de crear un sistema virtual es bajo, si se hace en exceso, el problema de multiplicidad, volumen y falta de control será mucho mayor que con los sistemas físicos.

Existe una concepción errónea de que la nube es una solución económica y preferible a otras soluciones como la virtualización o la tecnología *serverless*. En la píldora *¿Cuál es el precio de la nube?* comentaremos los pros y contras de algunas de estas estrategias.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=bb948542-7a89-4489-a229-ad78013d2c84>

---

## Caso éxito virtualización - MAPFRE

Agilitix - Beyond Productivity. (2016, mayo 23). *Caso Éxito Virtualización - MAPFRE [Vídeo]*. YouTube. <https://www.youtube.com/watch?v=n-JeZ23NkM0>

En este vídeo podrás ver cómo funcionó la implementación de la virtualización dentro de MAPFRE para dar paso hacia mejores prácticas de implementación y uso de esta tecnología integrada.

1. Señala la opción que no se encuentra entre los casos de uso apropiados para virtualización:

  - A. Consolidación de servidores.
  - B. Entornos de desarrollo y pruebas.
  - C. Computación de nube privada.
  - D. Incrementar la seguridad de red.
  
2. Señala la opción que ayudará a encauzar la puesta en marcha de un proyecto de virtualización:

  - A. Establecer el gobierno y las políticas necesarias.
  - B. Consolidación de servidores siempre.
  - C. Contratar gente nueva.
  - D. Ninguna de las anteriores.
  
3. Señala la opción que no se encuentra entre los casos de uso apropiados para virtualización:

  - A. Calidad del servicio.
  - B. Alta disponibilidad.
  - C. Aumentar la capacidad de almacenamiento sin inversión adicional.
  - D. Recuperación de desastres.
  
4. Señala la opción que se corresponde con un error común en proyectos de virtualización

  - A. Pensar que el hardware no importa y cualquier hardware es bueno para los proyectos de virtualización.
  - B. Organizar sesiones de entrenamiento para todo el equipo.
  - C. Contratar Expertos.
  - D. Ninguna de las anteriores.

5. Señala la opción que mejor encaja con una estrategia de éxito para virtualización
  - A. Aumentar la calidad del servicio.
  - B. Una nueva estrategia requiere gente nueva.
  - C. Establecer un sistema de almacenamiento virtual compartido.
  - D. Recuperación de desastres.
6. Señala la opción que se corresponde con un error común en proyectos de virtualización:
  - A. Organizar sesiones de entrenamiento que no incluyen a toda la empresa.
  - B. Incurrir en problemas legales y/o de *compliance*.
  - C. A y B son ciertas.
  - D. Ninguna de las anteriores.
7. Señala la opción que no se encuentra entre los casos de uso apropiados para virtualización:
  - A. Clusterización.
  - B. Entornos de desarrollo y pruebas.
  - C. Recuperación de desastres.
  - D. Incrementar capacidad computacional de la empresa.
8. Señala la opción que mejor encaja con una estrategia de éxito para virtualización:
  - A. Reducir el equipo de gestión.
  - B. Invertir en hardware, pero no en formación.
  - C. Realizar un análisis previo a iniciar el proyecto de virtualización.
  - D. Todas las anteriores.

9. Señala la opción que mejor encaja con «buena práctica de administración de TI»:
  - A. Realizar *backups* de los sistemas más críticos.
  - B. Invertir en software de virtualización.
  - C. Monitorizar las infraestructuras tanto virtuales como físicas.
  - D. Todas las anteriores.
  
10. Señala la opción que se corresponde con el significado de POC:
  - A. POC significa *proof of concept* o prueba de concepto.
  - B. Push to Talk over Celular.
  - C. Power on connectivity y permite acceder a funciones de red sin tener el hardware encendido.
  - D. Ninguna de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 6. Cloud Computing

# Índice

[Esquema](#)

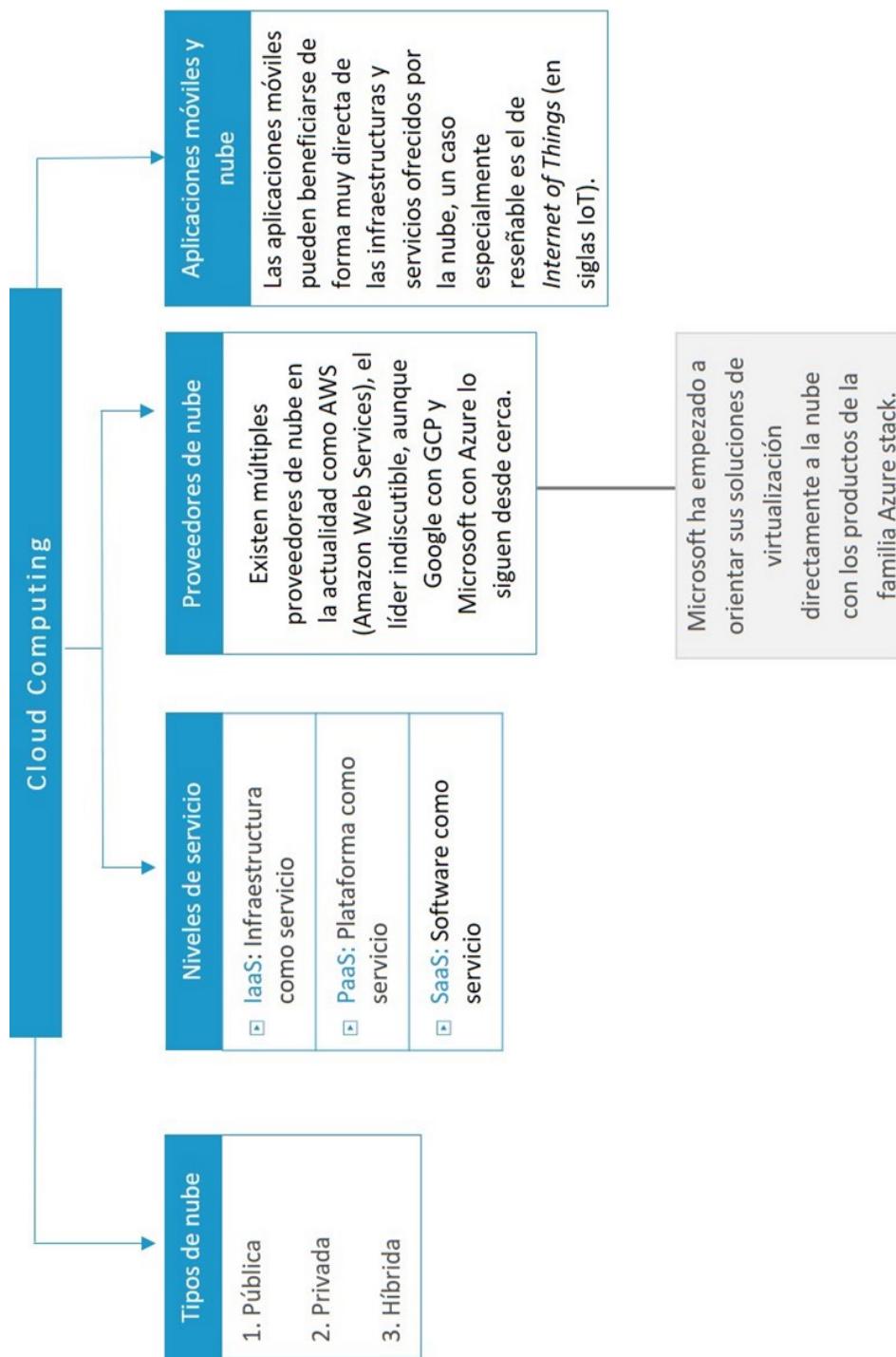
[Ideas clave](#)

- [6.1. Introducción y objetivos](#)
- [6.2. Tipos de nube](#)
- [6.3. Niveles de servicio](#)
- [6.4. Proveedores de nube pública](#)
- [6.5. Aplicaciones móviles y nube](#)
- [6.6. Referencias bibliográficas](#)

[A fondo](#)

[Introducción a la gestión de máquinas virtuales](#)

[Test](#)



## 6.1. Introducción y objetivos

DevOps se originó en las, así llamadas, empresas «nacidas en la web» (empresas que se originaron en la Internet) como Etsy, Flickr y Netflix. Mientras estas empresas daban solución a los desafíos tecnológicos complejos de gran escala, tenían arquitecturas bastante simples (a diferencia de las grandes empresas que crecieron en torno a los sistemas heredados y/o a través de adquisiciones y fusiones), con sistemas de múltiples y complejas tecnologías que tenían que interactuar. Como puedes imaginar, en la actualidad todas las empresas de tecnología se enfrentan a nuevos retos y desafíos, como los modelos de entrega de aplicaciones móviles y de nube y las cadenas de suministro de software. En los siguientes apartados explicaremos algunos de estos retos y, sobre todo, veremos cómo puede ayudar DevOps a resolverlos con la ayuda del Cloud Computing.

Uno de los cambios más importantes que ha afectado a DevOps (y, de hecho, también lo ha ayudado a madurar) es el Cloud Computing (computación en la nube). La omnipresencia de la tecnología de nube privada (pública e híbrida) ha permitido a las organizaciones proveer entornos bajo demanda. En las organizaciones tradicionales, donde utilizan servidores físicos, solicitar un nuevo servidor es un proceso complejo, largo y tedioso. Puede tomar días o incluso semanas adquirir un nuevo servidor físico, instalar y configurar el sistema operativo (en siglas, OS) y configurar todo el software necesario para desplegar la aplicación. La virtualización y la nube ayudan a resolver ese problema en gran medida, y este hecho ha traído aparejado un enorme crecimiento en la escala en la que se utilizan los entornos para desplegar las aplicaciones. A raíz de este fenómeno, se ha hecho necesaria la automatización de las tareas de implementación de software, abordadas y resueltas por DevOps.

Al mismo tiempo, la llegada de la nube facilita el despliegue continuo y proporciona un acceso más fácil a los entornos símiles a producción durante las etapas de desarrollo y pruebas del canal de entrega. También ha llevado a un novedoso modelo de autoservicio para los desarrolladores, quienes normalmente trabajaban de forma separada en relación con operaciones. Con las tecnologías de nube, un desarrollador puede no solo realizar una construcción automática, sino también pedir el aprovisionamiento y configuración de un entorno de producción similar a la nube para desplegar la aplicación que está construyendo... ¡todo ello sin la intervención directa del equipo de operaciones! Ambas capacidades proporcionadas por la nube facilitan la adopción DevOps.

**El paradigma de la nube introduce un cambio en la visualización del sistema y los datos que son propiedad de una empresa.** Además, el uso compartido de servicios o recursos tales como el almacenamiento, hardware y aplicaciones de Cloud Computing, ha facilitado, de una manera totalmente diferente, la coherencia de los recursos y las economías de escala a través de su modelo de negocio de pago por uso. Ya no se trata de un conjunto de dispositivos en una ubicación física que ejecutan un programa de software específico con todos los datos y los recursos presentes en un lugar físico, sino que es un sistema que se distribuye geográficamente, involucrando tanto a la aplicación como a los datos.

El desarrollo de arquitecturas y servicios distribuidos en la nube está lidiando con los mismos problemas de: escalabilidad, elasticidad respecto a la demanda, acceso a la red amplia, medición de utilización, aspectos de seguridad (tales como la autorización y autenticación) y muchos otros conceptos relacionados con los servicios multiusuario, con el fin de servir a un gran número de usuarios simultáneos en Internet. La solución puede ser una nube pública, privada o híbrida, dependiendo del tipo de industria u organización del que se trate.

Los objetivos de este tema son:

- ▶ Conocer los tipos de nube.
- ▶ Analizar los diferentes niveles de servicio.
- ▶ Comprender las ventajas de la utilización de las tecnologías de nube.

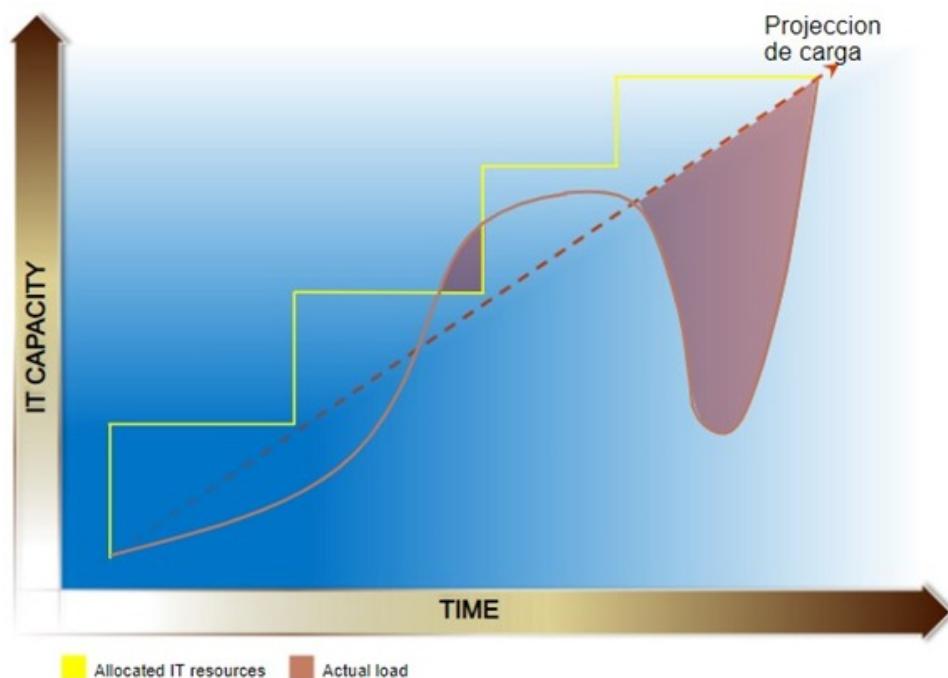


Figura 1. Consumo frente a recursos con TI tradicional. Fuente: SlidePlayer, s.f.

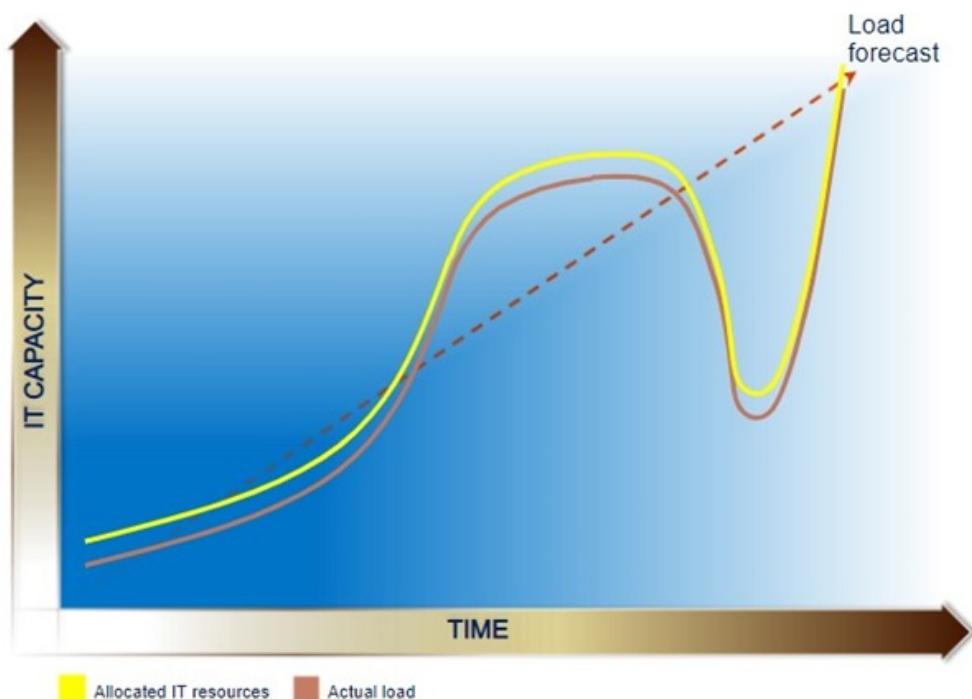


Figura 2. Consumo frente a recursos con utilización de nube. Fuente: SlidePlayer, s.f.

## 6.2. Tipos de nube

El hecho de que la información resida, de forma temporal o definitiva, en servidores de nube da como resultado que dichos servicios ofrezcan distintos formatos de privacidad que cada usuario puede elegir, según sus necesidades. De ahí que se planteen varios modelos de nubes como espacios de desarrollo de los servicios ofertados. Estos son:

- ▶ Nube pública.
- ▶ Nube privada.
- ▶ Nube híbrida.

### Nubes públicas

**Los usuarios acceden a los servicios de manera compartida sin que exista un exhaustivo control sobre la ubicación de la información**, que reside en los servidores del proveedor. Es importante resaltar que el hecho de sean públicas no es un sinónimo de sean inseguras, pero la realidad es que suelen ser más vulnerables a los ataques.

Cuando hablamos de nube pública, queremos decir que toda la infraestructura de computación se encuentra en las instalaciones de una empresa de Cloud Computing que ofrece el servicio en la nube. La ubicación permanece, por lo tanto, separada del cliente y este no tiene control físico sobre la infraestructura. Por último, como las nubes públicas utilizan recursos compartidos, se destacan principalmente por su buen rendimiento.

### Nubes privadas

**Nube privada significa usar una infraestructura en la nube (red) por cada cliente u organización.** Si bien no se comparte con otros, se encuentra

remotamente localizada. Las empresas tienen la opción de elegir una nube privada en la propia sede, que es más cara, pero tiene la ventaja de que así se puede tener control físico sobre la infraestructura. Resulta evidente que el nivel de seguridad y control es más alto cuando se utiliza una red privada que una red pública. Sin embargo, la reducción de costes puede ser mínima si la empresa necesita invertir en una infraestructura en la nube *on premise*.

## Nubes híbridas

**Combinan características de las dos anteriores**, de manera que parte del servicio se puede ofrecer de manera privada (por ejemplo, la infraestructura) y otra parte de manera compartida (por ejemplo, las herramientas de desarrollo).



Figura 3. Ventajas del uso de la nube. Fuente: SlidePlayer, s.f.

## 6.3. Niveles de servicio

Cuando hablamos de servicios en la nube, es importante establecer que existen diferentes opciones, dependiendo del tipo de servicios que se ofrecen y la implicación que tiene el cliente en la gestión de esta. La clasificación más aceptada consiste en dividir los niveles de servicio en tres:

- ▶ Infraestructura como servicio (en siglas, IaaS).
- ▶ Plataforma como servicio (en siglas, PaaS).
- ▶ Software como servicio (en siglas, SaaS).



Figura 4. Niveles de Servicio en la nube. Fuente: SlidePlayer, s.f.

### Infraestructura como servicio (IaaS)

La idea básica es la de hacer **uso externo de servidores para espacio en disco**, base de datos, *roters*, *switches*, así como tiempo de cómputo para evitar tener un servidor local y toda la infraestructura necesaria para la conectividad y mantenimiento dentro de una organizaron. Con una IaaS, lo que se tiene es una solución en la que se paga solamente por el consumo de los recursos usados: espacio en disco utilizado, tiempo de CPU, espacio para base de datos, transferencia

de datos, etc.

## Plataforma como servicio (SaaS)

Se trata de un modelo en el que se proporciona un **servicio de plataforma con todo lo necesario para dar soporte al ciclo de diseño, desarrollo y puesta en marcha de aplicaciones y servicios web a través de esta**. El proveedor es el encargado de escalar los recursos, en caso de que la aplicación lo requiera, para que la plataforma tenga un rendimiento óptimo, de la seguridad de acceso, etc. Para desarrollar software se necesitan: bases de datos, herramientas de desarrollo y, en ocasiones, servidores y redes. Con PaaS, el cliente únicamente se enfoca en desarrollar, depurar y probar que las herramientas necesarias para el desarrollo de software son ofrecidas a través de Internet, lo que teóricamente permite aumentar la productividad de los equipos de desarrollo, gracias a que abstrae del hardware físico al cliente.

## Software como servicio (SaaS)

Consiste en la **entrega de aplicaciones completas como un servicio** que permite la abstracción completa, no solo del hardware subyacente, sino incluso de la plataforma y habilita al usuario (cliente) dedicarse únicamente a su uso (consumo).

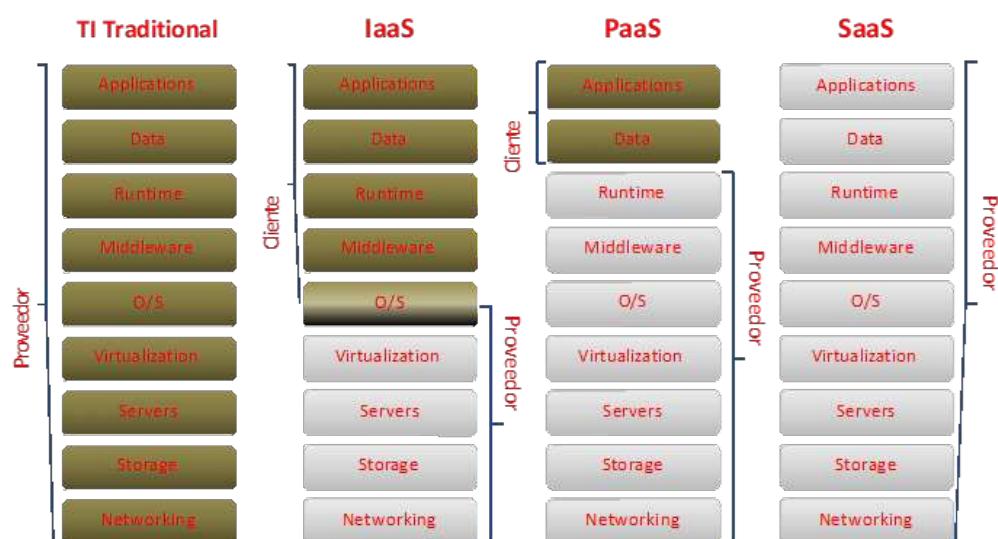


Figura 5. Comparativa de los niveles de servicio en la nube. Fuente: adaptado de SlideShare, 2012.



## 6.4. Proveedores de nube pública

En la actualidad, existen múltiples proveedores de nube pública que intentan repartirse el mercado de los servicios en la nube, sin embargo, existe un claro líder tanto de mercado como por número de servicios ofrecidos, regiones donde se pueden contratar y capacidad total. Ese líder es Amazon Web Services, también conocido como AWS, a quien exploraremos más en detalle luego.

Existen otros proveedores, también con amplias capacidades, que siguen de cerca al líder, en este caso hablamos de Google con su nube Google Cloud y Microsoft con Azure.



Figura 6. Cuadrante Mágico. Fuente: MetaCompliance, 2019.

## 6.5. Aplicaciones móviles y nube

Por lo general, en las empresas, las aplicaciones móviles no están aisladas, diríamos que sirven más bien como *front-end* a múltiples aplicaciones empresariales, ya en uso por la empresa. Estas aplicaciones *back-end* de empresa pueden variar desde sistemas de procesamiento de transacciones hasta portales de empleados o sistemas de adquisición de clientes. **El desarrollo móvil y la entrega son complejos y requieren de un conjunto de servicios interdependientes, a fin de que puedan ser entregados de forma coordinada, fiable y eficiente.**

Para el caso de las aplicaciones móviles empresariales, sus ciclos de entrega y el lanzamiento de nuevas funciones deberán estar coordinados con las aplicaciones y servicios de la empresa con los que estas aplicaciones móviles interactúan. Por lo tanto, la adopción de DevOps debe incluir a los equipos de aplicaciones móviles prioritariamente, que a su vez sean colaborativos con resto de los equipos de desarrollo de software empresarial.

### DevOps y tiendas de aplicaciones

Un aspecto característico de las aplicaciones móviles es la necesidad de despliegue en las tiendas de aplicaciones. **La mayoría de las aplicaciones tiene que «comercializarse» través de un vendedor que gestione la tienda de aplicaciones a fin de llegar a los usuarios.** Apple presentó este formato de distribución con su App Store y bloqueó sus dispositivos contra la instalación directa de aplicaciones, tanto de los desarrolladores de aplicaciones como de los proveedores. Los fabricantes de dispositivos tales como Research In Motion y Microsoft, que permitían antaño la instalación directa de aplicaciones, ahora siguen el modelo de Apple. Esta situación da como resultado que los desarrolladores ya no puedan realizar cambios bajo demanda a una aplicación. Incluso para las correcciones de errores críticos, las nuevas versiones de la aplicación tienen que

pasar por procesos de revisión y examen de la tienda de aplicaciones. La entrega continua se convierte en «enviar y esperar». El despliegue continuo de desarrollo y pruebas sigue estando disponible, siendo el entorno de prueba simulado el de los dispositivos en los que se implementará la aplicación o bancos de dispositivos físicos.

## La Internet de las cosas (*Internet of Things*)

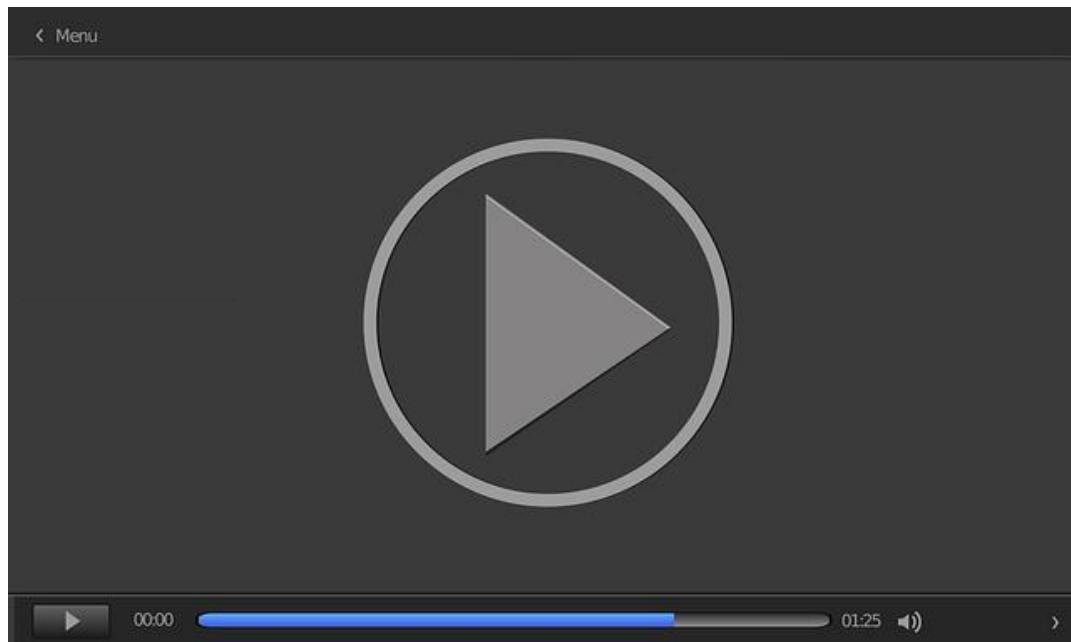
El siguiente gran paso para DevOps es su evolución en los sistemas de dispositivos embebidos. Cuando comenzó la era de la Internet, la mayoría de los datos compartidos eran generados por humanos. Hoy en día, innumerables dispositivos conectados a Internet (tales como sensores) generan muchos más datos que los seres humanos. **Esta red de dispositivos interconectados a través de Internet se conoce comúnmente con el nombre de Internet de las cosas.**

La **Internet de las cosas (IoT, por sus siglas en inglés)** es un término acuñado por Kevin Ashton, un pionero de la tecnología de origen británico, especialista en la **identificación por radiofrecuencia (en siglas, RFID)**, quien concibió un sistema de sensores omnipresentes conectando el mundo físico a Internet. Si bien «las cosas», Internet y la conectividad son los tres componentes básicos de IoT, el valor está en cerrar la brecha entre el mundo físico y digital mediante el autorrefuerzo y la automejora de los sistemas.

En este escenario, DevOps es potencialmente aún más esencial, debido a la codependencia del hardware y el software embebido que se ejecuta en él. Los principios de DevOps aseguran que el software incorporado y entregado a los dispositivos es un software de alta calidad, con las especificaciones técnicas adecuadas. Operaciones, en este caso, se sustituye por hardware o ingenieros de sistemas que diseñan y construyen hardware a medida para los dispositivos. La colaboración entre los equipos de desarrollo y pruebas y los ingenieros de sistemas es crucial para asegurar que el hardware y el software se desarrollan y entregan de

manera coordinada, a pesar de que tengan diferentes ciclos de entrega.

Un MVP es un Producto Mínimo Viable (*Minimum Viable Product*). La creación de un MVP es un primer paso muy recomendable para la creación de una nueva aplicación. Podrás ver en detalle estas cuestiones en el vídeo *Producto mínimo viable*.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=0a635615-c932-4a8e-aa71-ad78013d2c12>

---

## 6.6. Referencias bibliográficas

MetaCompliance. (2019). *2019 Gartner Magic Quadrant for Security Awareness Computer Based Training*. <https://go.metacompliance.com/gartnermagicquadrant>

## Introducción a la gestión de máquinas virtuales

Google Cloud Tech. (2019, abril 12). *Introduction to Virtual Machines (Cloud Next '19)* [Vídeo]. YouTube. [https://www.youtube.com/watch?v=3aNDcgoJ-\\_8](https://www.youtube.com/watch?v=3aNDcgoJ-_8)

Este vídeo te permitirá profundizar en la utilización de Google Compute Platform para la creación y gestión de máquinas virtuales a escala en GCP.

1. ¿A qué tipo de nube nos referimos si la aplicación y los datos tienen que ser gestionados por el usuario, pero lo demás está gestionado por el proveedor de nube?

  - A. PaaS.
  - B. SaaS.
  - C. IaaS.
  - D. Híbrida.
  
2. ¿A qué tipo de nube nos referimos si la infraestructura es propiedad de un usuario, normalmente no está accesible desde internet y ofrece más control?

  - A. Privada.
  - B. PaaS.
  - C. IaaS.
  - D. Híbrida.
  
3. ¿A qué tipo de nube nos referimos si todo el stack, incluida la aplicación y los datos son gestionados por el proveedor de nube?

  - A. PaaS.
  - B. SaaS.
  - C. IaaS.
  - D. Pública.

**4.** La infraestructura como servicio:

- A. Consiste en la entrega de aplicaciones completas como un servicio.
- B. Se proporciona un servicio de plataforma con todo lo necesario para dar soporte al ciclo de planteamiento, desarrollo y puesta en marcha de aplicaciones y servicios web a través de esta.
- C. Es hacer uso externo de servidores para espacio en disco, base de datos, ruteadores, *switches*, así como tiempo de cómputo, evitando de esta manera tener un servidor local y toda la infraestructura necesaria para la conectividad y mantenimiento dentro de una organizaron.
- D. Ninguna de las definiciones corresponden.

**5.** No es un proveedor de Infraestructura de nube pública:

- A. IBM.
- B. AWS.
- C. Netflix.
- D. Google.

**6.** Selecciona la respuesta correcta en referencia a Microsoft Azure:

- A. Azure ofrece exclusivamente servicios de almacenamiento.
- B. Azure ofrece servicios de nube centrados exclusivamente en Windows.
- C. Azure ofrece servicios de Cloud Computing basados tanto en Windows como en Linux, además de almacenamiento y múltiples productos en un modelo de pago por uso.
- D. Todas las anteriores son falsas.

- 7.** ¿A qué tipo de nube nos referimos si no es propiedad de los usuarios que comparten infraestructura y su uso en principalmente desde Internet?
- A. Pública.
  - B. PaaS.
  - C. IaaS.
  - D. Privada.
- 8.** ¿Cómo afecta la nube al desarrollo e implantación de DevOps?
- A. Proveer entornos bajo demanda.
  - B. Aumenta el marketing.
  - C. Reduce la necesidad de DevOps.
  - D. No hay relación.
- 9.** ¿Cómo es el consumo frente al aprovisionamiento de recursos cuando se utiliza la nube?
- A. Aumenta de forma descontrolada.
  - B. Los recursos aprovisionados crecen en paralelo con la demanda.
  - C. No hay relación.
  - D. Desciende de forma progresiva.
- 10.** ¿Qué tecnología facilita el aprovisionamiento y configuración de un entorno de producción?
- A. La nube pública.
  - B. La nube privada.
  - C. La nube híbrida.
  - D. Cualquiera de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 7. Amazon Web Services

# Índice

[Esquema](#)

[Ideas clave](#)

[7.1. Introducción y objetivos](#)

[7.2. ¿Qué es AWS?](#)

[7.3. Control de costes en AWS](#)

[7.4. Despliegues en AWS](#)

[7.5. Automatización](#)

[7.6. Seguridad](#)

[7.7. Referencias bibliográficas](#)

[A fondo](#)

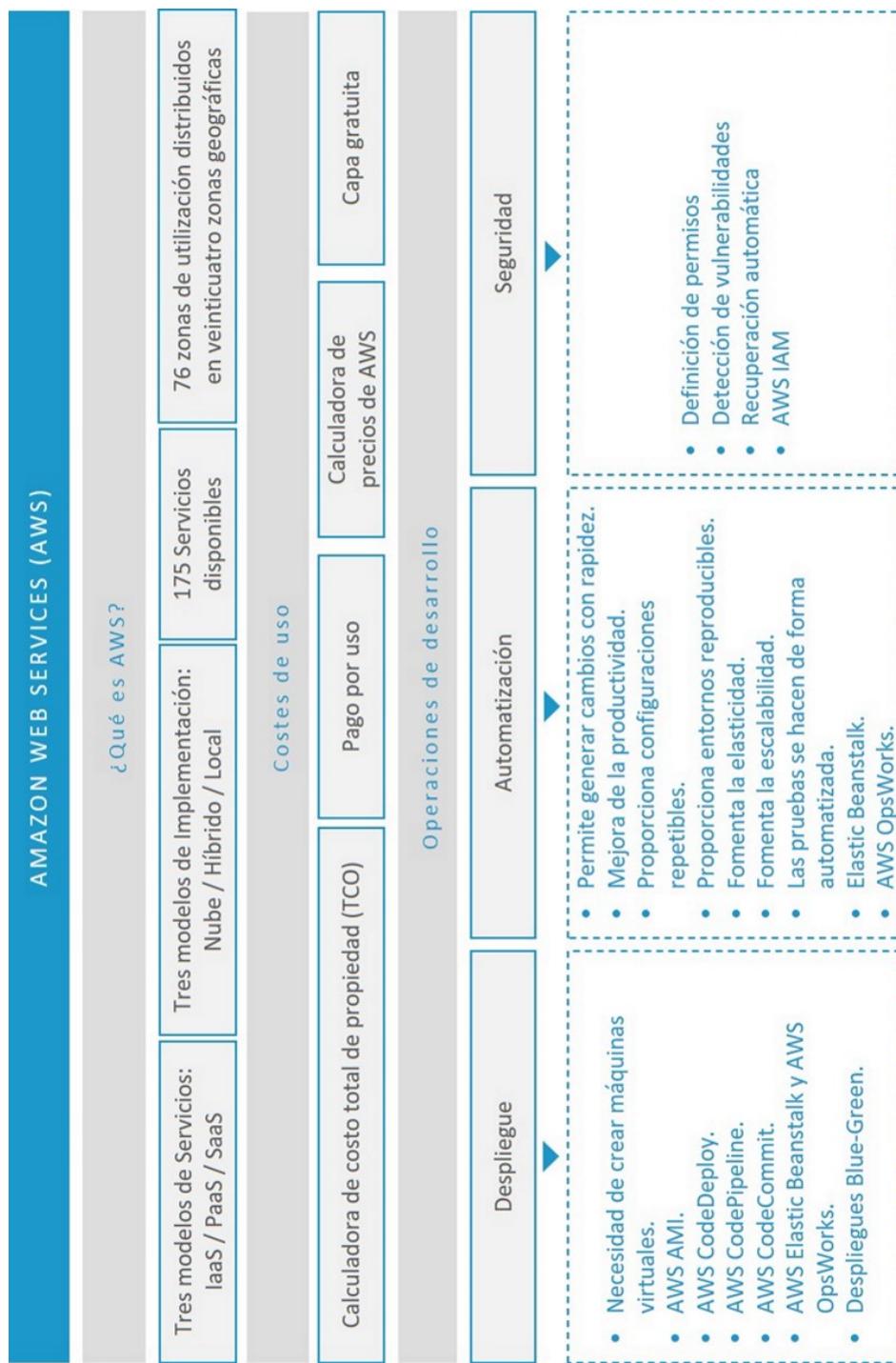
[Documentación oficial AWS](#)

[Comparativa y crecimiento de AWS frente a sus competidores](#)

[SAP on AWS](#)

[Desplegando y desarrollando aplicaciones modernas en la nube](#)

[Test](#)



## 7.1. Introducción y objetivos

En este tema haremos una aproximación a los conceptos fundamentales de AWS y las capacidades esenciales que proporciona.

Los objetivos que se pretenden conseguir son:

- ▶ Conocer qué es AWS.
- ▶ Adquirir el conocimiento a alto nivel sobre qué podemos realizar con AWS.
- ▶ Descubrir cómo realizar un buen control de costes y de la capa gratuita de AWS.
- ▶ Establecer un plan para automatizar el ciclo de vida en AWS.
- ▶ Aprender a desplegar en AWS.
- ▶ Conocer cuáles son los controles de seguridad en AWS.

## 7.2. ¿Qué es AWS?

Amazon Web Services (en adelante, AWS) es una plataforma de servicios en la nube que provee un gran número de servicios de infraestructuras, tales como:

- ▶ Almacenamiento.
- ▶ Comunicaciones.
- ▶ Bases de datos (BBDD).
- ▶ Macrodatos (Big Data).
- ▶ Aprendizaje automático (*machine learning*).
- ▶ Servicios de movilidad.
- ▶ Soluciones empresariales pre-paquetizadas y certificadas.
- ▶ Securización.
- ▶ Gestión de identidades.

Dichas infraestructuras proporcionan a las empresas un crecimiento y control de gastos a medida y, sobre todo, aportan escalabilidad.

En la actualidad, AWS dispone de tres modelos de servicios compatibles entre ellos que proporcionan diferentes soluciones, según las necesidades de cada proyecto y empresa. Veamos, a continuación, cuáles son:

## Infraestructura como servicio (en siglas, IaaS)

Este modelo está relacionado con la infraestructura de tecnología de la información (en siglas, TI) y proporciona una gran flexibilidad al cliente, ya que le permite decidir cuáles son los recursos que realmente necesita (procesador, memoria RAM, disco, seguridad, copia de seguridad, etc.) y pagar por ellos.

AWS ofrece las siguientes prestaciones a sus clientes:

- ▶ **Creación, mantenimiento y actualización de las infraestructuras** de los centros de datos de forma regular.
- ▶ **Securización** frente actores externos.
- ▶ **Creación de un punto de acceso** para los clientes puedan acceder a sus recursos asignados.
- ▶ **Provisión de herramientas de gestión** para la administración de infraestructura.
- ▶ **Provisión de un mapa de su arquitectura.**

Los clientes, por su parte, tienen las siguientes tareas a cargo:

- ▶ Decidir la arquitectura adecuada a sus necesidades.
- ▶ Instalar, mantener y actualizar el sistema operativo (en siglas, SO) y programas específicos.
- ▶ Administrar la configuración de comunicaciones y *firewalls*.
- ▶ Configurar las autorizaciones y métodos de autenticación.
- ▶ Cifrar de los datos y gestión de *backups*.

Dentro de los proveedores de servicios de infraestructura, actualmente AWS se encuentra en la primera posición en el cuadrante de Gartner, como se puede ver en la siguiente imagen:



Figura 1. Cuadrante de Gartner. Fuente: Gartner. (2019, julio). Fuente: Amazon Web Services, s.f.

## Plataforma como servicio (en siglas, PaaS)

Cuando hablamos de plataforma como servicio (en adelante, PaaS), hacemos referencia a **un servicio que se encuentra en la nube del proveedor, que proporciona al cliente un entorno de desarrollo junto con las herramientas necesarias para construir nuevas aplicaciones**. Cuando hablamos de PaaS nos referimos al nexo entre la infraestructura IaaS y el software como servicio (en siglas, SaaS). PaaS proporciona al cliente un conjunto de herramientas y entornos con los cuales diseñar, probar, automatizar y desplegar un producto concreto sin necesidad

de preocuparse de los recursos de este.

Es importante resaltar que el modelo PaaS se encuentra muy ligado al modelo IaaS y, gracias a ello, no hay que preocuparse de la adquisición y gestión de la infraestructura, sino solo de la plataforma de desarrollo. Existen diferentes ámbitos en los cuales es posible utilizar PaaS, además de para el desarrollo de aplicaciones, como son:

- ▶ Creación de *Application Programming Interfaces* (en siglas, API).
- ▶ Análisis de un gran volumen de datos.
- ▶ Implantación de sistemas de gestión empresarial como pueden ser SAP, Oracle.
- ▶ Plataforma de comunicación.
- ▶ Base de datos.
- ▶ Dentro de los proyectos de innovación, como ejemplo para la implementación de soluciones de Internet de las cosas.

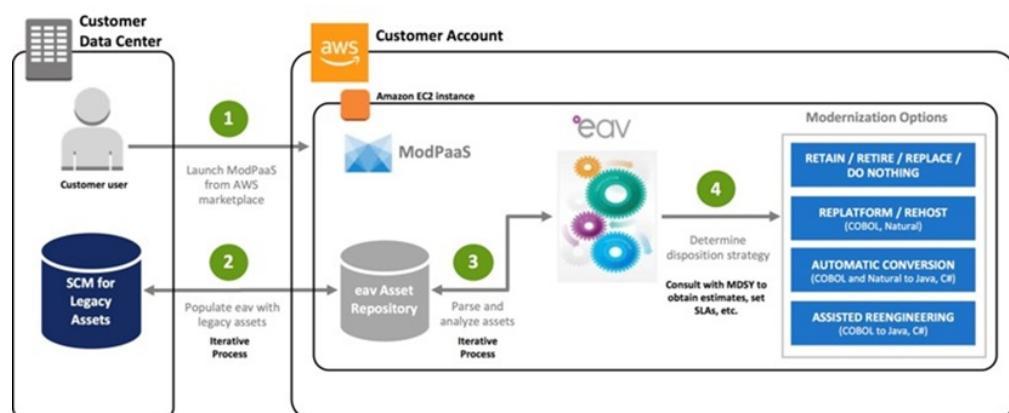


Figura 2. Ejemplo de PaaS. Fuente: de Valence, 2018.

## Software como servicio (SaaS)

Proporciona un producto completo al cliente, el cual ejecuta y administra el

proveedor. En este caso, el cliente solo debe preocuparse por sus necesidades de software. **Es una forma de disponer de diferentes tipos de software de forma centralizada y el usuario final puede acceder a este de forma online.** Como programas o softwares más destacados dentro de los servicios SaaS, podemos destacar SAP, CRM, gestores de proyectos o gestores de archivos.

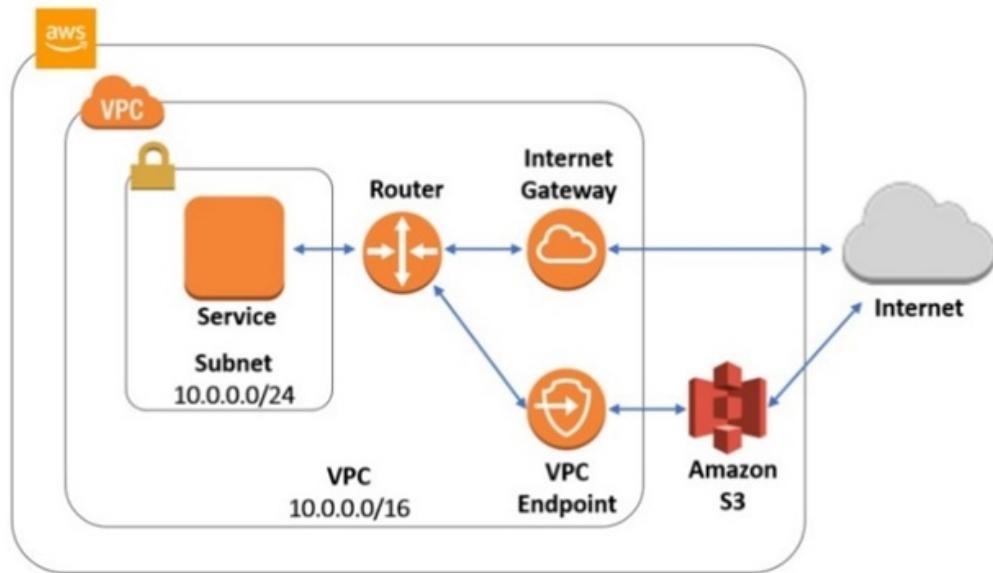
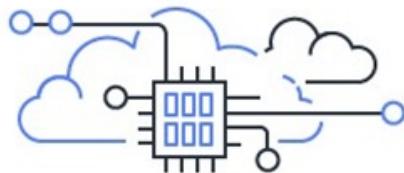


Figura 3. Ejemplo de SaaS. Fuente: Golding, 2017.

A su vez, existen tres modelos de implementación de IaaS, PaaS y SaaS:



## Nube

Una aplicación basada en la nube se encuentra implementada totalmente en la nube, de modo que todas las partes de la aplicación se ejecutan en esta. Las aplicaciones en la nube se han creado directamente en la nube o se han transferido de la infraestructura existente para aprovechar los **beneficios de la informática en la nube**. Las aplicaciones basadas en la nube se pueden construir en partes de infraestructura de bajo nivel o pueden utilizar servicios de nivel superior que proporcionan abstracción de los requisitos de administración, arquitectura y escalado de la infraestructura principal.

Figura 4. Modelos de implementación informática en la nube. Fuente: adaptado de Amazon Web Services,

s. f.



## Solución híbrida

Una implementación híbrida es una manera de conectar la infraestructura y las aplicaciones entre los recursos basados en la nube y los recursos existentes situados fuera de la nube. El método más común de implementación híbrida consiste en conectar la nube y la infraestructura existente en las instalaciones para ampliar e incrementar la infraestructura de la organización en la nube al mismo tiempo que se conectan estos recursos en la nube con el sistema interno. Para obtener más información sobre cómo AWS lo puede ayudar a establecer una implementación híbrida, visite nuestra página acerca de la [nube híbrida](#).

Figura 5. Modelos de implementación informática en la nube. Fuente: adaptado de Amazon Web Services, s. f.



## En las instalaciones

La implementación local de recursos mediante herramientas de administración de recursos y virtualización se denomina a veces "nube privada". La implementación local no aporta muchos de los beneficios de la informática en la nube, pero a veces se utiliza por su capacidad de ofrecer **recursos dedicados**. En la mayoría de los casos, este modelo de implementación es idéntico al de la infraestructura de TI antigua, mientras que utiliza tecnologías de virtualización y administración de aplicaciones para intentar incrementar el uso de los recursos.

Figura 6. Modelos de implementación de informática en la nube. Fuente: adaptado de Amazon Web Services, s. f.

En la actualidad, AWS proporciona más de 175 servicios combinables entre ellos, que se agrupan de la siguiente manera:

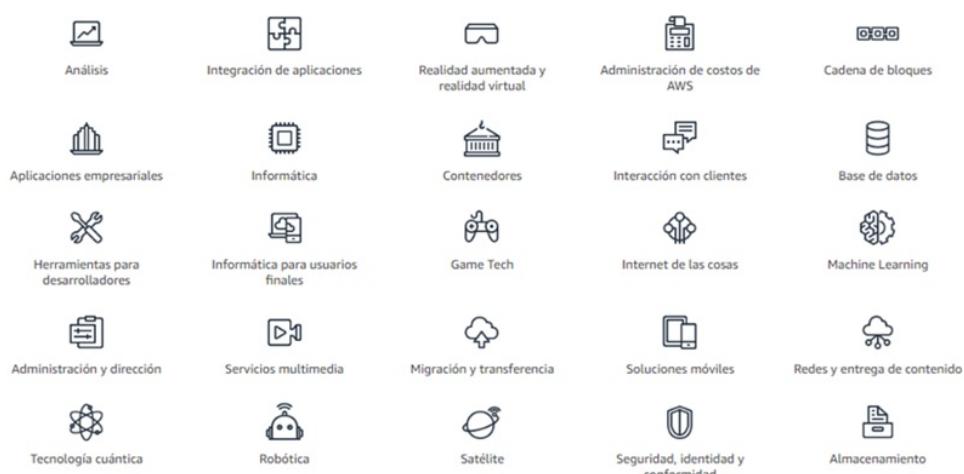


Figura 7. Grupo de Productos AWS. Fuente: adaptado de Amazon Web Services, s. f.

A continuación, pasamos a detallar algunos de los grupos dentro del abanico de servicios de AWS:

Integración de aplicaciones	
	1) AWS Step Functions <ul style="list-style-type: none"> <li>▣ Creación de aplicaciones distribuidas con flujos de trabajo visuales</li> <li>▣ Automatización de procesos</li> <li>▣ Mejora la resiliencia</li> </ul>
	2) Amazon MQ <ul style="list-style-type: none"> <li>▣ Agente de mensajes administrado para Apache ActiveMQ</li> <li>▣ Uso protocolos JMS, NMS, AMQP, STOMP, MQTT y WebSocket</li> <li>▣ No necesita adaptación código existente</li> </ul>
	3) Amazon AppSync <ul style="list-style-type: none"> <li>▣ API para acceder a diferentes orígenes de datos</li> <li>▣ Utilización de GraphQL</li> <li>▣ Proporciona información offline</li> </ul>
	4) Amazon AppFlow <ul style="list-style-type: none"> <li>▣ Integración de apps sin necesidad de código</li> <li>▣ Creación automática de flujos de trabajo</li> <li>▣ Análisis de eventos</li> <li>▣ Nutrido de datos para lagos de datos</li> </ul>
	5) Amazon Simple Notification Services (SNS) <ul style="list-style-type: none"> <li>▣ Simplificación de arquitectura de mensajes</li> <li>▣ Políticas de seguridad</li> <li>▣ Desacoplamiento de microservicios</li> <li>▣ Distribución de notificaciones a usuarios finales</li> </ul>
	6) Amazon EventBridge <ul style="list-style-type: none"> <li>▣ Creación de aplicaciones basadas en eventos</li> <li>▣ Creación de esquemas sin necesidad de generar código</li> <li>▣ Escalable automáticamente dependiendo del número de eventos creados</li> <li>▣ Reducción de gastos</li> </ul>
	7) Amazon Simple Queue Service (Amazon SQS) <ul style="list-style-type: none"> <li>▣ Servicio de cola de mensajes</li> <li>▣ Creación dinámica</li> <li>▣ Escalable</li> <li>▣ Cifrado completo punto a punto de los mensajes</li> <li>▣ Procesamiento estándar y FIFO</li> </ul>

Tabla 1. Integración de aplicaciones. Fuente: elaboración propia.

## Contenedores

- |  |   |
|--|---|
| <p>1) Amazon Elastic Container Registry</p> <ul style="list-style-type: none"><li>▣ Es un almacén de contenedores <b>Docker</b> que proporciona a los desarrolladores el almacenamiento, administración e implementación de imágenes de contenedores de Docker de forma ágil y escalable.</li></ul> <p>2) AWS Fargate</p> <ul style="list-style-type: none"><li>▣ Motor computacional que proporciona la posibilidad de ejecutar sin necesidad de servidor, es decir, no es necesario un aprovisionamiento. El pago se realiza según los recursos que sean necesario para ejecutar los contenedores.</li><li>▣ Ejecuta las tareas o <i>pods</i> en un kernel dedicado y da, a su vez, un entorno aislado y securizado.</li></ul> | <p>3) Amazon Elastic Container Service (en siglas, ECS)</p> <ul style="list-style-type: none"><li>▣ Servicio de orquestación de contenedores totalmente administrable, complementario a <b>AWS Fargate</b>. Es un servicio de computo sin servidor dedicado. El pago es por lo que se necesita.</li><li>▣ Integración de forma nativa con otros servicios de AWS como: <b>Amazon Route 53, Secrets Manager, AWS Identity and Access Management (IAM)</b> y <b>Amazon CloudWatch</b>.</li></ul> <p>4) Amazon Elastic Kubernetes Service (en siglas, EKS)</p> <ul style="list-style-type: none"><li>▣ Proporciona un servicio de <b>Kubernetes</b> con el que es posible sacar partido de todas las herramientas de código abierto, con una fácil migración sin necesidad de refactorización.</li><li>▣ Integrado con <b>Amazon CloudWatch, grupos de Auto Scaling, AWS Identity and Access Management (en siglas, IAM)</b> y <b>Amazon Virtual Private Cloud (en siglas, VPC)</b>, con una integración nativa con <b>AWS App Mesh</b>.</li></ul> |
|--|---|

Tabla 2. Contenedores. Fuente: elaboración propia.

<p><b>1) Amazon EC2</b></p> <ul style="list-style-type: none"> <li>■ Nos proporciona servidores virtuales en la nube según características, necesidades o demandas del producto o la empresa.</li> </ul> <p><b>2) AWS Batch</b></p> <ul style="list-style-type: none"> <li>■ Servicio que nos permite ejecutar <i>jobs</i> programados por lotes. Dicho servicio aprovisiona los recursos necesarios para la ejecución de esos <i>Jobs</i> según los requisitos específicos de cada <i>job</i>.</li> <li>■ El coste de dicho servicio dependerá del número de <i>job</i> por lotes que sea necesario ejecutar.</li> </ul> <p><b>3) AWS Outposts</b></p> <ul style="list-style-type: none"> <li>■ Proporciona la posibilidad de trasladar servicios, infraestructura y modelos operativos de AWS a cualquier otro modelo, ya sea híbrido o local.</li> </ul> <p><b>4) Vmware Cloud on AWS</b></p> <ul style="list-style-type: none"> <li>■ Vmware y AWS se han unido con la posibilidad de ofrecer todos los servicios de AWS en un modelo de despliegue híbrido sin necesidad de hardware personalizado.</li> </ul> <p><b>5) Amazon EC2 Auto Scaling</b></p> <ul style="list-style-type: none"> <li>■ Servicio que podemos adaptar de los servicios contratados de AWS a partir de los niveles de demanda actuales, es decir, podemos ampliar y disminuir el cómputo contratado dependiendo del nivel de demanda actual.</li> </ul>	<p><b>6) AWS Elastic Beanstalk</b></p> <ul style="list-style-type: none"> <li>■ Servicio en el que podemos implementar y escalar servicios y aplicaciones web que estén desarrollados en los siguientes lenguajes: Java, .NET, PHP, Node.js, Python, Ruby, Go y Docker en servidores familiares como Apache, Nginx, Passenger e IIS.</li> </ul> <p><b>7) AWS Serverless Application Repository</b></p> <ul style="list-style-type: none"> <li>■ Con dicho servicio es posible implementar, paquetizar, almacenar y reutilizar desarrollos, sin necesidad de un servidor dedicado.</li> </ul> <p><b>8) Amazon Lightsail</b></p> <ul style="list-style-type: none"> <li>■ Una plataforma de bajo coste con la que es posible desarrollar una aplicación o un sitio web. Amazon Lightsail proporciona servidores virtuales, almacenamiento, bases de datos y las comunicaciones en un pack.</li> </ul> <p><b>9) AWS Lambda</b></p> <ul style="list-style-type: none"> <li>■ AWS Lambda nos proporciona la posibilidad de ejecutar cualquier código desarrollado sin necesidad de realizar tareas de administración. Solo es necesario cargar el código a ejecutar y AWS Lambda se encarga de escalarlo y ejecutarlo, pudiendo ser llamado desde otros servicios, aplicaciones desplegadas en un sitio web.</li> </ul> <p><b>10) AWS Wavelength</b></p> <ul style="list-style-type: none"> <li>■ Servicio de alto rendimiento que proporciona una latencia mínima en la ejecución de aplicaciones desarrolladas en dispositivos 5G, lo que proporciona un salto mínimo de red a la hora de ejecutar dicha aplicación.</li> </ul>
---	--

Informática

Tabla 3. Informática. Fuente: elaboración propia.

Migración y transferencia	
	<b>1) AWS Migration Hub</b>
	<ul style="list-style-type: none"> <li>▣ Proporciona herramientas propias de AWS o herramientas de terceros para la realización de migraciones a la nube, brindando así, un seguimiento personalizado, por lo que se consiguen una flexibilidad y visibilidad totales.</li> </ul>
	<b>2) AWS DataSync</b>
	<ul style="list-style-type: none"> <li>▣ Con este servicio es posible mover gran cantidad de datos entre nuestro <i>datacenter</i> y Amazon S3, Amazon Elastic File System (Amazon EFS) o Amazon FSx for Windows File Server. Obtenemos una automatización de tareas manuales, monitoreo de la transferencia, validación y optimización de esta.</li> </ul>
	<b>3) AWS Transfer Family</b>
	<ul style="list-style-type: none"> <li>▣ Servicio que proporciona la transferencia de archivos de forma ágil y sin problemas a Amazon S3. Es compatible con los protocolos SFTP, FTPS y FTP.</li> </ul>
	<b>4) AWS Application Discovery Service</b>
	<ul style="list-style-type: none"> <li>▣ Ayuda a los clientes con sistemas de gestión empresarial a planificar proyectos de migración a la nube proporcionando: un análisis completo de los datos de configuración, uso y comportamiento de los servidores actuales antes de realizar dicha migración.</li> </ul>
	<b>5) AWS Server Migration Service</b>
	<ul style="list-style-type: none"> <li>▣ Servicio sin agente que permite realizar migraciones de cargas de trabajo locales a AWS y automatizar, programar y monitorizar las replicaciones de forma gradual.</li> </ul>
	<b>6) CloudEndure Migration</b>
	<ul style="list-style-type: none"> <li>▣ Da la posibilidad de trasladar, rápidamente, una gran cantidad de máquinas locales a AWS. Para ello, lo que realiza es una replicación de las máquinas origen en un Sandbox y, cuando todo está correcto, se ponen las máquinas en producción sin tiempos de latencia ni cortes de operaciones.</li> </ul>
	<b>7) AWS DataBase Migration Service</b>
	<ul style="list-style-type: none"> <li>▣ Servicio con el que es posible realizar la migración de todas las bases de datos que dispongamos, con un tiempo mínimo de inactividad, admitiendo migraciones homogéneas y heterogéneas.</li> </ul>
	<b>8) Familia de productos AWS Snow</b>
	<ul style="list-style-type: none"> <li>▣ Dispositivos físicos para realizar la migración de datos desde y hacia AWS. Para más información acerca de este servicio, es necesario contar con AWS.</li> </ul>

Tabla 4. Migraciones y Transferencias. Fuente: elaboración propia.

<p style="text-align: center;">Almacenamiento</p>	<p><b>1) Amazon Simple Storage Service (S3)</b></p> <ul style="list-style-type: none"> <li>■ S3 proporciona un servicio de almacenamiento en la nube con un gran nivel de seguridad, soporte y control.</li> </ul> <p><b>2) Amazon FSx for Lustre</b></p> <ul style="list-style-type: none"> <li>■ Servicio de sistema de archivos de alto rendimiento, diseñado para dar un alto y rápido almacenamiento a aplicaciones complejas.</li> </ul> <p><b>3) AWS Backup</b></p> <ul style="list-style-type: none"> <li>■ Servicio que proporciona una gestión total para la centralización y automatización de las copias de seguridad de todos los servicios de AWS. Crea políticas de seguridad y las supervisa.</li> </ul> <p><b>4) CloudEndure Disaster Recovery</b></p> <ul style="list-style-type: none"> <li>■ Proporciona servicio que es capaz de recuperar, de una forma rápida y automatizada, la información, ya sea de bases de datos, ERPs como SAP, aplicaciones de sistemas y realiza copias de estos en entornos de bajo coste que pueden ser desplegados en minutos, en caso de que ocurra algún desastre.</li> </ul> <p><b>5) Amazon Elastic Block Store (EBS)</b></p> <ul style="list-style-type: none"> <li>■ Amazon EBS se proporciona una gran gama de cargas de trabajo, como bases de dato relacionales y no relacionales, aplicaciones empresariales, aplicaciones en contenedores, motores de Big Data, sistemas de archivos y flujos de trabajo de medios.</li> </ul> <p><b>6) Amazon FSx for Windows File Server</b></p> <ul style="list-style-type: none"> <li>■ Servicio que proporciona un almacenamiento de archivos totalmente administrado y de alto rendimiento, basado en Windows Server y con una integración directa con Microsoft Active Directory.</li> </ul> <p><b>7) Familia de productos AWS Snow</b></p> <ul style="list-style-type: none"> <li>■ Dispositivos físicos para realizar la migración de datos desde y hacia AWS. Para más información acerca de este servicio, es necesario contactar con AWS.</li> </ul> <p><b>8) Amazon Elastic File System (EFS)</b></p> <ul style="list-style-type: none"> <li>■ EFS proporciona un sistema de archivos NFS ágil, escalable, fácil de usar y totalmente administrable con los servicios de AWS e incluso con recursos en local.</li> </ul> <p><b>9) Amazon S3 Glacier</b></p> <ul style="list-style-type: none"> <li>■ Servicio de almacenamiento en la nube basado en Amazon S3 para el archivado de datos y almacenamiento de copias de seguridad a un bajo coste.</li> </ul> <p><b>10) AWS Storage Gateway</b></p> <ul style="list-style-type: none"> <li>■ Servicio que proporciona un almacenamiento local con un acceso a nube híbrida casi ilimitado, con mecanismos de trasferencia de datos con un gran ancho de banda de transferencia y permitiendo almacenar archivos, volúmenes y cintas en AWS.</li> </ul>
---	---

Tabla 5. Almacenamiento. Fuente: elaboración propia.

Dichos servicios se encuentran disponibles a partir de centros de datos distribuidos alrededor del globo, divididos en 76 zonas disponibles de utilización y repartidos por veinticuatro zonas geográficas, con una zonal local para proporcionar latencias ínfimas, por lo que proporciona una **disponibilidad global**, como se puede ver a continuación:



Figura 8. Infraestructura global AWS. Fuente: Amazon Web Services, s. f.

## 7.3. Control de costes en AWS

Es necesario que las empresas puedan realizar un control exhaustivo de los costes derivados del consumo de servicios, por lo que AWS proporciona **un sistema de pago por uso sin suscripciones**. Esto significa que en el momento en que dejemos de necesitar algún servicio, ya no deberemos pagar por él.

Dentro del catálogo de productos existen los siguientes servicios para el control de costes:

<p><b>Administración de costos AWS</b></p> <ul style="list-style-type: none"> <li><b>1) AWS Cost Explorer</b> <ul style="list-style-type: none"> <li>Conjunto de informes predeterminados con los que podremos analizar los costes totales de los servicios subscriptos a AWS.</li> </ul> </li> <li><b>2) Informes de instancias reservadas</b> <ul style="list-style-type: none"> <li>Solución que proporciona AWS para administrar y monitorear las reservas de <b>Amazon EC2, Amazon RDS, Amazon Elasticsearch, Amazon ElastiCache y Amazon Redshift</b>.</li> </ul> </li> <li><b>3) Presupuesto AWS</b> <ul style="list-style-type: none"> <li>AWS proporciona un panel de control con el cual es posible crear, monitorizar e inspeccionar los presupuestos creados de contratación de servicios, e incluso ver su estado. Dichos presupuestos se pueden crear en modalidad mensual, trimestral o anual.</li> <li>Posibilidad de crear alertas en caso de superar los presupuestos contratados.</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li><b>4) Saving Plans</b> <ul style="list-style-type: none"> <li>Modelo creado por AWS que ofrece un ahorro en los servicios e instancias contratadas a condición de una suscripción mínima de uno a tres años, con el compromiso de utilizar una cantidad específica de cómputo. Dicho ahorro puede llegar a ser hasta del 70 %.</li> </ul> </li> <li><b>5) Informe de uso y costo AWS</b> <ul style="list-style-type: none"> <li>Contiene la información completa del costo y la utilización de los servicios contratados. Dicha información puede ser visualizada por horas, días, semanas, meses y años.</li> </ul> </li> </ul>
---	---

Tabla 6. Administración de Costes AWS. Fuente: elaboración propia.

Adicionalmente a estos servicios, antes de empezar a utilizar AWS es necesario tener un presupuesto del servicio o conjunto de servicios a utilizar. A este fin, **AWS proporciona dos calculadoras** para la realización de presupuestos y análisis del

coste de la utilización de servicios individuales, paquetes o, incluso, de soluciones completas.

## Calculadora de precios de AWS

Es posible realizar una estimación de costos de los productos y servicios que ofrece AWS adaptados a las necesidades de la empresa, o incluso de forma personal. La utilización de esta es totalmente gratuita y para usarla deberemos seguir los siguientes pasos:

- ▶ Acceder a [AWS Pricing Calculator](#) y seleccionar el botón **crear estimación**.
- ▶ Seleccionar el o los **servicios** en los que estemos interesados, la **configuración** que necesitamos e indicar en qué **región** lo queremos activado.

Una vez realizado el presupuesto, se podrá guardar o compartir con quien sea necesario y podremos tener vinculados tantos presupuestos como necesitemos o queramos crear, sin que estos sean vinculantes.

## Calculadora de coste total de propiedad (TCO)

La calculadora de coste total de propiedad permite realizar una evaluación, o análisis de ahorro, en la utilización de productos y servicios de AWS. Realiza una comparación con entornos locales o híbridos de forma completa y realista. Para utilizar esta calculadora, se deben seguir los siguientes pasos:



Figura 9. Calculadora de coste total de propiedad (TCO) de AWS. Fuente: elaboración propia.

## Capa gratuita AWS

Dentro del ecosistema AWS, existe una capa gratuita que proporciona más de sesenta productos, que pueden probarse o utilizarse sin ningún coste. Existen tres tipos de ofertas dentro de la capa gratuita, dependiendo del tipo de producto que se quiera utilizar.

- ▶ **Gratis para siempre:** oferta disponible para clientes existentes o nuevos clientes. Pasados los doce meses, estas ofertas no vencerán. Gracias a esto, será posible empezar a familiarizarse con AWS.
- ▶ **Doce meses de uso gratuito:** consiste en una oferta disponible durante doce meses a partir de la fecha de suscripción y solo para nuevos usuarios. Dichas ofertas tienen un límite de cómputo y, en el caso de que se supere dicho computo, será necesario pagar las tarifas estipuladas para dicho servicio.
- ▶ **Pruebas:** es una oferta exclusiva para pruebas que comienza desde el primer uso de estas y, una vez que haya expirado el plazo de prueba, al igual que en la modalidad anterior, se empezará a tarificar según el uso del servicio.

---

Para poder ver todos los servicios disponibles en las modalidades anteriormente explicadas, puedes acceder a la capa gratuita de AWS a través de la siguiente dirección web: <https://aws.amazon.com/es/free/>

---

**Detalles de la capa gratuita**

Filtrar por:

COMPUTACIÓN		COMPUTACIÓN		HERRAMIENTAS PARA DESARROLLADORES	
Capa gratuita	12 MESES GRATIS	Capa gratuita	GRATUITO PARA SIEMPRE	Capa gratuita	GRATUITO PARA SIEMPRE
<b>Amazon EC2</b> <b>750 horas</b> al mes		<b>AWS Lambda</b> <b>1 millón</b> solicitudes gratuitas al mes		<b>Amazon CloudWatch</b> <b>10</b> alarmas y métricas personalizadas	
Capacidad de cómputo de tamaño variable en la nube.		Servicio informático que ejecuta su código como respuesta a eventos y se admite la monitorización automática los		Monitoreo de recursos y aplicaciones en la nube de AWS.	
750 horas por mes de uso de instancias t2.micro con Linux, RHEL o SLES					
750 horas por mes de uso de instancias t2.micro con Windows					
Capa gratuita		Capa gratuita		Capa gratuita	
<b>Amazon API Gateway</b> <b>1 millón</b> de llamadas a la API recibidas al mes		<b>Amazon Elastic Container Registry</b> <b>500 MB</b> al mes de almacenamiento		<b>Amazon MQ</b> <b>750 horas</b> de un agente mq.t2.micro con una única instancia al mes	
Públique, mantenga, monitoree y proteja API a cualquier escala.		Almacene y recupere imágenes de Docker.		Amazon MQ es un servicio de agente de mensajes administrado para Apache ActiveMQ.	
1 millón de llamadas de API recibidas por mes					

**Categorías de productos**

- Análisis
- Integración de aplicaciones
- RA y RV
- Productividad empresarial
- Informática
- Interacción con clientes
- Base de datos
- Herramientas para desarrolladores
- Informática para usuarios finales
- Game Tech
- Internet de las cosas
- Machine Learning
- Administración y dirección
- Servicios multimedia
- Soluciones para dispositivos móviles
- Conexión en red y entrega de contenido
- Robótica
- Seguridad, identidad y conformidad
- Almacenamiento

Figura 10. Ejemplo de servicios capa gratuita. Fuente: Amazon Web Services, s.f.

## 7.4. Despliegues en AWS

Desde un punto de vista simplista, realizar un despliegue básico en Amazon implica la creación de una máquina virtual. Para la creación de dicha máquina virtual, son necesarios algunos de los conceptos que veremos a continuación.

### AWS AMI

**Amazon Machine Image (en adelante, AMI)** es un ejemplo de esta «infraestructura como código». Este componente fundamental de la computación de AWS es una especie de plantilla digital que puede lanzar (aprovisionar) instancias de Amazon EC2, el entorno fundamental de AWS de cómputo en la nube. Desde un punto de vista práctico, podemos decir que en la mayoría de los casos el AMI es el sistema operativo.

Se puede elegir entre tres tipos de AMI:

- ▶ Publicada por AWS.
- ▶ De terceros.
- ▶ Personalizada.

**AWS publica las AMI** que contienen configuraciones comunes de software basado en sistemas operativos populares como Linux y Microsoft Windows. Las **AMI** también pueden obtenerse a partir de terceros, algunos de los cuales están disponibles en el

AWS Marketplace. Una organización también puede crear y publicar sus propias **AMI personalizadas**.

Las AMI específicas de una organización incluyen, generalmente: software corporativo (sistemas operativos reforzados), software antivirus y suites de software

de oficina. Tienen la posibilidad de incluir software de aplicación que viene empaquetado junto con la AMI o puede contener scripts y software que permitan a la instancia de instalar el software de aplicación al momento del lanzamiento, llamada *booting*.

¿Cuáles son los pros y contras de cargar, de antemano, software de nivel de aplicación en una AMI? Como ventaja, hemos de indicar que pueden **ser lanzados con gran rapidez**, ya que no es necesario instalar ningún software adicional en el arranque. Como desventaja, hay que indicar que puede ser necesario **crear una nueva AMI cada vez que haya cambios** de software a nivel de aplicación.

## AWS CodeDeploy

El despliegue continuo es otro concepto fundamental en una estrategia de DevOps. Su objetivo principal es permitir el despliegue automático del código de aplicación «listo para producción». Algunas veces, al despliegue continuo se lo llama entrega continua. La única diferencia es que el despliegue continuo, por lo general, se refiere a los despliegues de producción.

Mediante el uso de las prácticas y herramientas de entrega continua, el software se puede implementar rápidamente, de forma reiterativa y fiable. Si un despliegue falla, se puede hacer *roll-back* automáticamente a la versión anterior. Un buen ejemplo de este principio en AWS es el servicio de implementación de código AWS CodeDeploy. Sus características principales proporcionan la habilidad de **desplegar aplicaciones a través de Amazon EC2 con un tiempo de inactividad mínimo, centralizando el control e integrándose con la versión existente de software o proceso de entrega continua**.

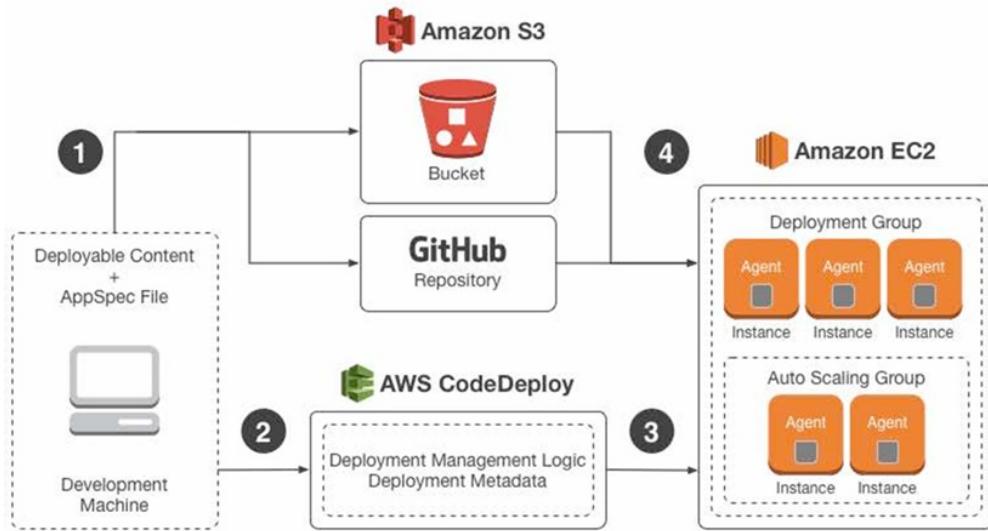


Figura 11. AWS CodeDeploy. Fuente: SupportPRO, s.f.

Veamos cómo funciona:

- ▶ El contenido de la aplicación se empaqueta y se despliega en Amazon S3 junto con un archivo de Aplicación Específica (AppSpec) que define una serie de pasos de despliegue que AWS CodeDeploy necesita ejecutar. El paquete se llama «CodeDeploy revision».
- ▶ Se crea una aplicación en AWS CodeDeploy y se definen las instancias en las que la aplicación debe ser desplegada (**DeploymentGroup**). La aplicación también define el *bucket* de Amazon S3, donde reside el paquete de despliegue.
- ▶ Un agente de AWS CodeDeploy se implementa en cada instancia Amazon EC2 participante. Los agentes de AWS CodeDeploy determinan qué y cuándo aplicar una revisión del depósito de Amazon S3 especificado.
- ▶ El agente AWS CodeDeploy obtiene el código de la aplicación empaquetada y lo despliega en la instancia.

## AWS CodePipeline

AWS CodePipeline es un **servicio de automatización de entrega continua** que ayuda a suavizar los despliegues. Se puede diseñar el desarrollo del flujo de trabajo para: la comprobación del código, la construcción del código, el despliegue de la aplicación en los distintos escenarios, las pruebas y el pase a producción. Se pueden integrar herramientas de terceros en cualquier etapa del proceso, o bien, se puede utilizar AWS CodePipeline como una solución de extremo a extremo. Con esta herramienta, se pueden realizar rápidamente actualizaciones y *features* (características) de alta calidad y tiene varios beneficios que se alinean con el principio de DevOps de despliegue continuo:

- ▶ Entrega rápida.
- ▶ Mejora de la calidad.
- ▶ Flujo de trabajo configurable.
- ▶ Facilidad de integración.

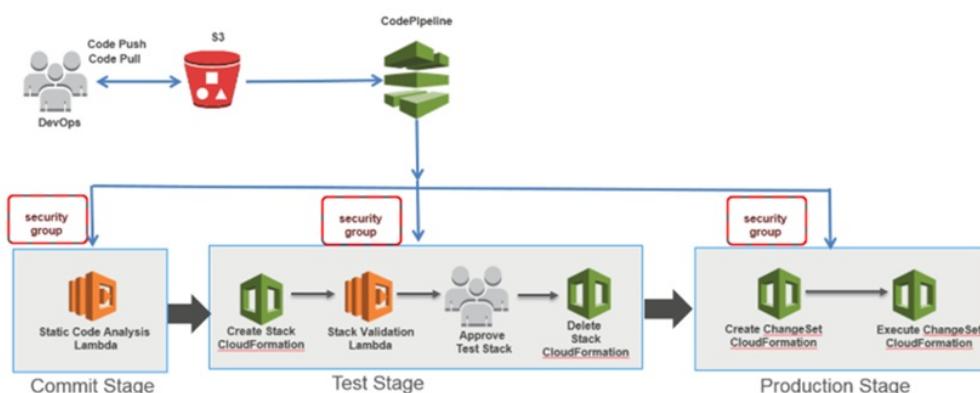


Figura 12. Implementación de DevSecOps con AWS CodePipeline. Fuente: Adabala, 2017.

## AWS CodeCommit

AWS CodeCommit es un **servicio de gestión de control de fuente** que aloja los repositorios privados Git y que se caracteriza por ser seguro y escalable. Elimina la

necesidad de operar con el sistema de control de código fuente propio o tener que preocuparse acerca de la expansión de la infraestructura. Además, se puede utilizar para almacenar cualquier cosa, desde código hasta números binarios, y es compatible con la funcionalidad estándar de Git, lo que permite trabajar sin problemas con las herramientas existentes, basadas en este último. Para trabajar en equipo, también se pueden utilizar las herramientas de código online de CodeCommit para navegar, editar y colaborar en proyectos. Entre sus beneficios, vale la pena destacar:

- ▶ Gestión completa.
- ▶ Almacenamiento flexible.
- ▶ Alta disponibilidad.
- ▶ Ciclos de vida de desarrollo más rápidos.
- ▶ Funcionamiento adecuado con las herramientas existentes.
- ▶ Seguridad.

## AWS Elastic Beanstalk y AWS OpsWorks

Tanto AWS OpsWorks como AWS Elastic Beanstalk soportan la implementación continua de cambios en el código de aplicación y modificaciones en la infraestructura. En AWS Elastic Beanstalk, los despliegues de cambio de código se almacenan como «versiones de las aplicaciones» y los cambios en la infraestructura se despliegan como «configuraciones guardadas». Un ejemplo de una versión de aplicación sería una nueva aplicación Java que se carga como un archivo ZIP o WAR. Un ejemplo de una configuración guardada sería una configuración de AWS Elastic Beanstalk, que utiliza Elastic Load Balancing y Auto Scaling en lugar de una sola instancia.

**AWS Elastic Beanstalk** apoya la práctica DevOps llamada *rolling deployments*. Cuando se activa, la configuración de los despliegues trabaja mano a mano con Auto Scaling para asegurar que siempre haya un número definido de instancias disponibles cuando se realizan cambios de configuración. Esto aporta control cuando las instancias de Amazon EC2 se actualizan. Por ejemplo, si se está cambiando el tipo de instancia EC2, el usuario puede determinar si AWS Elastic Beanstalk debe actualizar todas las instancias al mismo tiempo o si se quieren dejar algunas instancias corriendo.

## Despliegue Blue-Green

El despliegue Blue-Green es una práctica de despliegue de DevOps que utiliza los servicios de nombres de dominio (en siglas, DNS) para realizar el despliegue de aplicaciones. **La estrategia consiste en comenzar con un entorno existente (azul) mientras se prueba uno nuevo (verde)**. Cuando el nuevo entorno ha pasado todas las pruebas necesarias y está listo para utilizarse, simplemente hay que redirigir el tráfico desde el antiguo entorno a la nueva vía DNS.

AWS ofrece todas las herramientas que se necesitan para implementar una estrategia de desarrollo Blue-Green. Se puede configurar un entorno ideal de nueva infraestructura mediante el uso de un servicio como AWS CloudFormation o AWS Elastic Beanstalk. Con las plantillas de CloudFormation AWS, se puede crear fácilmente un nuevo entorno, idéntico al entorno de producción existente.

Si se utiliza el servicio AWS DNS Amazon Route 53, se puede dirigir el tráfico a través de conjuntos de registros de recursos ponderados. Mediante el uso de estos conjuntos, se pueden definir varios servicios o equilibradores de carga (*load balancers*) con una resolución de DNS. La resolución del servicio DNS (conversión de un nombre de dominio a una dirección IP) es ponderada, lo que significa que se puede definir la cantidad de tráfico que se dirige al entorno de producción. De esta forma, se pueden hacer pruebas del entorno y, cuando se tiene la certeza de que el

despliegue es bueno, aumentar la ponderación. Cuando el entorno de producción antiguo está recibiendo 0 % de tráfico, se puede guardar como copia de seguridad, o bien desmantelarse. A medida que la cantidad de tráfico en los nuevos entornos aumenta, se puede utilizar Auto Scaling para ampliar las instancias adicionales de Amazon EC2. Esta capacidad de crear y disponer de entornos idénticos fácilmente en la nube de AWS posibilita la aplicación de las prácticas de DevOps como el despliegue Blue-Green. También, se puede utilizar el despliegue Blue-Green para los servicios de *back-end* como el despliegue de bases de datos y conmutación por error.

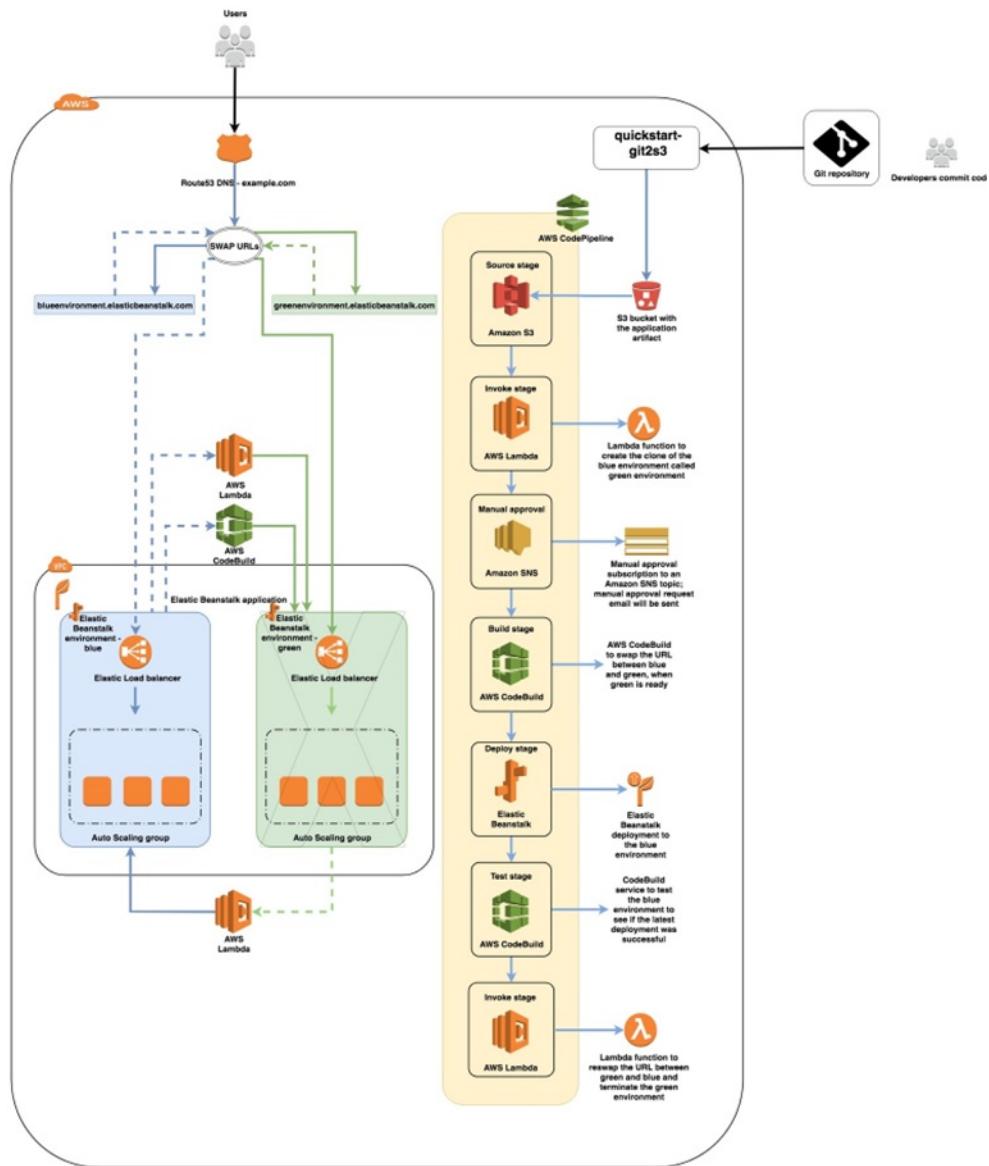


Figura 13. Implementación azul-verde en AWS. Fuente: Abdala, 2017.

Accede a la Figura 13, «Implementación azul-verde en AWS», a través del aula virtual.

## 7.5. Automatización

Otra filosofía y práctica fundamental de DevOps es la automatización. **La automatización se centra en la instalación, configuración, implementación y soporte de la infraestructura y las aplicaciones que se ejecutan en ella.** Mediante el uso de la automatización, se pueden configurar entornos con mayor rapidez de una manera estandarizada y repetible.

La eliminación de los procesos manuales es la clave para una estrategia DevOps exitosa. Históricamente, **la configuración del servidor y la implementación de aplicaciones han sido procesos manuales, pero es sabido que la automatización es fundamental para obtener el máximo beneficio de la nube.** AWS se apoya, en gran medida, en la automatización para proporcionar las características básicas de elasticidad y escalabilidad. Los procesos manuales son propensos a errores, poco fiables e insuficientes para soportar un negocio ágil. Si bien las organizaciones pueden contar con recursos altamente calificados para proporcionar la configuración manual, en la mayoría de los casos se podría aprovechar mejor el tiempo, invirtiéndolo en otras actividades de valor que resulten más críticas dentro de la empresa y que estén dirigidas a los objetivos del negocio.

Los entornos operativos modernos, comúnmente, se basan en la automatización completa para eliminar la intervención manual, esto incluye: todo el software, la configuración del equipo, parches del sistema operativo, resolución de problemas o corrección de errores o *bugs*. Muchos niveles de prácticas de automatización se pueden utilizar juntos para proporcionar un proceso automatizado de alto nivel de extremo a extremo.

Como puedes imaginar, la automatización tiene muchos beneficios. Entre ellos encontramos:

- ▶ Posibilidad de generar cambios con rapidez.
- ▶ Mejora de la productividad.
- ▶ Configuraciones repetibles.
- ▶ Entornos reproducibles.
- ▶ Elasticidad.
- ▶ Escalabilidad.
- ▶ Pruebas automatizadas.

## AWS Elastic Beanstalk

AWS Elastic Beanstalk es un ejemplo de automatización en AWS. Este **es un servicio que permite a los desarrolladores desplegar aplicaciones en pilas o stacks de tecnología de uso común**. Su (bien lograda) interfaz ayuda a los desarrolladores a desplegar aplicaciones con múltiples capas de forma rápida y sencilla.

AWS Elastic Beanstalk es compatible con la automatización y numerosas buenas prácticas de DevOps, incluyendo el despliegue automatizado de aplicaciones, la monitorización, la configuración de la infraestructura y la gestión de versiones. Los cambios en las aplicaciones y la infraestructura se pueden gestionar fácilmente, yendo hacia adelante o hacia atrás, según sea necesario.

Elastic Beanstalk es también un buen ejemplo de la automatización en la creación de entornos: solo hay que especificar los detalles del entorno y este se encarga de todo el trabajo de configuración y aprovisionamiento. Por ejemplo, aquí hay solo algunas de las opciones que se pueden especificar en el asistente de creación de la aplicación:

- ▶ Definir si quieres un servidor web Tier (que contiene un servidor web y un servidor aplicación) o un nivel Tier (que utiliza Amazon Simple Queue Service).
- ▶ Definir qué plataforma utilizar para la aplicación. Las opciones incluyen IIS, Node.js, PHP, Python, Ruby, Tomcat, Java, Go, .Net, Docker, por ejemplo.
- ▶ Definir si vas a lanzar una sola instancia o crearás un entorno auto escalable de carga balanceada.
- ▶ Definir qué URL asignarás automáticamente al entorno.
- ▶ Definir si el entorno incluirá una instancia Amazon Relational Database.
- ▶ Definir si crearás el entorno dentro de Amazon Virtual Private Cloud.
- ▶ Definir qué URL utilizarás para los controles de salud (automáticas) de la aplicación.
- ▶ Definir qué etiquetas (si las hay) utilizarás para identificar el entorno.

**AWS Elastic Beanstalk** también utiliza la automatización para desplegar aplicaciones. Dependiendo de la plataforma, todo lo que se necesita hacer para desplegar aplicaciones es cargar paquetes en forma de archivos WAR o ZIP directamente desde un ordenador o desde Amazon S3. A medida que se está creando el entorno, AWS Elastic Beanstalk registra automáticamente los eventos en la consola de gestión que proporciona retroalimentación sobre el progreso y el estado del lanzamiento. Una vez completado, se puede acceder a la aplicación mediante la URL definida.

## AWS OpsWorks

AWS OpsWorks lleva los principios DevOps incluso aún más lejos que AWS Elastic Beanstalk, ya que proporciona **más niveles de automatización con funciones adicionales como la integración con el software de gestión de la configuración (Chef) y la administración del ciclo de vida de aplicaciones**. Permite, entonces,

definir cuándo configurar los recursos, desplegarlos, replegarlos o terminarlos.

Para aumentar la flexibilidad, AWS OpsWorks permite definir su aplicación en pilas o *stacks* configurables, pero también se pueden seleccionar pilas de aplicaciones predefinidas. Estas pilas contienen todo el aprovisionamiento de recursos de AWS que requiere la aplicación, incluyendo servidores de aplicación, servidores web, bases de datos y los equilibradores de carga.

Los *stacks* de aplicaciones se organizan en capas o niveles arquitectónicos con el propósito de que las pilas se puedan mantener de forma independiente. Las capas o niveles podrían ser de web, de aplicación y de bases de datos. A su vez, AWS OpsWorks también simplifica la creación de grupos de Auto Scaling y equilibradores de carga, lo que ilustra, además, el principio DevOps de automatización. Al igual que AWS Elastic Beanstalk, es compatible con la generación de versiones de aplicaciones (*versioning*), el despliegue continuo y la gestión de la configuración de la infraestructura.

AWS OpsWorks también es compatible con las prácticas DevOps de supervisión y registro (cubierto en la siguiente sección). El soporte de la supervisión está proporcionado por Amazon CloudWatch. Todos los eventos del ciclo de vida se registran y, en un registro separado, los documentos de Chef, todas las «recetas» de Chef que se ejecutan, sin excepción.

# OpsWorks

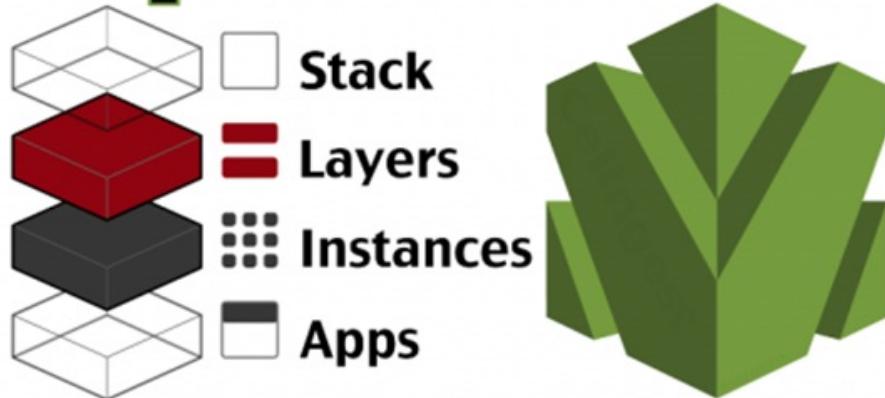


Figura 14. AWS OpsWorks. Fuente: Trager, 2017.

## 7.6. Seguridad

En un entorno DevOps, el foco en la seguridad es un aspecto de suma importancia. La infraestructura y los activos de la empresa deben ser protegidos y, cuando surjan problemas, estos deben ser resueltos de forma rápida y eficaz.

### Gestión de Identidad y Acceso (IAM)

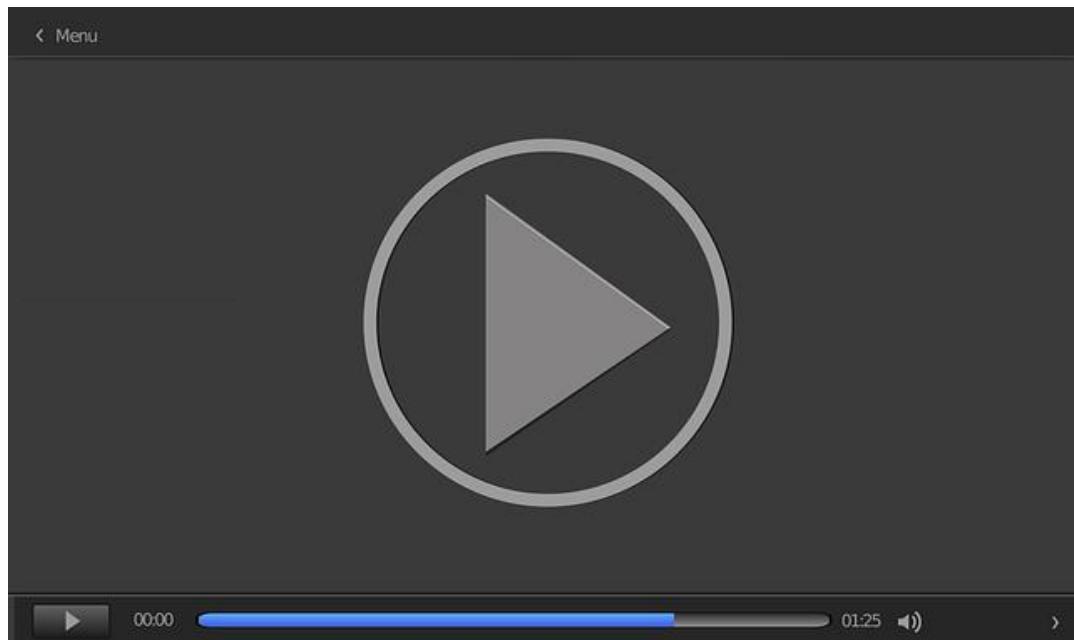
El servicio AWS IAM (gestión de identidad y acceso) es un componente de la infraestructura de seguridad de AWS. Con IAM, se puede administrar de forma centralizada a los usuarios y las credenciales de seguridad, tales como: contraseñas, claves de acceso y políticas de permisos que controlan a qué servicios de AWS y a qué recursos pueden acceder los usuarios. También se puede utilizar IAM para crear roles que se utilicen comúnmente dentro de una estrategia DevOps. Con un rol IAM puedes definir un conjunto de permisos para acceder a los recursos de un usuario o servicio según las necesidades.

Pero en lugar de endosar permisos a un usuario o a un grupo específico, se pueden asociar a una determinada función o rol. Los recursos pueden estar asociados con los roles y los servicios pueden ser, entonces, definidos mediante programación para un determinado papel o rol dentro de la organización. Los requisitos de seguridad y los

controles deben ser agregados durante el proceso de automatización y hay que tener especial cuidado cuando se trabaja con contraseñas y claves.

La colaboración es un elemento clave en DevOps, pero no solo entre desarrolladores y trabajadores del área de operaciones, sino que también hablamos de colaboración entre los propios desarrolladores. Hace ya un tiempo que estos cuentan con herramientas como los repositorios de código y el control de versiones para materializar este verdadero trabajo coordinado que llevan a cabo en las

organizaciones que apuestan por la agilidad y la filosofía DevOps. En el vídeo *Repositorios y control de visiones* se tratarán varios de estos tópicos.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=c2296b60-33d8-4206-8e88-ad78013d2b44>

---

## 7.7. Referencias bibliográficas

Abadala, R. (2017, marzo 23). *Implementing DevSecOps Using AWS CodePipeline*.

Amazon Web Services. <https://aws.amazon.com/es/blogs/devops/implementing-devsecops-using-aws-codepipeline/>

Amazon Web Services. (s.f.). *Productos en la nube*.  
[https://aws.amazon.com/es/products/?pg=WIAWS-N&tile=learn\\_more](https://aws.amazon.com/es/products/?pg=WIAWS-N&tile=learn_more)

Amazon Web Services. (s.f.). *Tipos de informática en la nube*.  
<https://aws.amazon.com/es/types-of-cloud-computing/>

Amazon Web Services. (s.f.). *Infraestructura global*.  
<https://aws.amazon.com/es/about-aws/global-infrastructure/>

Amazon Web Services. (s.f.). *2020 Magic Quadrant for Cloud Infrastructure & Platform Services*. <https://pages.awscloud.com/Gartner-Magic-Quadrant-for-Infrastructure-as-a-Service-Worldwide/?pg=WIAWS-mp>

Golding, T. (2018, marzo 21). Enabling New SaaS Strategies with AWS PrivateLink. *AWS Partner Network (APN) Blog*. <https://aws.amazon.com/es/blogs/apn/enabling-new-saas-strategies-with-aws-prvatealink/>

Martins, F. (2020, enero 17). Calculadora Cloud: Quanto a minha TI poderá custar na nuvem? *Inova Globalweb*. <https://inova.globalweb.com.br/post/quanto-a-minha-ti-podera-custar-na-nuvem>

Trager, D. (2017, junio 23). AWS OpsWorks: fallacies and pitfalls. *Flat Stack*. <https://medium.flatstack.com/aws-opsworks-fallacies-and-pitfalls-f97944bf3e81>

SupportPRO. (s.f.). *AWS CodeDeploy*. <https://www.supportpro.com/blog/aws-codedeploy/>

Valence, P. (2018, septiembre 10). Mainframe Modernization Platform-as-a-Service (ModPaaS) from Modern Systems. *AWS Partner Network (APN) Blog*.  
<https://aws.amazon.com/es/blogs/apn/mainframe-modernization-platform-as-a-service-with-modern-systems-modpaas/>

## Documentación oficial AWS

Amazon Web Services. (s.f.) *AWS Documentation.*

[https://docs.aws.amazon.com/index.html?nc2=h\\_ql\\_doc\\_do](https://docs.aws.amazon.com/index.html?nc2=h_ql_doc_do)

AWS proporciona la documentación específica de todos sus productos y servicios junto con documentación de los SDK de los diferentes lenguajes de programación, recursos generales, diferentes tutoriales y proyectos.

## Comparativa y crecimiento de AWS frente a sus competidores

McAfee. (2019, octubre 25). *Cloud Market in 2019 and Predictions for 2022.*  
<https://www.mcafee.com/blogs/enterprise/cloud-security/microsoft-azure-closes-iaas-adoption-gap-with-amazon-aws/>

En este enlace podemos ver la cuota de mercado de AWS frente a sus competidores y la predicción de crecimiento en los próximos años, donde vemos que AWS tiene una expectativa de crecimiento grande y gran cuota de mercado.

## SAP on AWS

Amazon Web Services. (s.f.). *SAP on AWS*. <https://aws.amazon.com/es/sap/>

AWS proporciona la posibilidad de implementar todas las suites disponibles de SAP bajo su ecosistema. Dos grandes se unen para dar un valor añadido a las empresas que están pensando en migrar sus sistemas a la nube y todo ello certificado por el mayor fabricante de software empresarial a nivel mundial.

## Desplegando y desarrollando aplicaciones modernas en la nube

Amazon Web Services LATAM. (2019, noviembre 21). *AWS Cloud Experience CA: Desplegando y Desarrollando Aplicaciones Modernas en la Nube*. SlideShare.  
<https://www.slideshare.net/AmazonWebServicesLATAM/aws-cloud-experience-ca-desplegando-y-desarrollando-aplicaciones-modernas-en-la-nube>

Sesión magistral del AWS Cloud Experience del 2019.

**1.** ¿Cuáles son los modelos de servicios e implementación que proporciona AWS?

- A. IaaS, VaaS, SaaS (nube y local).
- B. Iaas, PaaS, SaaS (nube, híbrida y local).
- C. IaaS, PaaS, MaaS (local e híbrida).
- D. IaaS, PaaS, SaaS (local, híbrida y nube).

**2.** ¿Cuántos grupos de servicios dispone AWS en la actualidad?

- A. 175.
- B. 25.
- C. 300.
- D. 17.

**3.** Relaciona grupo y servicio de AWS

Amazon QM	1	A	Informática
AWS Lamdba	2	B	Integración Apps.
Familia de productos AWS Snow	3	C	Migración y Transferencia
Amazon S3	4	D	Almacenamiento

**4.** ¿En caso de superar el cómputo de la capa gratuita, es necesario pagar?

- A. No, porque son gratuitos todos los servicios a utilizar.
- B. Si, porque es necesario pagar para crearte una cuenta de AWS.
- C. Solo hay que pagar en el caso de que superes el cómputo en algunos de los servicios.
- D. Todas son falsas.

5. Señala el beneficio o los beneficios de la utilización de AWS PipeLine:
  - A. Automatización de entregas.
  - B. Fácil integración.
  - C. Empobrece la calidad del producto.
  - D. Solo compatible con productos de terceros.
6. ¿En qué consiste el Despliegue Blue Green?
  - A. Despliegue de DevOps que utiliza los servicios de nombres de dominio (en siglas, DNS) para realizar el despliegue de aplicaciones.
  - B. Despliegue de soluciones empresariales para desplegar en AWS.
  - C. Es un código de colores que identifica el estado en el que se encuentra nuestra aplicación.
  - D. Despliegue de un servicio directamente en producción.
7. ¿AWS OpsWorks permite la creación y administración de servidores Chef?
  - A. Sí.
  - B. No.
  - C. No porque es compatible.
  - D. Sí, pero en la región de Estados Unidos únicamente.
8. ¿Qué tipos de aplicaciones son compatibles con AWS Elastic Beanstalk?
  - A. Net, PHP, Python, Ruby, Docker, Java, ABAP for Hana.
  - B. ABAP, Hana, Python,.Net, PHP.
  - C. Java.
  - D..Net, PHP, Python, Ruby, Docker, Java, Node.js, Go.

**9.** ¿El pago de AWS es mediante el método de suscripción, pagando un coste fijo todos los meses?

- A. Sí, en AWS pagas una suscripción mensual.
- B. No, pagas una suscripción mensual fija más el computo utilizado.
- C. No, la modalidad de pago es un pago por uso.
- D. AWS es totalmente gratuita y el coste es el del proveedor de Internet.

**10.** ¿Por qué es importante AWS IAM?

- A. Porque nos muestra el cómputo que llevamos utilizado en el último mes.
- B. Porque sin AWS no tenemos acceso a la red.
- C. Porque con AWS IAM gestionamos usuarios, grupos, autorizaciones y accesos.
- D. Ninguna de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 8. Infraestructura como código

# Índice

[Esquema](#)

[Ideas clave](#)

[8.1. Introducción y objetivos](#)

[8.2. ¿Qué es la IaC?](#)

[8.3. Ciclo de vida de los recursos de infraestructura](#)

[8.4. Tipos de arquitecturas en Cloud](#)

[8.5. Herramientas para IaC](#)

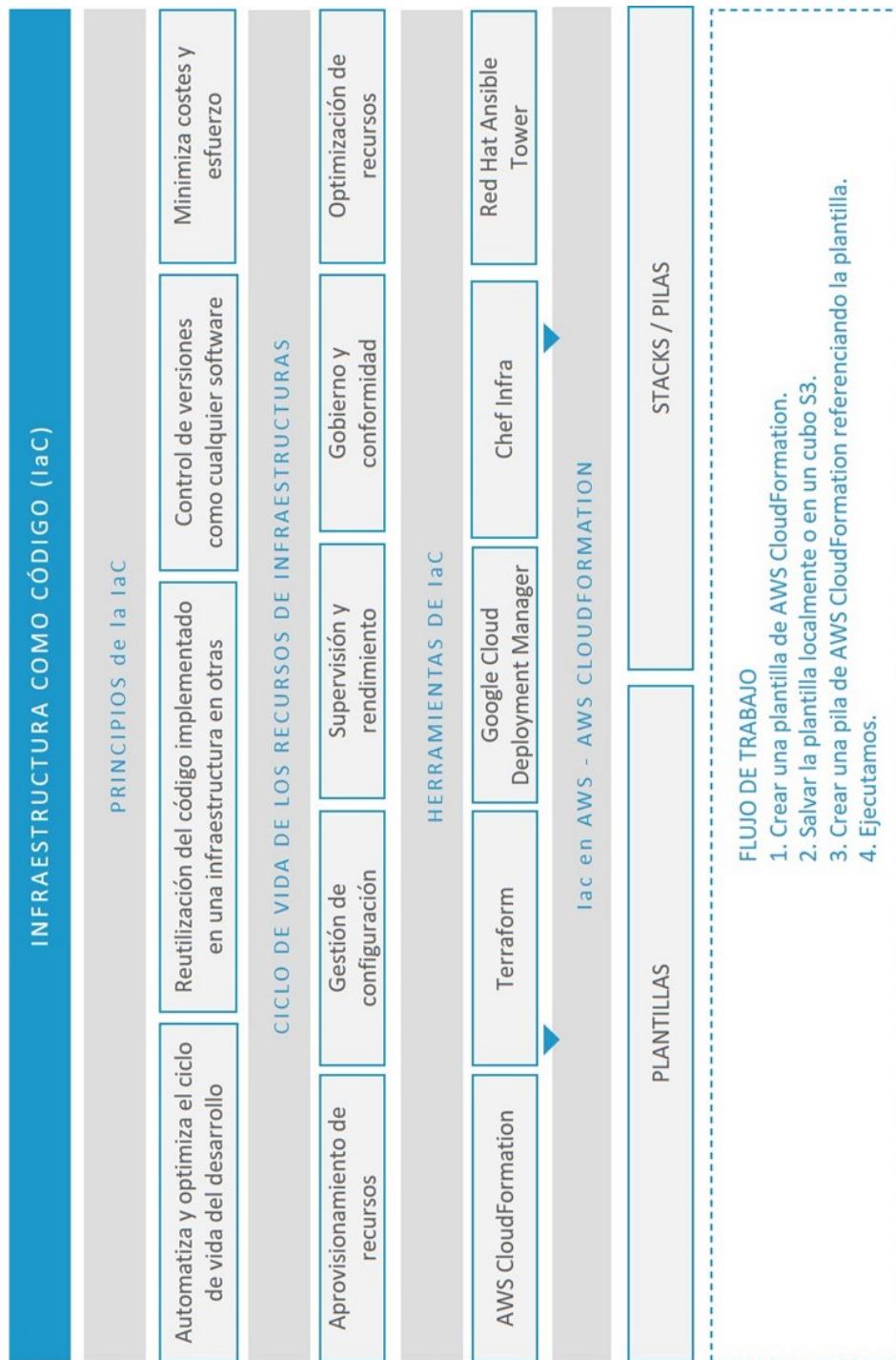
[8.6. IaC en AWS](#)

[8.7. Referencias bibliográficas](#)

[A fondo](#)

[Definición de IaC](#)

[Test](#)



## 8.1. Introducción y objetivos

Hasta hace unos años, los responsables de sistemas tenían que montar sus infraestructuras de hardware y software de forma manual. Es decir, que montaban y apilaban el hardware para posteriormente instalar y configurar los sistemas operativos y aplicaciones necesarias para el negocio. Como puedes imaginar, este proceso, además de ser muy tedioso, normalmente se ejecutaba por más de una persona, lo que generaba una larga espera hasta que la infraestructura de aplicaciones y sistemas estuviese disponible. Para agilizar estas implementaciones, optimizar el rendimiento y entregar con más rapidez el software desplegado en las plataformas, los responsables de sistemas crearon scripts (pequeñas porciones de código a bajo nivel) para unificar, automatizar y optimizar el proceso.

A su vez, la aparición de las metodologías ágiles ha instaurado una nueva forma de gestionar el software que implica que la instalación, configuración y mantenimiento de los sistemas informáticos se haga más compleja, si es posible. Estas metodologías requieren de varios ambientes donde ejecutar las aplicaciones (desarrollo, *testing*, *staging*, etc.), lo cual termina duplicando, al menos, el esfuerzo de los administradores.

La respuesta que ha dado la informática a estos problemas es la optimización y automatización del despliegue de aplicaciones y la configuración de los servidores. Actualmente, es posible configurar servidores a través de la programación y sin intervención de los administradores. Esta nueva forma de administración, que considera a la infraestructura como un aplicativo más a ser gestionado de manera análoga al software, se la conoce **como infraestructura programable o infraestructura como código**, más conocida como IaC (Huttermann, 2012).

En este tema se pretenden conseguir los siguientes objetivos:

- ▶ Conocer qué es la Infraestructura como código.
- ▶ Analizar el ciclo de vida de los recursos de infraestructura.
- ▶ Conocer cuáles son las herramientas de la Infraestructura como código.
- ▶ Analizar cómo se implementa la infraestructura como código en AWS.

## 8.2. ¿Qué es la IaC?

En 2011, Andressen Marc escribió un artículo llamado «Why Software is Eating the World». La idea central de este artículo está en que cualquier proceso que puede ser programado, se convierte en software. Esto se ha convertido en una especie de abreviatura para las tesis de inversión detrás de la actual ola de *startups* unicornio de Silicon Valley. También es una idea unificadora detrás del conjunto más amplio de tendencias tecnológicas que vemos hoy en día como *machine learning*, IoT, *ubiquitous mobile connectivity*, SaaS, y Cloud Computing. Estas tendencias están haciendo que el software sea más abundante y capaz y están ampliando su alcance dentro de las arquitecturas de sistemas.

La IaC ofrece las siguientes ventajas frente a la configuración manual:

IaC Vs Op. Manual			
IaC		Op. Manual	
1	<b>Alta eficiencia:</b> automatiza la mayor parte de la administración de los recursos, que conlleva a optimizar el ciclo de vida del desarrollo del software.	1	<b>Costos altos:</b> la necesidad de capital humano puede ir hacia otras necesidades del negocio más importantes.
2	<b>Reutilización:</b> una vez se ha escrito el código para una infraestructura, este se puede ejecutar en cualquier momento y todas las veces que se desee.	2	<b>Inconsistencias:</b> debido al error humano, conducen a desviaciones del estándar de configuración.
3	<b>Control de versiones:</b> donde haya código, también es posible controlar las versiones.	3	<b>Falta de agilidad:</b> se limita limitar la velocidad a la que la organización puede lanzar nuevas versiones de servicios en respuesta a las necesidades e indicadores de mercado.
4	<b>Minimizar costes/esfuerzo:</b> automatizar la administración de la infraestructura ahorra muchos costes y horas de trabajo.	4	<b>Dificultad:</b> en alcanzar y mantener conformidad a la corporación o estándares de la industria debido a la falta de procesos repetibles.

Tabla 1. Enfoques de la IaC. Fuente: elaboración propia.

La infraestructura como código (en siglas, IaC) trata la infraestructura de configuración de sistemas como un software de programación. Esto genera una delgada línea entre los límites de la escritura de aplicaciones y la creación de entornos en los que se ejecutan. Se trata de una parte fundamental de la computación en la nube y esencial para DevOps. La IaC es el marco que ha dado origen a DevOps.



Figura 1. Enfoques de la IaC. Fuente: elaboración propia.

## 8.3. Ciclo de vida de los recursos de infraestructura

Las etapas del ciclo de vida de los recursos son las siguientes:

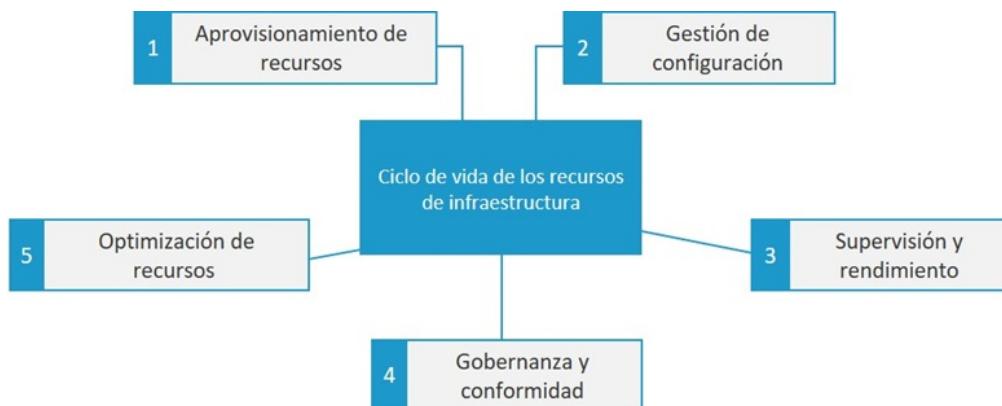


Figura 2. Ciclo de vida de los recursos de infraestructura. Fuente: elaboración propia.

Cada etapa involucra procedimientos que pueden aprovecharse en código, lo cual extiende los beneficios de la IaC de su rol tradicional de aprovisionar al ciclo de vida entero. Cada ciclo de vida se beneficia de la consistencia y repetitividad que la infraestructura como código ofrece.

- ▶ **Aprovisionamiento de recursos.** Etapa en la que los administradores utilizan este principio de la IaC para aprovisionar recursos de acuerdo con sus especificaciones y necesidades. AWS CloudFormation puede ser un claro ejemplo para aplicar.
- ▶ **Gestión de configuración.** Los recursos se vuelven componentes de un sistema de gestión de configuración que soportan actividades tales como optimización y actualización.
- ▶ **Supervisión y rendimiento.** Herramientas de supervisión y rendimiento validan el estado operacional de los recursos examinando ítems como métricas, transacciones sintéticas y archivos de registro.
- ▶ **Gobierno y conformidad.** Los marcos de trabajo de conformidad y gobierno

gestionan la validación adicional a fin de asegurar la relación y consonancia con la corporación y los estándares de la industria, así como requerimientos regulatorios.

- ▶ **Optimización de recursos.** Los administradores revisan datos de rendimiento e identifican cambios necesarios para optimizar el ambiente alrededor de los criterios tales como rendimiento y los costes.

## 8.4. Tipos de arquitecturas en Cloud

Con la aparición de la IaC, surgen diferentes tipos de arquitecturas en la nube, con conceptos y enfoques diferentes que solucionan el paradigma de cómo y con qué arquitectura empezar a trabajar. Nos encontramos con tres tipos de infraestructuras en la nube, cada una con un fin específico.

### Infraestructura como Servicio (IaaS)

También conocida como HaaS (del inglés *Hardware as a Service*), esta arquitectura se basa en el modelo de dotar de forma externalizada a sus usuarios/empresas, del hardware necesario. IaaS proporciona hardware, almacenamiento, servidores y espacio de centro de datos o componentes de red. Como inconveniente podemos destacar que se requiere de los mismos conocimientos informáticos en sistemas operativos y redes informáticas que necesitábamos con una arquitectura tradicional.

### Plataforma como Servicio (PaaS)

Abarca el ciclo completo de desarrollo de software para ser puesto en producción. El hardware y servidor donde se aloje es específico del proveedor, y por ello no causa ningún problema al software que alojemos y despleguemos en él. El concepto *serverless* es un PaaS para aplicaciones orientadas a Internet o web, en el que el coste es por llamada a la aplicación. Esto significa que, si no tiene llamadas, el servicio se apaga y queda en espera automáticamente.

### Software como Servicio (SaaS)

Es un modelo de distribución de software donde la aplicación, el servidor y los datos son gestionados por un servidor externo. Se requiere de formación y adaptación para el uso correcto del producto y su pago, normalmente, es por usuario o licencia.

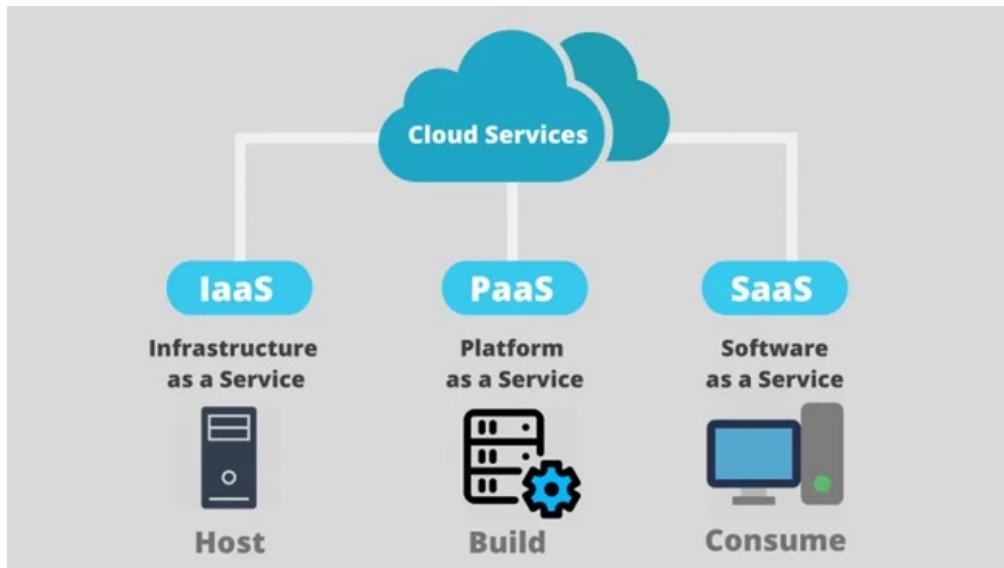


Figura 3. Overview arquitecturas Cloud. Fuente: Preesoft, s.f.

## 8.5. Herramientas para laC

Independientemente de la herramienta que usemos, es necesario trabajar con algún lenguaje para la descripción de la configuración, pudiendo ser Json, Yaml o algún nuevo lenguaje propio del software. Dentro del archivo de configuración es donde definiremos todos los componentes necesarios para crear nuestra infraestructura. Las herramientas y *frameworks* especiales de IaC ofrecen sus propios lenguajes de configuración, que permiten administrar recursos de múltiples proveedores y eliminan la necesidad de conocer las API correspondientes en profundidad.

**Mapa de herramientas IaC**

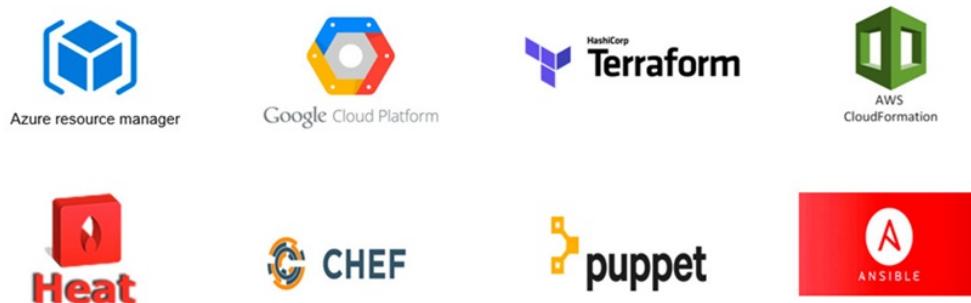


Figura 4. Mapa de herramientas IaC. Fuente: EdynTechnology, s. f.

Las herramientas más utilizadas actualmente son:

- ▶ **Azure Resource Manager:** servicio de Azure, plataforma en la nube de Microsoft, que proporciona la administración de infraestructuras mediante plantillas, de forma que implementa y supervisa todos los recursos. Esto da coherencia a la hora de reimplementar recursos ya existentes y permite definir las dependencias de estos.

- ▶ **Google Cloud Deployment Manager:** Deployment Manager es, para la plataforma Google Cloud, lo que CloudFormation es para AWS. Con esta herramienta gratuita, los usuarios de recursos de IaaS de Google pueden administrarlos fácilmente mediante archivos de configuración central en el lenguaje de marcado Yaml.
- ▶ **Terraform:** la versión básica de Terraform, un software de código libre desarrollado por HashiCorp, está disponible de forma gratuita. Sus dos versiones de pago proporcionan funciones de configuración y organización, además de características para el trabajo grupal.
- ▶ **AWS CloudFormation:** CloudFormation es la herramienta interna de IaC de Amazon Web Services (en siglas, AWS) y, como tal, es prácticamente imprescindible para cualquiera que trabaje con productos de AWS como ELB, S3 o EFS. Utilizarla no conlleva ningún coste adicional, tan solo hay que pagar por los recursos reservados.
- ▶ **Heat:** implementa un motor de orquestación para poder lanzar múltiples aplicaciones en la nube basadas en plantillas, proporcionando una compatibilidad total con las plantillas de AWS CloudFormation. Proporciona, a su vez, una API nativa y una API compatible con AWS CloudFormation. Heat es el proyecto más ambicioso de OpenStack.
- ▶ **Chef Infra:** la solución de IaC de la empresa estadounidense Chef, está disponible desde abril de 2019 bajo la licencia gratuita Apache 2.0 y es utilizado por Facebook, entre otras empresas. Entre las plataformas compatibles se incluyen Google Cloud, Microsoft Azure, Amazon EC2 y OpenStack.
- ▶ **Puppet:** herramienta open source de gestión desarrollada en Ruby para la administración de sistemas de forma declarativa que, al estar basada en modelos, no requiere un alto conocimiento de programación para su uso.

- ▶ **Red Hat Ansible Tower:** la herramienta de infraestructura como código de Ansible forma parte del catálogo de desarrollo de software Red Hat desde el año 2015. Ofrece un panel de control, su propia línea de comandos y una potentísima API Rest. En este caso, ambos paquetes disponibles, tanto el estándar como el extendido, son de pago.

## 8.6. IaC en AWS

Amazon Web Services (en adelante, AWS) proporciona la funcionalidad de IaC a través del servicio de CloudFormation. Este servicio contempla todos los pilares y fundamentos de IaC vistos anteriormente como:

- ▶ **Simplificación y rapidez.** Una aplicación web escalable que también incluye una base de datos de última generación, puede utilizar un grupo de escalado automático, un balanceador de carga elástico y una instancia de base de datos de Amazon Relational Database Service. Todas estas tareas pueden añadir complejidad y tiempo, incluso antes de conseguir que la aplicación funcione. En su lugar, podemos crear o modificar una plantilla de AWS Cloudformation existente. Una plantilla describe todos los recursos y propiedades. Cuando usamos esa plantilla para crear una pila de Cloudformation AWS, AWS Cloudformation provee el grupo de Auto Scaling, el balanceador de carga y la base de datos automáticamente. Después de que la pila se haya creado con éxito, los recursos AWS estarán en funcionamiento. Al utilizar AWS Cloudformation, podemos gestionar fácilmente una colección de recursos como una sola unidad.
- ▶ **Replicación y escalabilidad.** Podemos replicar nuestras aplicaciones en varias regiones de modo que, si una región no está disponible, sus usuarios todavía pueden utilizar la aplicación en otras regiones. Utilizando AWS Cloudformation, podemos reutilizar la plantilla para configurar los recursos de forma consistente y repetida. Una vez creados los recursos, podemos proveer los mismos recursos una y otra vez en múltiples regiones.

- ▶ **Automatización mediante las API.** Una de las características que más flexibilidad proporciona en CloudFormation es la posibilidad de crear recursos personalizados. En el caso de que no sea posible crear un recurso con las opciones o características de forma nativa, mediante los tipos de recursos que proporciona CloudFormation, se pueden crear a través de un tipo de recurso «Custom». Para crear un recurso de este tipo se define primero en la plantilla un recurso Lambda (es donde se programan las llamadas a la API de AWS necesarias para la creación del recurso final), luego se referencia este recurso Lambda desde el recurso «Custom» y se le proporcionan los parámetros que necesite la función Lambda.
- ▶ **Actualizaciones controladas y *rollbacks*.** Se pueden actualizar los recursos subyacentes de forma incremental. Por ejemplo, podemos necesitar un cambio a un tipo de instancia de mayor rendimiento en la configuración de lanzamiento de Auto Scaling para que pueda reducir el número máximo de instancias en su grupo de Auto Scaling. Si se producen problemas después de completar la actualización, es posible que tengamos que hacer *rollbacks* de la infraestructura a la configuración original. Cuando utilizamos AWS CloudFormation, la plantilla de AWS CloudFormation describe exactamente qué recursos están provisionados y su configuración. Debido a que estas plantillas son archivos de texto, solo es necesario rastrear las diferencias entre las plantillas para ver los cambios en la infraestructura, similar a la forma en la que los desarrolladores controlan las revisiones al código fuente. Por ejemplo, puedes usar un sistema de control de versiones con plantillas para saber exactamente qué cambios se hicieron, quién los hizo y cuándo. Si en algún momento necesitáramos revertir cambios en la infraestructura, podemos usar una versión anterior de la plantilla.

AWS ClouFormation utiliza otros servicios de AWS complementarios, como ya hemos explicado en temas anteriores:

- ▶ AWS S3
- ▶ AWS CodeCommit

- ▶ AWS Lambda

AWS Cloudformation está compuesto principalmente por dos pilares:

## Plantillas

Son los ficheros donde se describen las infraestructuras en sí mismas. Nombraremos algunas de las características fundamentales:

- ▶ Son susceptibles de ser anidadas e importadas por otras plantillas.
- ▶ El motor de plantillas ofrece funciones auxiliares similares a las de cualquier lenguaje de programación.
- ▶ Permiten definir parámetros y, a través de estos, se pueden escribir plantillas reutilizables para distintos entornos. Se pueden definir condiciones y utilizarlas para crear un recurso u otro en base a una condición.

## Stacks

Es la implementación de una plantilla combinada con los parámetros necesarios. Una vez creada, permite comprobar y acceder a la información sobre: qué plantilla se ha usado, qué parámetros tenía, cuáles son los recursos creados y la información de salida que proporciona.

AWS CloudFormation es un servicio para modelar y configurar los recursos de AWS de forma sencilla, mediante la creación de una plantilla que describa todos los recursos de AWS que se necesiten (como las instancias de Amazon EC2 o DB de Amazon RDS), y AWS CloudFormation se encarga de aprovisionar y configurar esos recursos automáticamente.



Figura 5. Imagen de AWS CloudFormation. Fuente: Amazon Web Services, s. f.

## Funcionamiento de AWS CloudFormation

Al crear una pila o *stack*, AWS CloudFormation realiza llamadas de servicio subyacentes a AWS para proporcionar y configurar sus recursos. AWS CloudFormation solo puede realizar acciones para las que tiene permiso, es decir, para crear instancias EC2 usando AWS CloudFormation, se necesitan permisos para realizar dicha acción. La administración de permisos se realiza a través de AWS Identity and Access Management (en siglas, IAM).

Las llamadas que hace AWS CloudFormation son declaradas en su totalidad en la plantilla. Por ejemplo, supongamos que tenemos una plantilla que describe una instancia EC2 con un tipo de instancia t1.micro. Cuando se utiliza esa plantilla para crear una pila, AWS CloudFormation llama a Amazon EC2 creando una API instancia y especifica el tipo de instancia como t1.micro. El siguiente diagrama resume el flujo de trabajo de AWS CloudFormation para crear pilas.



Figura 6. Flujo de trabajo en CloudFormation. Fuente: Amazon Web Services, s. f.

- ▶ Crea una plantilla de AWS CloudFormation (un documento con formato JSON o YAML) en AWS CloudFormation Designer o escribe una en un editor de texto. También podemos elegir una plantilla proporcionada por AWS CloudFormation. La plantilla describe los recursos que queremos y su configuración.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A simple EC2 instance",
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "ami-0ff8a91507f77f867",
        "InstanceType" : "t1.micro"
      }
    }
  }
}
```

Figura 7. Ejemplo en JSON usando plantillas para crear una instancia EC2. Fuente: Amazon Web Services, s.f.

- ▶ Guarda la plantilla localmente o en Amazon S3. Si es una creación desde cero, se guardará con cualquiera de las siguientes extensiones: .json, .yaml, o .txt.
- ▶ Crea una pila de AWS CloudFormation especificando la ubicación de la plantilla como ruta local o URL de Amazon S3. Si la plantilla contiene parámetros, es posible especificar valores de entrada en el momento de crear la pila. Los parámetros permiten pasar valores a la plantilla para personalizar los recursos cada vez que cree una pila. Adicionalmente, es posible crear pilas usando AWS CloudFormation, API o AWS CLI.

Una vez que los recursos hayan sido creados, AWS CloudFormation informa que la pila ha sido creada. Posteriormente, será posible empezar a utilizar los recursos de la pila creada. Si en el momento de la creación hubiese algún problema, AWS CloudFormation realizaría un *rollback* y eliminaría los recursos que se hayan creado.

## Best practices en AWS CloudFormation

Las *best practices* o buenas prácticas de AWS CloudFormation son recomendaciones para tener en cuenta a la hora utilizar dicha plataforma. Estas, sin dudas, fomentan el uso de la plataforma de una forma más eficaz, eficiente y segura.

Veamos cuáles son:

- ▶ Planificación y organización
  - Utilizar IAM para controlar el acceso.
  - Organizar las pilas por ciclo de vida y titularidad.
  - Utilizar referencias de pilas cruzadas para exportar recursos compartidos.
  - Reutilizar plantillas para replicar pilas en otros entornos.
  - Verificar las cuotas de los tipos de recursos.
  - Utilizar pilas anidadas para reutilizar patrones de plantillas comunes.

- ▶ Creación de plantillas
  - Utilizar AWS::CloudFormation::Init para implementar aplicaciones de software en las instancias de Amazon EC2.
  - Utilizar tipos de parámetros específicos y sus limitaciones de AWS.
  - No integrar credenciales en las plantillas.
  - Utilizar los scripts más recientes y validar las plantillas antes de usarlas.
- ▶ Gestión de pilas
  - Administrar todos los recursos de las pilas a través de AWS CloudFormation.
  - Utilizar las políticas de pilas de AWS.
  - Crear conjuntos de cambios antes de actualizar.
  - Actualizar con regularidad las instancias Linux de Amazon EC2.
  - Utilizar AWS CloudTrail para registrar las llamadas.

## Ejemplo de acceso a AWS y la *template* de AWS CloudFormation

A continuación, veremos más a fondo un ejemplo de plantillas disponible dentro de AWS CloudFormation. Se debe tener en cuenta que, dependiendo de la región en la que estén nuestros servicios vinculados, tendremos unas plantillas genéricas y otras específicas de la región.

Para poder acceder a AWS CloudFormation, lo primero que debemos hacer es acceder a nuestra cuenta raíz de AWS y configurarla para crear un subdominio y un usuario del subdominio que será con el que se trabajará en adelante. Es recomendable que la cuenta *root* no se utilice para otro propósito que no sea el de la administración.

Para crear un subdominio y un usuario, debemos seguir los siguientes pasos:

- ▶ Accederemos a nuestra cuenta *root* y seleccionaremos el servicio IAM para configurar el subdominio y el usuario real con el que trabajaremos:

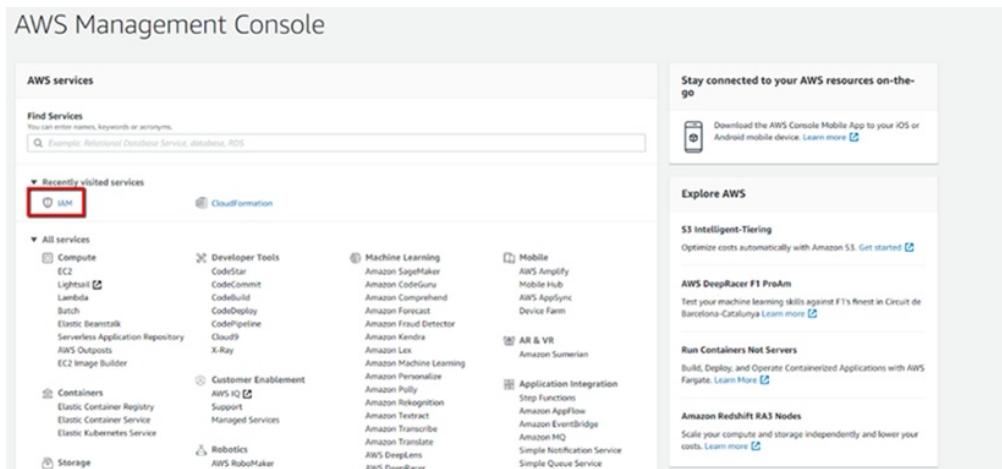


Figura 8. Consola de administración de AWS. Fuente: elaboración propia.

- ▶ Dentro del servicio IAM crearemos de un grupo de administración y un usuario; y configuraremos las políticas de seguridad y acceso. Una vez hecho esto, nuestra consola de IAM debe quedar de la siguiente forma:

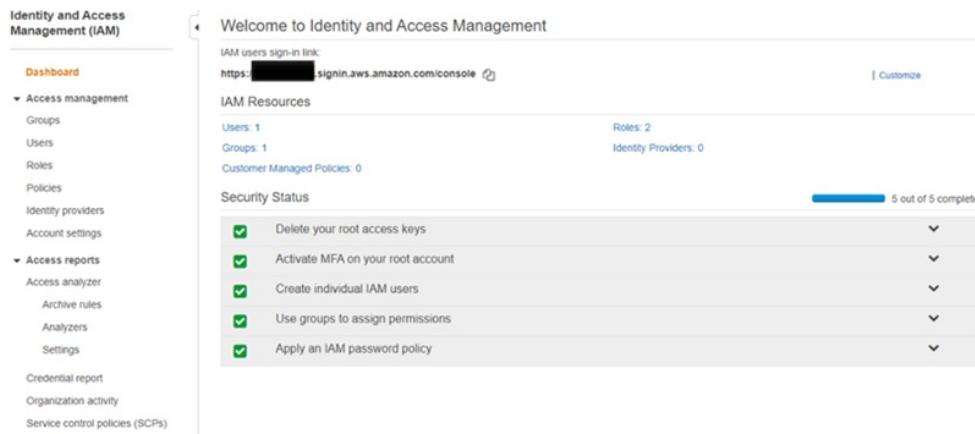


Figura 9. Consola de administración de IAM. Fuente: elaboración propia.

- ▶ Posteriormente, debemos copiar la URL que nos ha generado para acceder con nuestro nuevo usuario. Cerraremos la sesión root y accederemos a nuestra URL, que debe de ser así:
  - <https://xxxxxxxxx.signin.aws.amazon.com/console>
- ▶ Ahora, accederemos a la siguiente página:

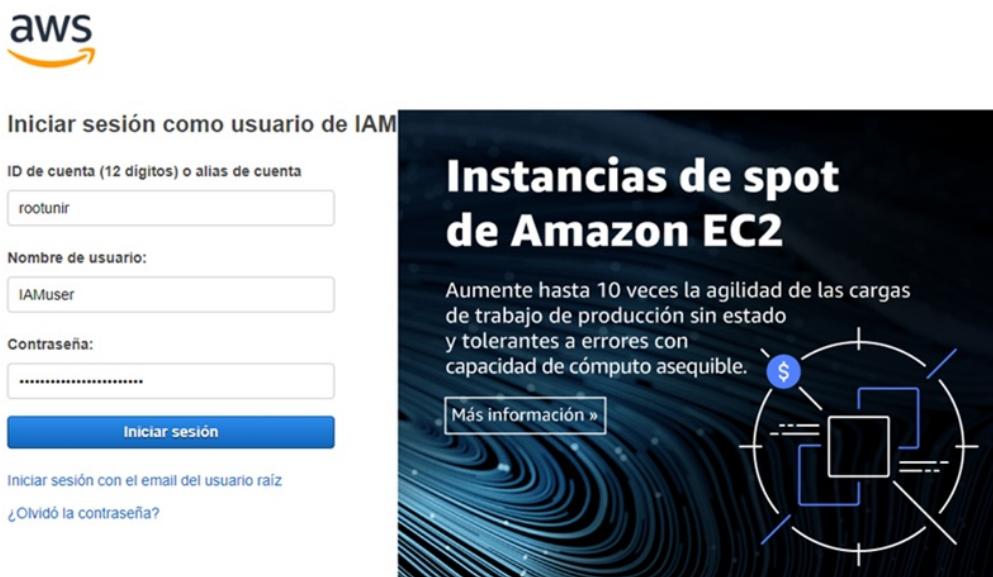


Figura 10. Acceso a consola AWS mediante usuario IAM. Fuente: elaboración propia.

- ▶ Desde ahora, siempre trabajaremos con nuestro usuario IAM por motivos de seguridad.
- ▶ Seleccionaremos dentro de la consola de administración el servicio AWS CloudFormation y accederemos a él:

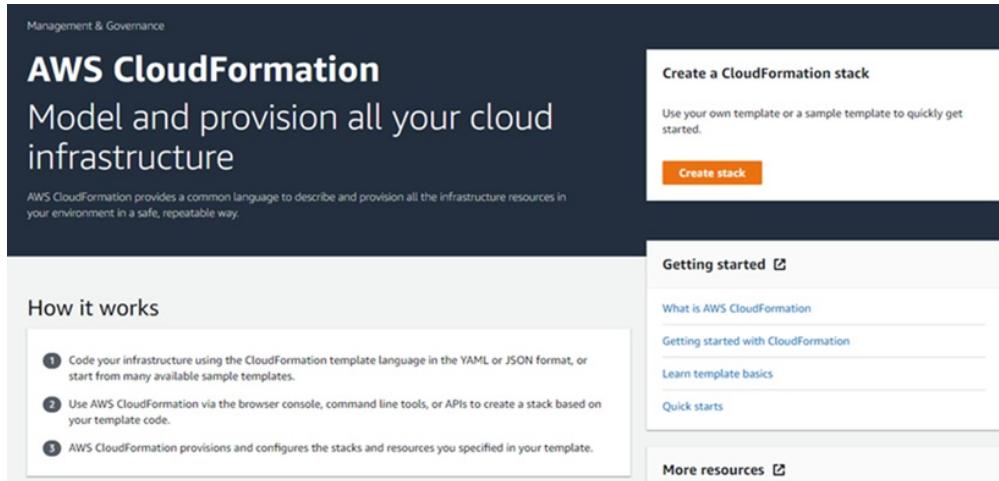


Figura 11. consola AWS CloudFormation. Fuente: elaboración propia.

Como ya se ha comentado anteriormente, las plantillas disponibles en AWS CloudFormation dependerán de la zona en la que nos encontremos, como se puede ver a continuación:

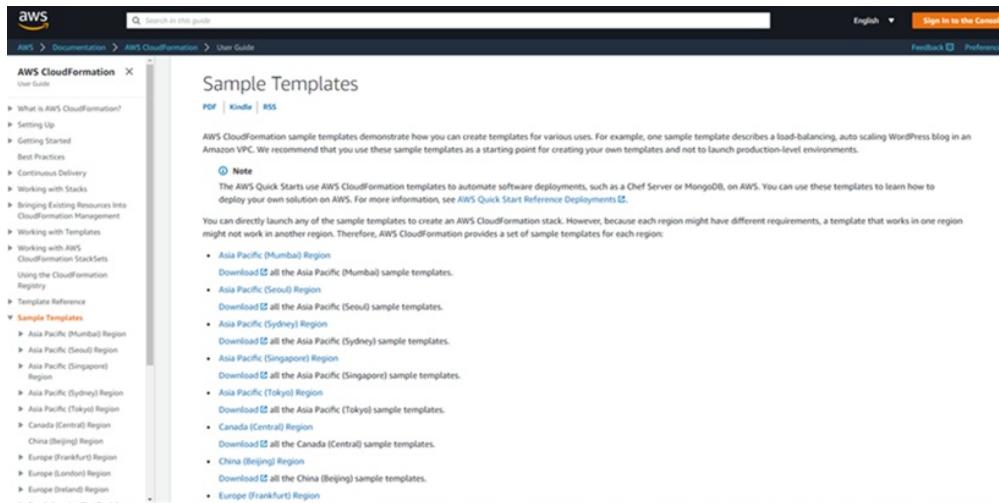


Figura 12. Plantillas ejemplo AWS CloudFormation por Zona Geográfica. Fuente: elaboración propia.

En nuestro caso, revisaremos una plantilla de la región de **Europa (Frankfurt)**. AWS creará una instancia individual básica de Wordpress con una **base de datos MySQL** local para el almacenamiento y una **instancia Amazon EC2**. Tenemos la posibilidad

# Ideas clave

de ver dicha plantilla en formato archivo, como se muestra en la siguiente imagen:

Figura 13. Fichero de configuración de una plantilla con AWS CloudFormation. Fuente: Amazonaws, s.f.

- ▶ Llegados a este punto, deberemos copiar la plantilla y configurarla según las necesidades o realizar la configuración completa mediante AWS CloudFormation Designer. Esta herramienta nos permitirá proceder con la configuración de una forma visual y sencilla mediante el *drag&drop* (arrastrar y soltar) de los elementos que queremos incorporar dentro de la interfaz.

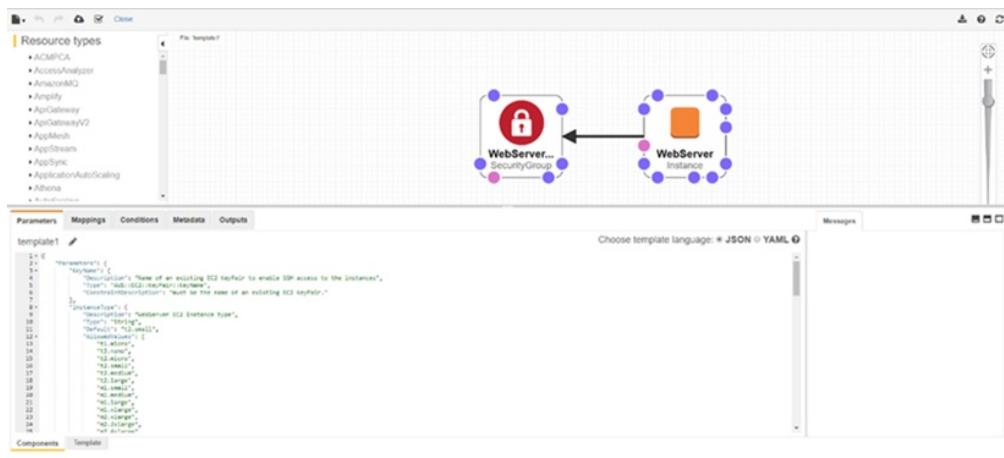


Figura 14. AWS CloudFormation Designer de AWS CloudFormation. Fuente: elaboración propia.

AWS CloudFormation Designer está compuesta por las siguientes áreas:

- ▶ **Diseño visual de la plantilla.** Aquí será posible realizar el diseño de nuestra solución a implementar. En el lado izquierdo de la página dispondremos de todos los recursos disponibles.

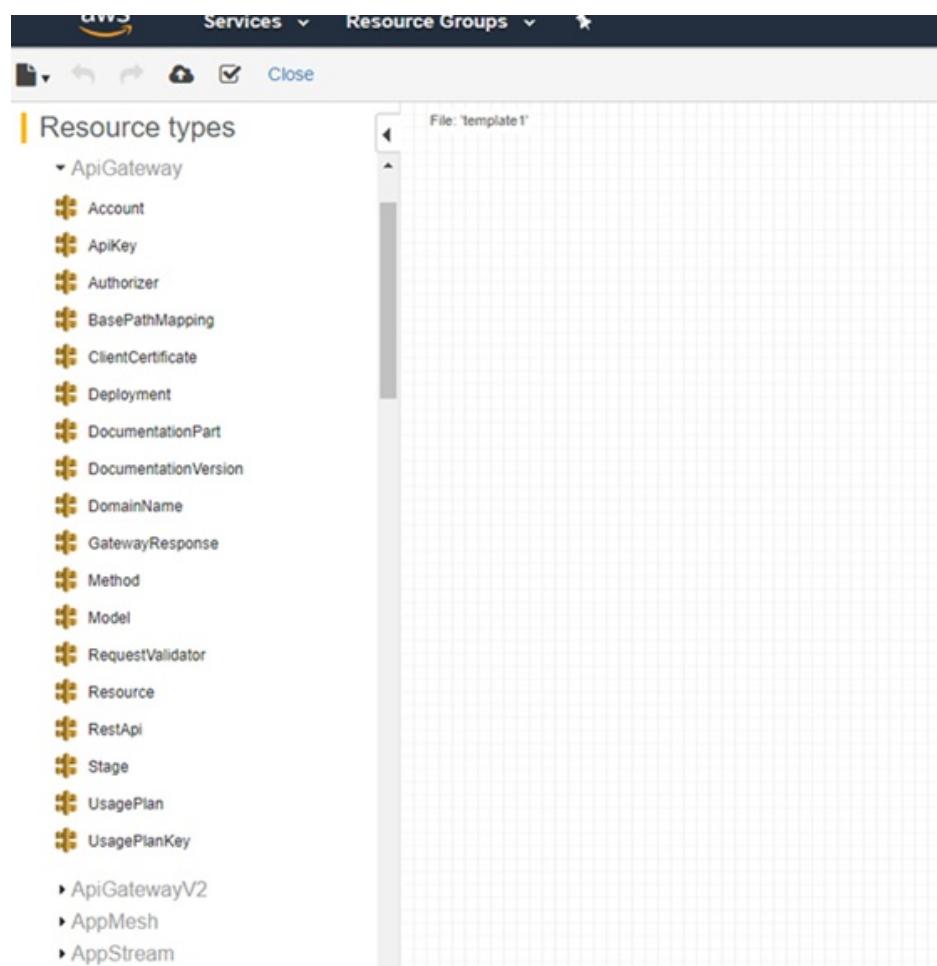


Figura 15. AWS CloudFormation Designer Selección de Recursos. Fuente: elaboración propia.

En la parte derecha de la interfaz haremos *drag&drop* para seleccionar los recursos que queremos desplegar, junto con la comunicación entre ellos y la configuración de seguridad, accesos, etc. La plantilla debería de quedar de esta forma:

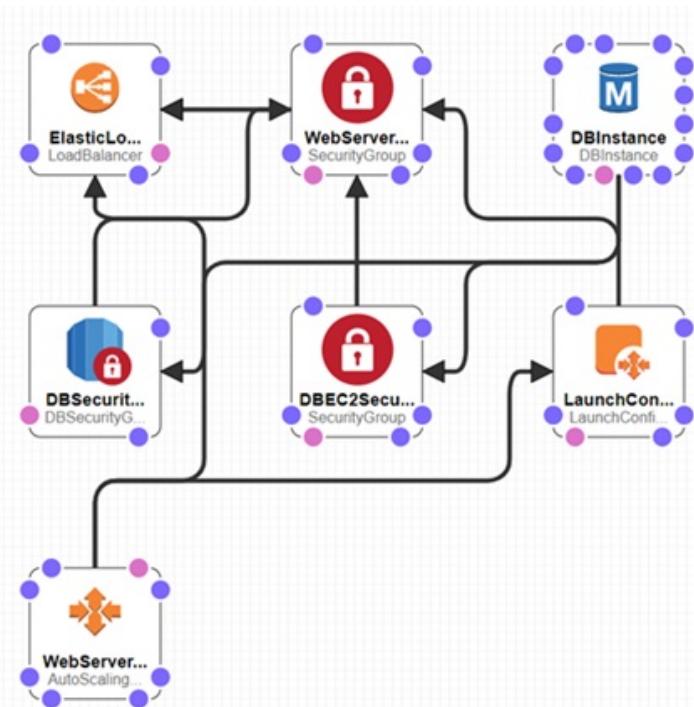


Figura 16. AWS CloudFormation Designer Visual Template. Fuente: elaboración propia.

- ▶ Configuración de parámetros, mapeos, condiciones, metadatos y salidas. Aquí es posible desarrollar los ficheros de configuración de forma manual, pudiendo elegir JSON o YAML como lenguaje.

The screenshot shows the AWS CloudFormation Designer interface. On the left, there's a sidebar titled 'Resource types' listing various AWS services. The main area displays the visual template of the architecture shown in Figure 16. Below the visual template, there's a red bar containing tabs for 'Parameters', 'Mappings', 'Conditions', 'Metadata', and 'Outputs'. The 'Parameters' tab is selected, showing a JSON-based configuration block. The configuration includes parameters for the Load Balancer, Web Server, DB Instance, and Auto Scaling group, along with their respective descriptions and default values. To the right of the visual template, there's a message 'Designer is out of date, hit refresh'. At the bottom right, it says 'Choose template language: \* JSON \* YAML'. The bottom navigation bar has tabs for 'Components' and 'Template', with 'Template' being the active tab.

```

template1
1: {
  "Parameters": {
    "keyName": {
      "Description": "Name of an existing EC2 key pair to enable SSH access to the instances",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "Must be the name of an existing EC2 key pair."
    }
  },
  "Resources": {
    "WebServer": {
      "Type": "AWS::EC2::WebServer"
    }
  }
}
  
```

Figura 17. AWS CloudFormation Configuración total Template. Fuente: elaboración propia.

Una vez terminada la configuración, lo primero que debemos hacer es **comprobar si la plantilla es válida** antes de realizar el despliegue. Para ello, debemos marcar la casilla como en la imagen que aparece debajo:



Figura 18. AWS CloudFormation validación Template. Fuente: elaboración propia.

El último paso que queda es crear el *stack* o pila y el tiempo de creación variará, dependiendo de los servicios que vayamos a desplegar y la complejidad de la configuración.

# Ideas clave

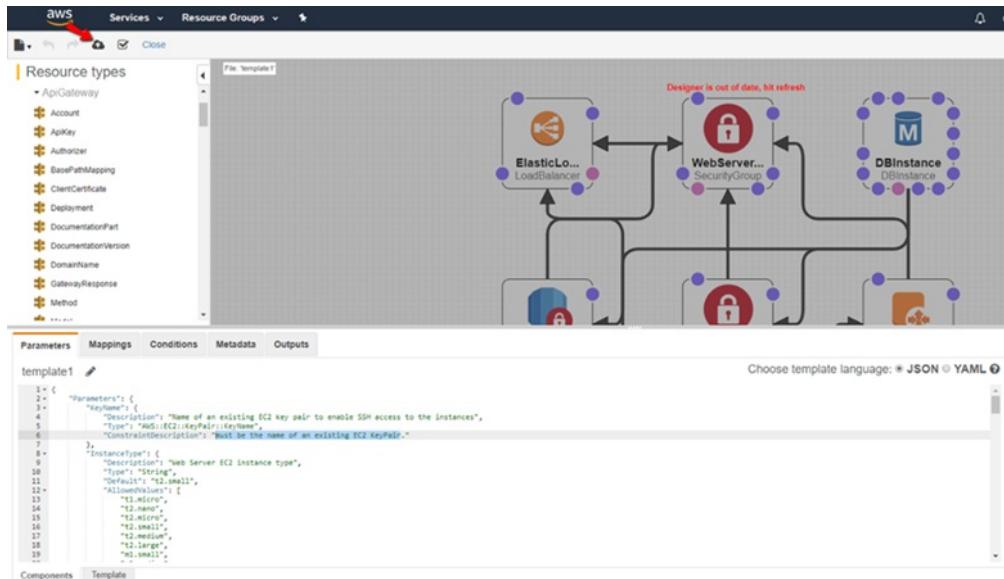


Figura 19. AWS CloudFormation creación Stack. Fuente: elaboración propia.

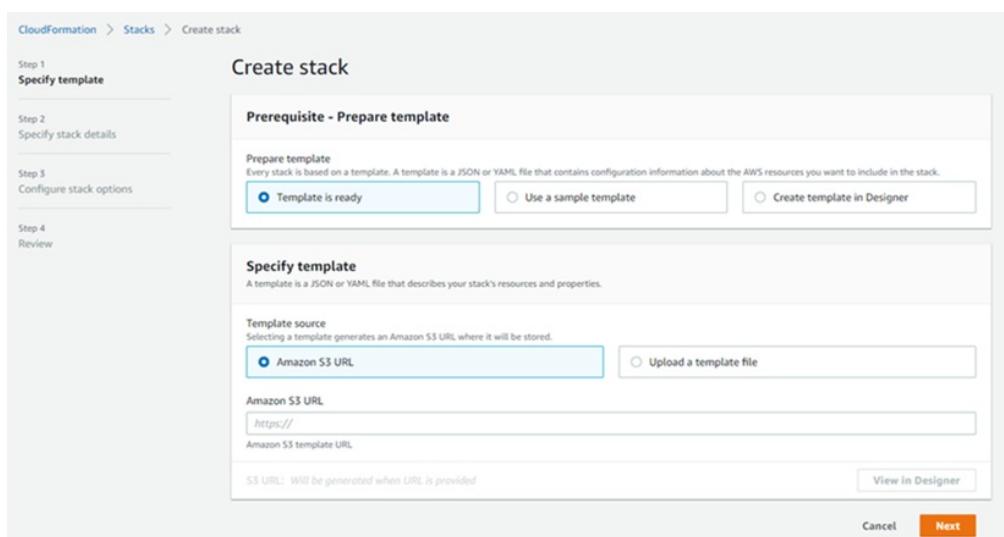
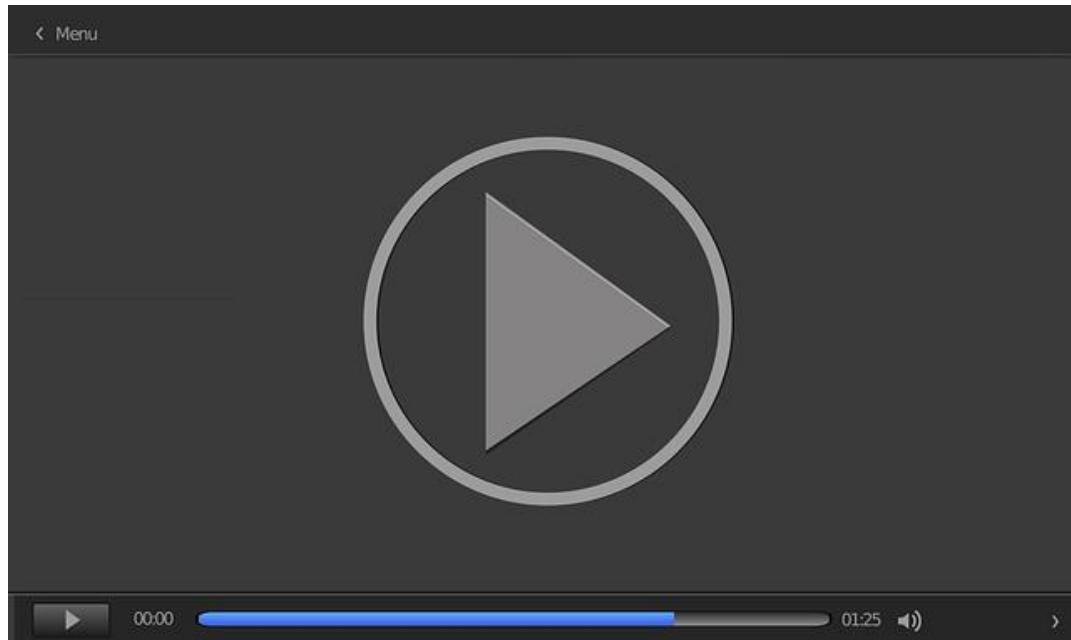


Figura 20. AWS CloudFormation creación Stack y carga de Template. Fuente: elaboración propia.

En la píldora *DevOps en Spotify* analizaremos cómo Spotify (plataforma de renombre mundial dedicada a la comercialización de servicios audiovisuales) implementa la filosofía DevOps y cuáles son las claves de su éxito.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=d3124a5d-5f6c-423b-8c6b-ad78013d2af6>

---

## 8.7. Referencias bibliográficas

Edyn Technology. (s.f.). *Infraestructura como código (IaC)*.  
<http://edyntechnology.com/servicios/infrastructure-as-code-iac/>

Amazon Web Services. (s.f.). *¿Cómo funciona AWS CloudFormation?*  
[https://docs.aws.amazon.com/es\\_es/AWSCloudFormation/latest/UserGuide/cfn-whatis-howdoesitwork.html](https://docs.aws.amazon.com/es_es/AWSCloudFormation/latest/UserGuide/cfn-whatis-howdoesitwork.html)

Amazon Web Services. (s.f.). *AWS CloudFormation*.  
<https://aws.amazon.com/es/cloudformation/>

Amazon Web Services. (s.f.). *AWS Template Format Version*. [https://s3.eu-central-1.amazonaws.com/cloudformation-templates-eu-central-1/WordPress\\_Single\\_Instance.template](https://s3.eu-central-1.amazonaws.com/cloudformation-templates-eu-central-1/WordPress_Single_Instance.template)

Andreessen, M. (2011). Why Software Is Eating the World. *The Wall Street Journal*.  
<https://www.wsj.com/articles/SB10001424053111903480904576512250915629460>

## Definición de laC

Andreessen, M. (2011). Why Software Is Eating the World. *The Wall Street Journal*.

<https://www.wsj.com/articles/SB1000142405311903480904576512250915629460>

A través de la lectura de este artículo, puedes profundizar en la infraestructura como código.

- 1.** ¿Cuál es el objetivo de la Infraestructura como Código o IaC?
  - A. Eliminar la necesidad configuración manual del hardware.
  - B. La evolución natural de los sistemas.
  - C. No tiene un objetivo claro.
  - D. Ahorrar puestos de trabajo, ya que reduce la intervención humana.
  
- 2.** ¿Es la IaC un software de programación?
  - A. Sí, porque es con la IaC con la que desarrollamos en la nube.
  - B. No, se trata de una parte fundamental de la computación en la nube y esencial para DevOps.
  - C. Sí, porque la programación es lo único que puedo hacer.
  - D. Sí, en todos los casos.
  
- 3.** ¿Cuáles de estas opciones son ventajas en de la IaC?
  - A. Optimización.
  - B. Inconsistencia de recursos.
  - C. Bajo coste y esfuerzo.
  - D. Alta dificultad.
  
- 4.** ¿Cuáles de estas herramientas son de IaC?
  - A. Microsoft Azure.
  - B. Terraform.
  - C. Chef Infra.
  - D. Json.

5. ¿Es la IaC un enfoque declarativo?

- A. Sí, porque se centra en la declaración del destino final y AWS CloudFormation se basa en este enfoque.
- B. No, es imperativo porque se centra en cómo la infraestructura se va a cambiar para cumplir este enfoque y, además, Azure se basa en este enfoque.
- C. No, es inteligente porque es esencialmente el «qué» frente al «cómo» y frente al «por qué».
- D. Todas las anteriores.

6. Relaciona Best Practice de AWS Cloudformation con su detalle:

Planificación y organización	1	A	Administrar todos los recursos de las pilas, a través de AWS CloudFormation
Creación de plantillas	2	B	Utilizar los scripts más recientes y validar las plantillas antes de usarlas.
Organizar las pilas por ciclo de vida y titularidad	3	C	Organizar las pilas por ciclo de vida y titularidad.

7. ¿Cuáles son las partes principales de AWS CloudFormation?

- A. AWS Identity y AWS S3.
- B. Plantillas y Stacks.
- C. AWS CodeCommit.
- D. Lambda.

- 8.** ¿Qué pilares y fundamentos de laC contempla AWS CloudFormation?

  - A. Replicación y escalabilidad.
  - B. Automatización mediante las API.
  - C. Actualizaciones controladas y *rollbacks*.
  - D. Las tres anteriores son ciertas.
  
- 9.** ¿En qué entorno podemos crear una plantilla?

  - A. En JSON o YAML
  - B. En AWS CloudFormation Designer.  
Es el entorno proporcionado por AWS.
  - C. En el Notepad.
  - D. No se pueden crear, solo se pueden utilizar las proporcionadas por AWS CloudFormation.
  
- 10.** ¿Cuál es el flujo de trabajo de AWS CloudFormation?

  - A. Crear permisos, crear una cesta en S3 y crear una pila.
  - B. Crear una pila, guardarla y crear la plantilla.
  - C. Crear o utilizar una plantilla, guardarla en local (o en S3) y crear la pila basada en la plantilla.
  - D. No se puede hacer flujos de trabajo en AWS.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 9. Monitorización

# Índice

[Esquema](#)

[Ideas clave](#)

[9.1. Introducción y objetivos](#)

[9.2. Herramientas de monitorización](#)

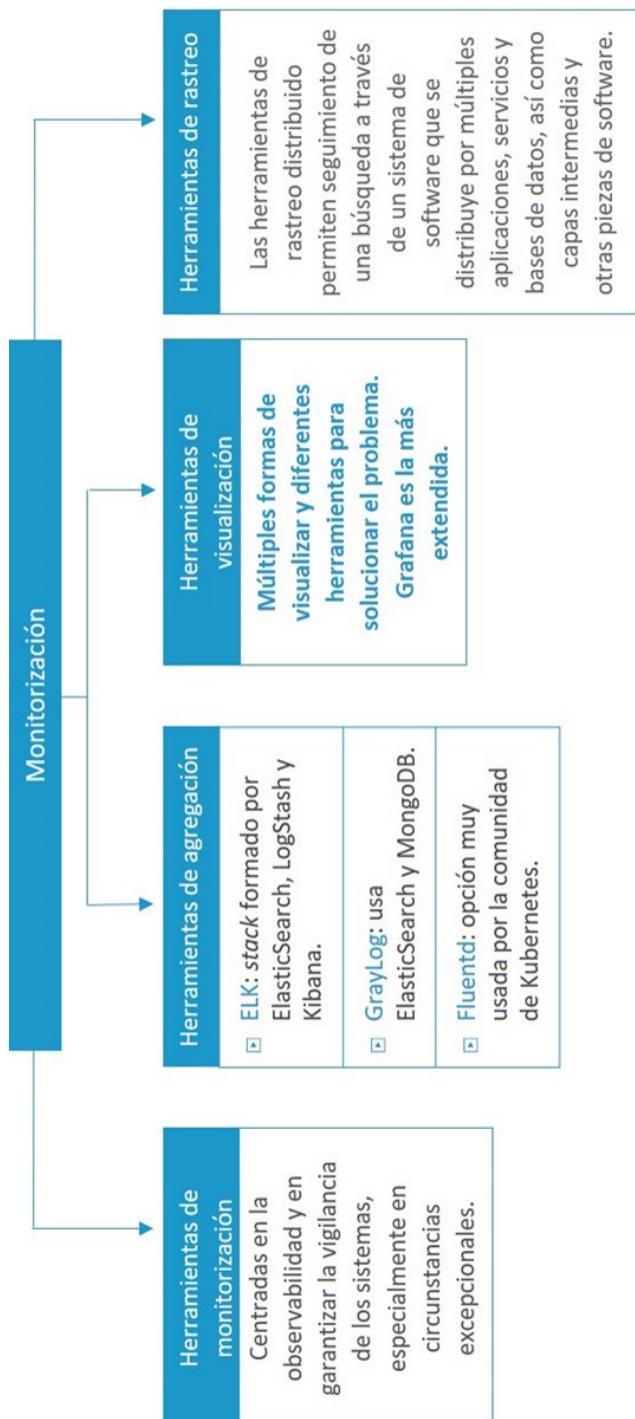
[9.3. Herramientas de agregación de logs](#)

[9.4. Referencias bibliográficas](#)

[A fondo](#)

[Tutorial de ELK](#)

[Test](#)



## 9.1. Introducción y objetivos

Los DevOps tienen, como parte de sus cometidos, no solo el desarrollar y desplegar los entornos de producción, sino el asegurar su correcto funcionamiento. Para hablar del correcto funcionamiento de un sistema, la monitorización es una pieza fundamental y necesaria. Empezaremos compartiendo algunos ejemplos del mundo real que nos ayudarán a ver la importancia de la monitorización para los DevOps.

Los objetivos que se pretenden conseguir a través del estudio de este tema son:

- ▶ Comprender la complejidad de establecer, gestionar y utilizar de forma eficiente un sistema de logs para aplicaciones complejas y multicapa.
- ▶ Entender la importancia de contar con herramientas de monitorización que permitan garantizar el mantenimiento de los sistemas durante veinticuatro horas al día y siete días a la semana.
- ▶ Descubrir herramientas que permitan abstraer la complejidad inherente de los logs mediante la agregación y el análisis automatizado.

### Una historia sobre DevOps

Antes de comenzar, es importante mencionar que la historia aquí descrita es de la autoría de Dan Barker y puedes encontrarla en su lengua original en el artículo «*The Open Source Guide to DevOps Monitoring Tools*» (en la bibliografía podrás acceder al enlace).

«Érase una vez una aplicación con graves problemas de escalabilidad, carencia de recursos en varios órdenes de magnitud y muy poco tiempo para desarrollarse. No disponíamos de agregación de logs, ni agregación de métricas, ni información distribuida ni herramientas de visualización.

Ante esta situación fue necesario trabajar directamente sobre los nodos reales de producción, utilizando herramientas como strace y grep, y accediendo de forma manual a los registros de cada sistema. No pretendo decir que estas herramientas no son estupendas, pero no facilitan el análisis de registros distribuidos y, en consecuencia, no son las herramientas más apropiadas para estudiar el rendimiento de una aplicación moderna.

Debido a ello se hizo necesario revisar una y otra vez los mismos registros múltiples veces. Creamos y probamos diferentes métodos, conectando y desconectando elementos de infraestructura y convirtiendo una tarea que debería ser sencilla en un verdadero trabajo detectivesco».

## Problemas (aparentemente) aleatorios

«En otra ocasión y escenario diferente, estábamos teniendo problemas con una aplicación en producción que sufría problemas de memoria. Los departamentos de Operaciones e Ingeniería colaboraban estrechamente para buscar una solución.

El problema no ocurría de manera consistente y era casi imposible relacionar los fallos con el tiempo de ejecución. Por ello, no podíamos predecir el fallo. Para empeorar más el panorama, el problema estaba extendido de forma sistemática en cientos de servidores y afectaba a otras aplicaciones.

Afortunadamente, contábamos con herramientas de agregación de registros (logs), herramientas de rastreo distribuido, históricos, trazas y métricas. Todas estas herramientas estaban conectadas a otras herramientas adicionales que facilitan las tareas de visualización.

Gracias a las herramientas de visualización nos fue posible detectar una tendencia y un claro incremento en el uso de memoria. Allí buscábamos un pico que nos

permitiese tener una alerta, a fin de diagnosticar el problema en tiempo real cuando este ocurriese.

Al detectar la alerta, el sistema de registros agregados permitió correlacionar otros eventos en el sistema con el pico de memoria. A su vez, esto nos permitió buscar las llamadas relacionadas y detectar una en concreto que parecía estar funcionando incorrectamente, generando una transacción que intentaba cargar un archivo enorme. La carga nunca llegaba a producirse de forma exitosa y generaba el fallo de memoria. Una vez detectado el lugar donde se producía el error, nos resultó muy fácil analizarlo en detalle y crear una solución.

Podría pensarse que la segunda situación ocurrió tiempo después de la primera y que hemos mejorado con el tiempo. O tal vez que se trata simplemente de diferentes empresas y trabajos, pero en realidad, la segunda situación tuvo lugar antes que la primera.

De hecho, cronológicamente se trata de un primer trabajo en una empresa con una cultura de automatización bastante avanzada y que hace buen uso de las herramientas e integración, mientras que en el segundo trabajo (el primero en mi historia) no se contaba con herramientas que permitieran “observabilidad o visibilidad” de sus sistemas».

## Observabilidad

La observabilidad no es solo un término comercial, sino que es un componente de control. Básicamente, **la observabilidad significa que se puede estimar un estado particular de un sistema basado en una salida**. En términos más generales, el estado de un sistema debería ser determinista de sus entradas. En un sistema suficientemente complejo, puede ser en extremo difícil implementar la plena observabilidad. Sin embargo, debemos esforzarnos por tener observabilidad como mecanismo de control. El imperativo de esta necesidad es mucho más evidente cuando el sistema falla.

## Tipos de herramientas de observabilidad

Comenzaremos por mencionar a la **agregación de métricas**. Este tipo de herramientas son, habitualmente, las que primero deberíamos incluir, ya que son fáciles de integrar en, prácticamente, cualquier aplicación escrita en un lenguaje moderno. La siguiente prioridad podría ser, por ejemplo, **monitorizar la sesión** porque, aunque probablemente requiera pequeñas modificaciones en las aplicaciones, proporciona un tremendo valor. El tercer nivel son las **alertas y sistemas de visualización** que, para su correcta implementación, se requerirá que las dos primeras herramientas (métricas y monitorización de la sesión) estén bien implementadas.

- ▶ **Agregación de métricas.** Este tipo de herramienta consiste un **conjunto de series de datos temporales**. Estas series son datos ordenados en el tiempo y tomados normalmente con un intervalo interno consistente. Su consistencia es la que permite aplicar cálculos avanzados a las series y proporcionar un análisis predictivo utilizando simples regresiones o también algoritmos más avanzados.
- ▶ **Agregación de registros.** Estas herramientas **interactúan con tipos de datos que están más relacionados con eventos** que con una serie de datos consistentes. Esta salida, a menudo, se emite cuando un sistema entra en un estado no deseado. Pensemos en ellos como una forma de componer información relevante.

- ▶ **Alertas/visualizaciones.** Puede parecer que esto no encaja con los otros tipos de herramientas, ya que es realmente posterior y más avanzado que las demás, pero proporciona una **forma eficiente de consumo para los otros tipos** e incluso puede producir su propia salida. Este tipo de herramienta hace que el sistema sea más comprensible y también ayuda a crear sistemas más interactivos a través de proactividad y reacción ante estados negativos del sistema.
- ▶ **Rastreo distribuido.** Al igual que el rastreo dentro de una sola aplicación, **el rastreo distribuido nos permite seguir una sola transacción, a través de una transacción que ocurre sobre el sistema completo.** Esto permite concentrarnos en transacciones específicas solo cuando estas podrían estar experimentando problemas. Por supuesto, es imposible trazar todo y, por lo tanto, (debido al rendimiento) lo más habitual y aconsejable es aplicar un algoritmo de muestreo.

## Características comunes para DevOps

Existen numerosas características que deben estar presentes en cualquier tipo de herramienta de observabilidad. Mencionaremos algunas de ellas ahora y profundizaremos, también, más adelante:

- ▶ **OpenAPI:** esta especificación se llamaba anteriormente Swagger, pero se renombró cuando fue adoptado por la iniciativa OpenAPI dentro de la Fundación Linux. La especificación OpenAPI es una **herramienta independiente del lenguaje, que puede generar, automáticamente, documentación de métodos, parámetros y modelos.** Swagger se utiliza para generar interfaces RESTful en HTTP, pero es independiente del protocolo. Un usuario puede crear un cliente en casi cualquier lenguaje muy fácilmente, si todavía no existe uno. Cualquier herramienta que seleccionemos debería tener este tipo de *Application Programming Interface* (en siglas, API). Si una herramienta aún no la tiene, es posible que debamos buscar otras opciones. Las herramientas que no han implementado esta especificación, o no lo tienen en su hoja de ruta, probablemente tengan otras deficiencias al adoptar estándares de código abierto y especificaciones modernas.

- ▶ **Open source:** hay muchas herramientas buenas que no son de código abierto y que pueden ser una opción adecuada para las empresas. Si eliges una de esas soluciones, asegúrate de que su documentación y las herramientas accesorias sean de código abierto. **La observación del código abierto de las herramientas puede proporcionar información valiosa sobre cómo están funcionando**, además de los otros beneficios derivados del open source.
- ▶ **Estándares abiertos:** independientemente de si una herramienta es de código abierto o no, siempre que sea posible, debe usar estándares abiertos. Ya hemos comentado uno de estos estándares, OpenAPI, pero hay muchos más. En lo sucesivo, profundizaremos un poco más en este tema para asegurarnos de saber que existen y dónde se usan.
- ▶ **Acceso fluido:** cuando mencionamos a la observabilidad y la transparencia, nos referimos a **que todos los usuarios implicados puedan ver los datos**. Las herramientas que elijas deben estar abiertas de forma predeterminada. Es posible que desees restringir algunas áreas, pero querrás facilitar el acceso y limitarlo, solo si es absolutamente necesario. Lo último que querrás son las barreras de acceso a la hora de intentar resolver un problema.
- ▶ **Federación:** es similar a la estrategia de acceso, pero permite **proporcionar información a todas las partes implicadas y controlar sus propias áreas de forma local**. Muchos sistemas heredados están diseñados de una manera que requiere que todos los datos fluyan a través de un sistema central, independientemente de la necesidad. Consecuentemente, esto también centraliza el control alrededor de esos datos. La federación habilita la agregación local en el sistema, el procesamiento y el control, mientras que la organización central recopila los mismos datos o datos resumidos. Este modelo aumenta la agilidad, flexibilidad y usabilidad.

## Agregación de métricas

Analizaremos, a continuación, las características de la agregación de métricas y las **características de los datos de series temporales**.

### Contadores

Un contador es **una métrica que representa un valor numérico que solo tenderá a aumentar** (en otras palabras, un contador nunca debería disminuir). Los contadores acumulan valores y presentan el total actual cuando el usuario lo solicita. Se utilizan para contabilizar: el número total de solicitudes web, el número de errores, número de visitantes, entre otros. Esto es análogo a una persona con un dispositivo contador en la entrada de un evento que contabiliza a todas las personas que ingresan a un establecimiento. Generalmente, no existe la opción de disminuir el contador sin tener que reiniciarlo.

### Medidores

Un medidor es similar a un contador que representa un valor numérico, pero a diferencia del contador, puede también disminuir. Es una **representación de un valor en un momento determinado**. Un termómetro es un buen ejemplo de un medidor, porque se mueve hacia arriba y hacia abajo al medir la temperatura y ofrece una lectura de un punto en el tiempo. Otros usos incluyen el mostrar la carga de la UCP (CPU), uso de memoria, red y número de hilos.

### Cuantiles

No son un tipo de métrica, pero están relacionados con las siguientes dos secciones: histogramas y resúmenes. Vamos a aclarar nuestra comprensión de los cuantiles con un ejemplo: el percentil. Este es un tipo de cuantil. Los percentiles son algo que vemos regularmente y deberían ayudarnos a comprender este concepto. Sabemos que un percentil tiene 100 valores posibles. A menudo, los vemos relacionados con pruebas médicas o resultados de rendimiento y, generalmente, afirmando la posición

de alguien dentro del percentil, por ejemplo 85 o algún otro valor. Esto significa que la persona que califica dentro de ese percentil tiene un valor real que cae dentro del 85 y 86 por ciento. Esta persona también obtuvo un puntaje en el 15 % superior de todos los estudiantes.

## Histogramas

Es **una muestra de observaciones con varios contadores**. Estos cuentan todas las observaciones y se comportan como medidores que suman los valores de las observaciones y utilizan «cubos» (cada uno de los elementos del histograma) o agrupaciones para segmentar los valores y para vincular los conjuntos de datos de una manera productiva. Esto se ve comúnmente con cuantiles relacionados con el cumplimiento de acuerdos de nivel de servicio (en siglas, SLA). Vamos a suponer que queremos asegurarnos de que el 95 % de nuestras solicitudes son respondidas en un plazo de tiempo menor a 500 milisegundos. Podríamos usar un «cubo» con un límite superior de 0,5 segundos para recopilar todos los valores que caen por debajo de los 500 milisegundos. Entonces, podríamos determinar cuántas de las solicitudes totales han caído en ese cubo (sección del histograma). También podemos determinar qué tan lejos estamos de lo establecido en el *Service Level Agreement* (en siglas, SLA), pero esto puede ser difícil de hacer con precisión. Los histogramas son métricas agregadas que se acumulan desde múltiples instancias hasta un servidor central. Esto proporciona una oportunidad para entender el sistema como un todo, en lugar de nodo por nodo.

## Resúmenes

Los resúmenes son similares a los histogramas en el sentido de que **unifican muchas observaciones, pero la agregación ocurre en el servidor**. Un resumen también usa una ventana de tiempo deslizante, por lo que sirve para casos ligeramente diferentes que un histograma, pero se usa para los mismos tipos de métricas. En la práctica, lo más usual es que se utilice un histograma, a menos que necesiten medidas más precisas.

## **Pull-push**

No se puede escribir ningún texto sobre las herramientas de agregación de métricas sin abordar el debate *push vs. pull*. ¿Qué es? El debate se centra en si es mejor tener un sistema de agregación para que se envíen datos o tener un sistema de agregación de métricas que reúna datos preguntando a un API.

Hay muchas herramientas disponibles, tanto de código abierto como comerciales. Nos centraremos en herramientas de código abierto, pero algunas tienen un modelo de núcleo abierto con algunos componentes de pago. Además, algunas de estas cuentan con componentes adicionales de observabilidad, principalmente centrados en las alertas y la visualización.

## 9.2. Herramientas de monitorización

### Prometheus

Esta es la **herramienta de monitorización de series temporales** más reconocida (y popular) y la solución ideal para aplicaciones nativas de la nube. Está alojada dentro del Cloud Native Computing Foundation (en siglas, CNCF), pero fue creado por Matt Proud y Julius Volz, y patrocinado por SoundCloud. Existe mucha documentación sobre Prometheus que te ayudará a entender cómo funciona, por ejemplo, visitando [su página web](#).

Prometheus es un sistema basado en extracción (*pull*) que utiliza una configuración local para definir cómo recolectar los datos. Esta información se recopila y guarda en un motor de almacenamiento altamente eficiente, dentro del disco local. El sistema de almacenamiento utiliza un archivo único agregado por métrica. Este almacenamiento no es con pérdida (nos estamos refiriendo a que, en este caso, almacenamos todos los datos, no solo resúmenes o los datos más significativos), lo que significa que la fidelidad de los datos hace un año es tan alta como la de los datos que se están recopilando en la actualidad. Sin embargo, es posible que un usuario no desee conservar tantos datos locales. Afortunadamente, hay una opción de almacenamiento remoto para retención y análisis de datos a largo plazo.



Figura 1. Ejemplo de datos de Prometheus con Grafana. Fuente: Admin Magazine, s.f.

Prometheus incluye un lenguaje avanzado para seleccionar y presentar datos llamados PromQL. La información se puede mostrar de forma gráfica, tabular o ser expuesta externamente a través de una API REST. El lenguaje permite a un usuario crear regresiones, analizar datos en tiempo real, o datos históricos de tendencias. Las etiquetas también son una gran herramienta para intercalar y consultar datos, y se pueden asociar con nombres de métricas, por ejemplo.

Prometheus también ofrece un modelo de federación, que favorece un control más localizado al permitir que los equipos tengan su propio Prometheus, mientras que los equipos centrales también pueden tener uno propio. Los sistemas centrales podrían atacar los mismos *end-points* que los Prometheus locales, pero también pueden atacar el local (del servidor central) para obtener los datos agregados de forma local.

Otro dato que resulta muy interesante sobre Prometheus es que dispone de un gestor de alertas: Alertmanager. Este sistema habilita la agregación de alertas, así como flujos más complejos para limitar cuando se envía una alerta. Por ejemplo, supongamos que diez nodos generan la misma alarma al mismo tiempo. Probablemente, no se necesite enviar una alerta por cada uno de los diez nodos,

sino que parece más eficiente enviar una única alerta que contenga toda la información. También, es posible enviar un correo electrónico al equipo de sistemas para que sepan que esos nodos están inactivos.

## Graphite

Graphite existe desde hace mucho tiempo y se ha vuelto omnipresente en la industria. Es un sistema de inserción de logs que recibe datos desde las aplicaciones. El funcionamiento es relativamente simple, cada aplicación inyecta (*push*) los datos hacia el componente de Graphite. Carbon, por su parte, almacena estos datos en la base de datos Whisper, y esa base de datos junto con Carbon se leen desde el componente web Graphite, que actúa como interfaz de usuario para representar gráficamente sus datos en un navegador. En la versión de código abierto, todo se ejecuta en un host único: no hay datos o configuración almacenados en sistemas externos, por lo que es bastante fácil de administrar, pero es menos robusto que la versión comercial.



Figura 2. Ejemplo de salida producida por Graphite. Fuente: Montilla, 2017.

## OpenTSDB

OpenTSDB es una **base de datos de series de tiempo de código abierto**, como su nombre lo indica (Open Time Series Data Base u OpenTSDB). Esta herramienta almacena sus métricas en Hadoop y esto significa que es inherentemente escalable. Si en tu entorno de trabajo tienes un clúster Hadoop, esta podría ser una buena opción para almacenar métricas a largo plazo. En el caso contrario, si no cuentas con un clúster Hadoop, la sobrecarga operativa podría ser demasiado grande. Sin embargo, OpenTSDB es compatible con Google Bigtable como *backend*, que es un servicio en la nube que no requiere mantenimiento.

Esta herramienta utiliza un sistema de emparejamiento clave-valor al que llama «etiquetas» para identificar métricas y agregar dimensionalidad. Tiene un lenguaje de consulta, pero es más limitado que Prometheus ('PromQL), aunque, sin embargo, tiene varias funciones integradas (muy interesantes) con el aprendizaje automático. La API es el principal punto de entrada para las consultas, similar a InfluxDB.

Por último, añadiremos que OpenTSDB no ofrece una capacidad de alerta, lo que dificulta la integración con la respuesta a incidentes que estén en proceso. Este tipo de sistema puede ser excelente a largo plazo. A diferencia de Prometheus, aquí se trata de garantizar el almacenamiento de datos para realizar un análisis histórico y revelar problemas sistémicos, en lugar de utilizarse para la identificación y respuesta rápida a problemas puntuales.

## Estándar OpenMetrics

OpenMetrics está formado por un grupo de ingenieros que busca establecer un **formato de publicación estándar para datos de métricas**. Si esta iniciativa es exitosa, en el corto o mediano plazo, tendríamos una abstracción en toda la industria que nos permitiría utilizar herramientas y proveedores con facilidad, cambiando frecuentemente de unas a otras sin ninguna fricción. Empresas líderes como Datadog ya han comenzado a ofrecer herramientas que pueden consumir el formato de publicación de Prometheus.

## InfluxDB

InfluxDB es un producto relativamente nuevo, más reciente que Prometheus, y que **utiliza un modelo de núcleo abierto (solo el producto básico es open source), pero con posible coste adicional, si se desea escalado y agregación**. InfluxDB es parte de la pila (*stack*) Telegraf, InfluxDB, Chronograf y Kapacitor (en siglas, TICK), por lo que incluiremos todas las características de esos componentes. InfluxDB utiliza un sistema de pares clave-valor llamado «etiquetas» para agregar dimensionalidad a las métricas, similar a Prometheus y Graphite. Los datos métricos pueden ser de tipo float64, int64, *bool* y *string* con resolución de nanosegundos, lo que es un rango más amplio que la mayoría de las otras herramientas en este apartado. De hecho, la pila TICK es más una plataforma de agregación de eventos que un sistema de agregación de series de tiempo nativa. InfluxDB utiliza un registro de escritura anticipada y una colección de archivos de datos de solo lectura. La arquitectura de esta pila es diferente dependiendo si se trata de la versión de código abierto o la comercial. El proyecto incluye Google e InfluxData (entre otros), lo que significa que InfluxDB, eventualmente, adoptará el estándar de OpenMetrics.

## Tipos de agregación

**Los sistemas de agregación de registros no pueden realizar las mismas tareas o funciones que los sistemas de agregación de métricas.** Muchos vendedores presentan sus sistemas de agregación de registros como la solución a todos los problemas de observabilidad. A este respecto, es necesario tener en cuenta que, si bien la agregación de registros es una herramienta valiosa, no es una buena herramienta para obtener datos de series temporales. Algunas características valiosas en una serie temporal de métricas son la existencia de un intervalo regular y el sistema de almacenamiento personalizado, específicamente diseñado para series de datos temporales. Si se utiliza un sistema de agregación de registros en un intervalo regular, puede funcionar de forma similar a las métricas. Sin embargo, el sistema de almacenamiento no está optimizado para las consultas que son típicas en un sistema de agregación de métricas. Por tanto, es probable que un sistema de agregación no sea apto para series temporales de datos. Entonces te preguntarás: ¿para qué sirve?

**Un sistema de agregación de registros sirve, principalmente, para colecciónar datos de eventos, es decir, actividades irregulares que son significativas.** Un ejemplo podrían ser los registros de acceso para un servidor web, que resultan relevantes porque queremos saber qué o quién está accediendo a nuestros sistemas y cuándo. Otro ejemplo sería un error de aplicación, porque no es una condición de funcionamiento normal.

Recomendaciones para guardar registros de eventos:

- ▶ Incluye una marca de tiempo.
- ▶ Usa un formato orientado a documentos (JSON).
- ▶ No registres eventos insignificantes.
- ▶ Registra todos los errores de la aplicación.

- ▶ Incluye *warnings*.
- ▶ Incluye un registro de «encendido».
- ▶ Escribe mensajes que resulten legibles (por seres humanos).
- ▶ No registres datos informativos de producción.
- ▶ No registres nada que un humano no pueda leer o ante lo que no pueda reaccionar.

## Costes de agregación en la nube

Al investigar las herramientas de agregación de registros, la nube podría resultarnos una opción atractiva. Sin embargo, **hay que tener en cuenta que puede traer acarreados costes significativos**. Los registros pueden representar una gran cantidad de datos cuando se agregan datos de cientos o miles de hosts y aplicaciones. La ingestión, almacenamiento y recuperación de esos datos puede llegar a ser muy costosa en sistemas basados la nube.

Como punto de referencia de un sistema real, una colección de alrededor de 500 nodos con unos cientos de aplicaciones resulta en 200 GB de datos de registro por día. Probablemente, hay margen de mejora para este sistema, pero incluso reduciéndolo a la mitad costará casi 10 000 dólares por mes en muchos softwares como servicio (en siglas, SaaS) especializados. Por supuesto, también hay que tener en cuenta que, en general, los costes tienden a bajar, que redunda en nuestro beneficio, pero al mismo tiempo nuestra necesidad de almacenamiento crecerá, con lo cual dicho beneficio se verá compensado.

## 9.3. Herramientas de agregación de logs

El listado de posibles herramientas no es muy largo, sobre todo si lo que pretendemos es que se puedan ver datos de las tendencias de más de un año atrás. Veremos, a continuación, algunas herramientas de código abierto y otras de pago pensadas para el despliegue en un entorno privado y otras que son ofrecidas como servicio.

### ELK

ELK es la sigla correspondiente a Elasticsearch, Logstash y Kibana, y es **la herramienta de agregación de registros de código abierto más popular del mercado**. Netflix, Facebook, Microsoft, LinkedIn y Cisco la han incorporado a su conjunto de herramientas. Elástic es el responsable de desarrollar y mantener todos los componentes. Elasticsearch es esencialmente una base de datos No SQL Lucene con una implementación de motores de búsqueda. Logstash es un sistema de tuberías de registros que ingiere datos, los transforma y los carga en un almacenamiento como Elasticsearch. Kibana, por su parte, es una capa de visualización sobre Elasticsearch.

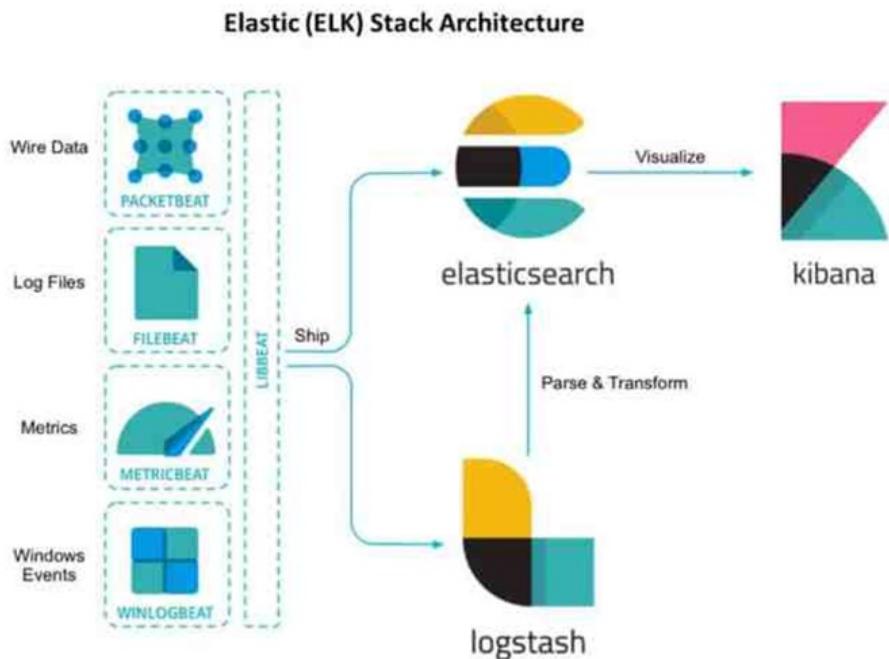


Figura 3. Arquitectura ELK. Fuente: SysAdminXpert, 2020.

**Hace unos años se presentaron los «Beats». Estos son colecciones de datos que simplifican el proceso de envío de datos a Logstash.** En lugar de necesitar comprender la sintaxis más apropiada para cada tipo de registro, un usuario puede instalar un Beat que exportará registros de NGINX o registros proxy de Envoy correctamente para que puedan ser usados efectivamente dentro de Elasticsearch.

Al instalar una pila ELK para dar servicio a un entorno de producción, a menudo se incluyen otras piezas como Kafka, Redis, y NGINX. Además, es común reemplazar Logstash con Fluentd, del que hablaremos más adelante. Este sistema puede ser complejo a la hora de operar y, de hecho, en sus primeros tiempos fueron muy frecuentes las quejas de los usuarios. Afortunadamente, muchos de los *issues* y fallos se han solucionado, pero sigue siendo un sistema complejo, por lo que es posible que no deseas probar suerte si vas a implementarlo en una organización pequeña.

Dicho esto, **hay servicios disponibles que hacen que el usuario no tenga que preocuparse por su ejecución**: Logz.io es uno de ellos. De todos modos, como comentamos antes, para el caso de ciertos servicios en la nube, el coste para una gran cantidad de datos es relativamente elevado. En caso de que no pudieses permitirte Logz.io, una buena opción podría ser Amazon Elasticsearch Service que es un servicio de Amazon Web Services (en siglas, AWS) integrado con Lambda y S3. Esta es una opción, *a priori*, mucho más barata, pero se requiere algo de administración y tiene ciertas limitaciones.

---

Puedes obtener más información acerca del servicio Elasticsearch en Amazon Web Services a través de la siguiente dirección web:  
<https://aws.amazon.com/es/elasticsearch-service/>

---

Elastic, la empresa matriz de ELK, ofrece un producto más robusto que utiliza el modelo de núcleo abierto (el módulo principal es *open source*, pero se lo pueden añadir componentes privativos) al que se le pueden añadir módulos comerciales con funcionalidad extra. También, se puede alojar en las plataformas Google Cloud o AWS. Esta podría ser la mejor opción, ya que esta combinación de herramientas y plataformas de alojamiento ofrece un precio más económico.

La pila ELK también ofrece excelentes herramientas de visualización a través de Kibana, pero carece de una función de alertas. Elastic proporciona funcionalidad de alertas dentro del complemento comercial X-Pack, pero no hay ninguna solución para la versión de código abierto. Yelp ha creado una solución a este problema, llamada ElastAlert.

## Graylog

Graylog ha aumentado recientemente en popularidad, pero fue creado por Lennart Koopmann en 2010. A pesar de que su uso es cada vez mayor, todavía va muy por

detrás de ELK. La razón principal es que tiene menos comunidad, soporte y características, aunque puede usar los mismos Beats que la pila ELK. Graylog ha ganado elogios en la comunidad Go con la introducción del Graylog Collector escrito en Go.

En su *backend*, Graylog utiliza Elasticsearch y MongoDB. Esto lo hace tan complejo de ejecutar como la pila de ELK o, incluso, un poco más. Sin embargo, Graylog viene con alertas integradas en la versión de código abierto, así como otras características notables como geolocalización.

Uno de los mayores problemas con los sistemas de agregación de registros es la latencia, los *streams* eliminan ese problema en Graylog. Tan pronto como el registro entra, se puede enrutar a otros sistemas a través de un *stream* sin ser procesado totalmente.

**La característica más sobresaliente de Graylog es la capacidad de geolocalización, que admite el trazado de direcciones IP en un mapa.** Esta es una característica bastante común y también está disponible en Kibana, pero agrega mucho valor, sobre todo al estar disponible en la versión de código abierto. Esta compañía ofrece soporte a los usuarios por un coste adicional. También ofrece un modelo abierto (*open source* con funcionalidad básica) y la posibilidad de ampliar a una versión *Enterprise* que ofrece registros de auditoría y soporte adicional. No hay muchas otras opciones de soporte o alojamiento, fuera de las oficiales.

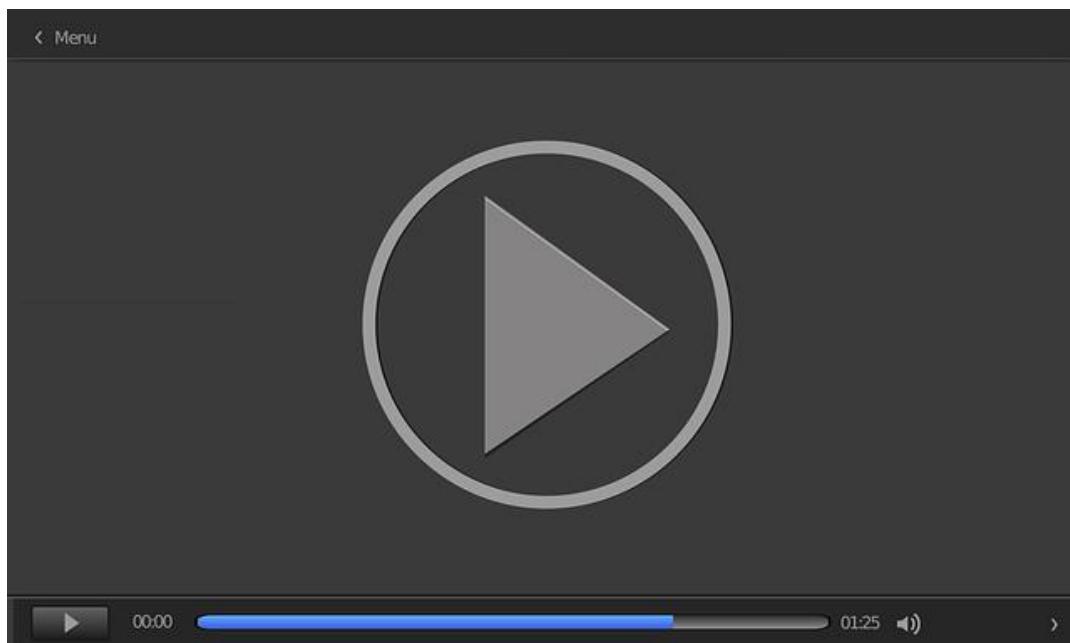
## Fluentd

Fluentd fue desarrollado por Treasure Data. Está escrito en C y es una de las herramientas de referencia recomendadas por AWS y Google Cloud. Para muchas empresas, se ha convertido en un posible reemplazo para Logstash. **Actúa como un agregador local que recopila todos los registros de nodos y los envía a los sistemas de almacenamiento centralizados.** Utiliza un robusto sistema de complementos para proporcionar variadas integraciones que lo hacen compatible con

numerosas fuentes de datos y formatos de salidas. Hay más de 500 complementos diferentes disponibles, así que los casos de uso más comunes están totalmente resueltos. Y, de hecho, en el caso excepcional de que no exista, el usuario puede optar por contribuir y construir uno.

Fluentd es una opción muy común en el entorno de Kubernetes, debido a sus bajos requisitos de memoria (solo decenas de megabytes) y su alto rendimiento. En un sistema basado en Kubernetes, donde cada pod tiene auto escalabilidad, el consumo de memoria aumentará linealmente con cada nuevo pod creado. Su utilización Fluentd reducirá drásticamente el uso de recursos del sistema, a diferencia de lo que ocurre con algunas aplicaciones java que no han sido diseñadas con esta capacidad.

Los microservicios son una arquitectura en auge en la actualidad y están siendo adoptados por múltiples equipos en todo el globo. Sin embargo, una incorrecta estrategia sobre la aplicación de los microservicios puede tener un impacto negativo de gran escala. En el vídeo *Microservicios* analizaremos los errores más comunes que todo desarrollador debe evitar.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=9db6635d-5b11-49fc-8962-ad78013d2a25>

---

## 9.4. Referencias bibliográficas

Admin Magazine. (s.f.). Efficiently planning and expanding the capacities of a cloud.

Admin Magazine. [https://www.admin-magazine.com/Archive/2018/44/Efficiently-planning-and-expanding-the-capacities-of-a-cloud/\(offset\)/6](https://www.admin-magazine.com/Archive/2018/44/Efficiently-planning-and-expanding-the-capacities-of-a-cloud/(offset)/6)

Montilla, I. F. (2017, febrero 11). Scaling django: measuring your django app's performance. Dev Community. <https://dev.to/ferminm/scaling-django-2-profiling-your-django-app-4in6>

SysAdminXpert. (2020, junio 12). *ELK Stack Architecture Elasticsearch Logstash and Kibana*. <https://sysadminxpert.com/elk-stack-architecture-elasticsearch-logstash-and-kibana/>

## Tutorial de ELK

Paradigma Digital. (2019, abril 5). *Curso Elastic Search – Capítulo 1* [Vídeo]. YouTube. <https://www.youtube.com/watch?v=UIN2NeMb7xc>

Si bien este tema solo pretende dar una visión general de las herramientas disponibles sin entrar en la instalación u operación de las herramientas concretas, ELK es una de las herramientas líderes en el mercado, y este recurso te permitirá saber más sobre eso.

- 1.** Selecciona la definición correcta de histograma:

  - A. Un histograma muestra los datos acumulados de tipos de datos.
  - B. Es un tipo de gráficos.
  - C. Facilidad de encontrar la información de forma rápida.
  - D. Ninguna de las anteriores.
  
- 2.** Selecciona la definición correcta de observabilidad:

  - A. Característica que permite estimar un estado particular de un sistema basado en una salida.
  - B. Capacidad de facilitar la visión de los datos.
  - C. Facilidad de encontrar la información en la herramienta.
  - D. Ninguna de las anteriores.
  
- 3.** Selecciona la definición correcta de agregación de métricas:

  - A. Es una herramienta que generalmente se utiliza para analizar series de datos temporales.
  - B. Capacidad de facilitar la visión de los datos agregados.
  - C. Facilidad de encontrar la información en la herramienta.
  - D. Ninguna de las anteriores.
  
- 4.** Selecciona la afirmación que se mejor describa el rastreo distribuido:

  - A. El rastreo distribuido es similar al rastreo estándar pero la carga se reparte.
  - B. El rastreo distribuido te permite seguir una sola transacción a través de una transacción que ocurre sobre el sistema completo.
  - C. Las respuestas A y B son correctas.
  - D. Ninguna es correcta.

5. Selecciona la afirmación que mejor describe Prometheus.

  - A. Esta es la herramienta de monitorización de series temporales más reconocida.
  - B. Una herramienta de monitorización.
  - C. Las respuestas A y B son correctas.
  - D. Ninguna es correcta.
6. Selecciona la afirmación que mejor describe la federación desde el punto de vista de la monitorización.

  - A. La federación permite tener control y datos globales, pero manteniendo el control local.
  - B. El modelo de federación tiene que ver con la gobernanza de IT y, por lo tanto, no está relacionado con DevOps.
  - C. Las respuestas A y B son correctas.
  - D. Ninguna es correcta.
7. Selecciona la afirmación que mejor describe las implicaciones del Open Source.

  - A. El software es gratis y puedes hacer lo que quieras.
  - B. El desarrollador del software lo pone a disposiciones de los usuarios bajo una licencia concreta donde, entre otros derechos, existe el acceso al código fuente.
  - C. El software está desarrollado siguiendo estándares que garantizan su compatibilidad.
  - D. Ninguna es correcta.

- 8.** Selecciona la afirmación que mejor describe a la Open API.

  - A. Es un API que está abierto, es decir, sin securizar.
  - B. Es un API que no contempla federación.
  - C. OpenAPI es un estándar abierto para las API.
  - D. A y B son correctas.
- 9.** Selecciona la afirmación que mejor describe la herramienta ELK.

  - A. Es una de las herramientas de monitorización más usadas.
  - B. Es un protocolo de transmisión de logs.
  - C. A y D son correctas.
  - D. ELK es el stack conformado por ElasticSearch, Logstash y Kibana.
- 10.** Selecciona la afirmación que mejor describe el término pull-push.

  - A. Es una de las herramientas de monitorización más usadas junto con ELK.
  - B. Es un protocolo de recepción de logs.
  - C. Se refiere al dilema sobre las ventajas e inconvenientes de construir sistemas que llaman a otro para solicitar información o por el contrario esperan a ser informados.
  - D. Ninguna de las anteriores.

Cloud Computing, DevOps y DevOps Culture

---

## Tema 10. Visualización y alarmas

# Índice

[Esquema](#)

[Ideas clave](#)

[10.1. Introducción y objetivos](#)

[10.2. Herramientas de alerta](#)

[10.3. Herramientas de visualización](#)

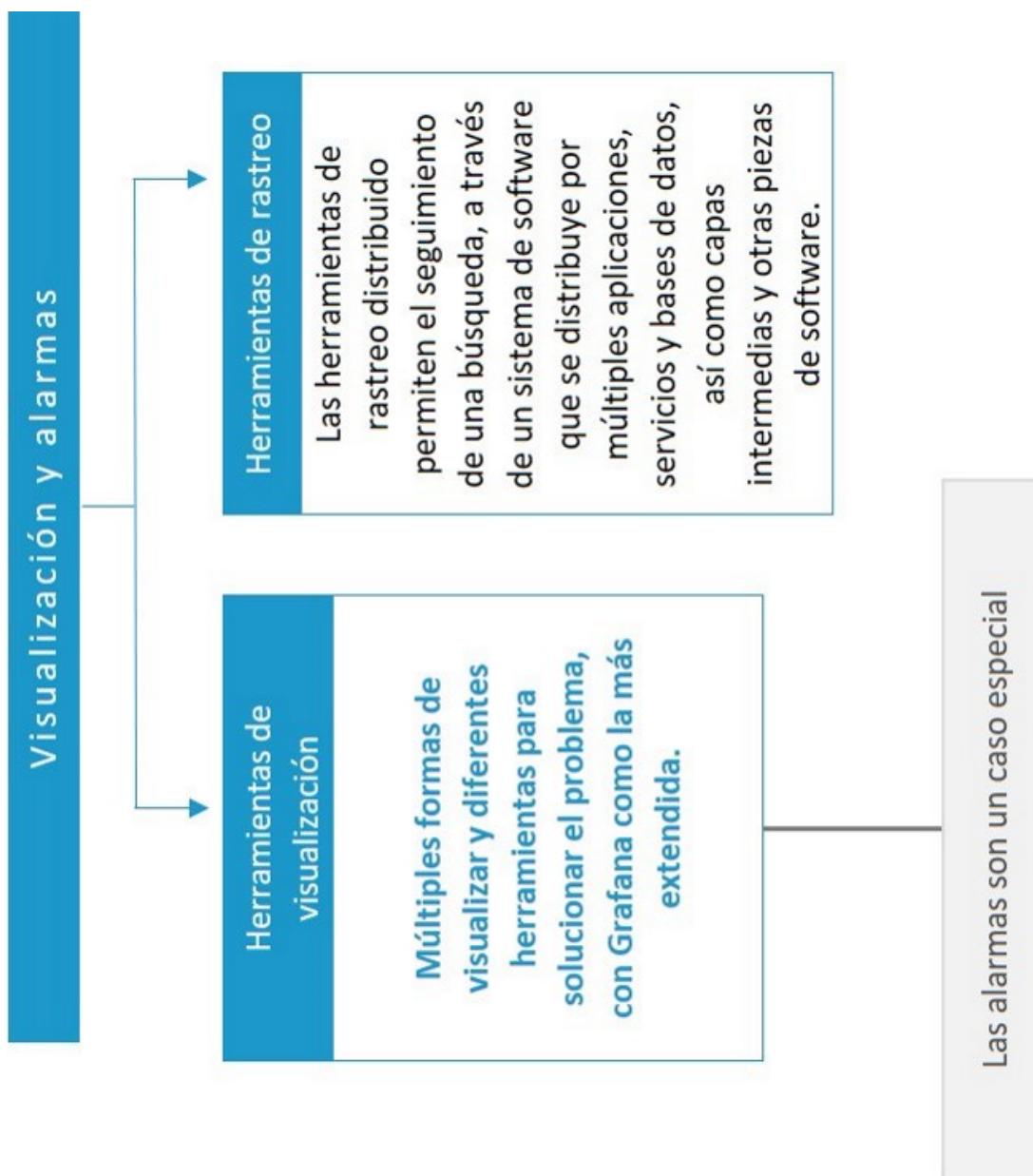
[10.4. Herramientas de rastreo distribuido](#)

[10.5. Referencias bibliográficas](#)

[A fondo](#)

[Tutorial de Grafana](#)

[Test](#)



## 10.1. Introducción y objetivos

Los ingenieros DevOps tienen como parte de sus cometidos desarrollar y desplegar los entornos de producción, pero, además, asegurar su correcto funcionamiento. La monitorización es una pieza fundamental para garantizar el funcionamiento de un sistema, pero también lo son las alertas y que todos los datos asociados estén accesibles para ser utilizados. Por poner un ejemplo, el mejor sistema de *logs* no sirve de nada si nadie puede acceder a él para analizarlo.

A partir del estudio de este tema, se pretenden conseguir los siguientes objetivos:

- ▶ Conocer la función que desempeñan las herramientas de visualización, las alarmas y las herramientas de rastreo para el mantenimiento de los sistemas.
- ▶ Reconocer cuáles son los casos de uso y las mejores prácticas de las alertas en el contexto de DevOps.
- ▶ Conocer cuáles son las herramientas más importantes y populares en el mercado actual para la visualización y rastreo.

Las herramientas de visualización también son llamadas herramientas de observabilidad. Estas últimas provienen de la teoría del control y hacen referencia a nuestra capacidad para entender un sistema a través de sus entradas y salidas. Podría considerarse a las herramientas de visualización y alerta como aquellas que proporcionan representaciones estructuradas de salidas del sistema. **Las alertas son, básicamente, una selección de salidas negativas del sistema y las visualizaciones son representaciones estructuradas desambiguadas, enfocadas en facilitar la comprensión al usuario.**

A la hora de hablar de las alertas, resulta importante destacar la importancia de poder filtrarlas y gestionarlas de forma correcta. Estas no deben enviarse si no son significativas para el usuario que las recibe, especialmente si dichas alertas van

dirigidas a varias personas y solo unos pocos pueden responder a situaciones anómalas del sistema que desencadena la alerta. La razón de peso para respetar esta práctica reside en el hecho de que **los usuarios que reciben una alerta, pasado algún tiempo, llegarán a ignorarlas y asumirán que no son importantes o que no van dirigidas a ellos.** Por ejemplo, si un operador está recibiendo cientos de correos electrónicos al día desde el sistema de alertas, ese operador en poco tiempo va a ignorar todos los correos electrónicos del sistema de alerta. El operador solo responderá a un incidente real cuando él o ella esté experimentando el problema en su propia piel, cuando se lo reporte su jefe o un cliente usuario del sistema.

**Las alertas no deben usarse como un flujo constante de información o una actualización de estado.** Están destinadas a notificar un determinado problema sobre el que el sistema no puede recuperarse de forma automática y se deben enviar solo a la persona que tenga la capacidad de poder recuperar el sistema. Aquello que quede fuera de esta definición no es una alerta y solo perjudicará a los empleados y a la cultura de la empresa.

Todos tenemos en mente un conjunto diferente de tipos de alerta, pero no deberían implicar niveles de prioridad (P1-P5) o utilizar palabras como «informativo», «advertencia» o «crítico». En su lugar, deberían describir las categorías genéricas que faciliten la respuesta a incidentes del sistema. Es posible que hayas notado que antes se ha mencionado un tipo de alertas «informativo», a pesar de que ahora se hace hincapié en que estas alertas no deberían estar incluidas. Esto sucede porque en la documentación de muchas herramientas y ejemplos se les llama incorrectamente (a mi juicio) alertas.

## Alertas y herramientas de visualización de alertas

Los gráficos de barras y de líneas son los más habituales y frecuentes a la hora de representar este tipo de información: agregación de eventos y alarmas. Dependiendo

de la complejidad de la información que se desee mostrar y las dimensiones de los datos, también existen otras opciones que enriquecen la información y facilitan su análisis. A continuación, veremos algunas de ellas.

## Mapas de calor

Este tipo de visualización resulta extremadamente útil para representar la información que proviene de histogramas. Es similar a un gráfico de barras, pero más enriquecido: puede mostrar gradientes dentro de las barras, que representan los diferentes percentiles respecto a la métrica general. Por ejemplo, **si necesitas buscar latencias de solicitud y comprender rápidamente la tendencia general, así como la distribución de todas las solicitudes, el mapa de calor resulta un gran aliado.** Además, permite usar colores para desambiguar la cantidad de cada sección de las barras.

## Medidores

Los medidores se usan para mostrar una sola métrica de forma rápida, igual que un velocímetro muestra la velocidad, o un medidor de gas representa la cantidad de gas. **Al igual que ocurre con un medidor de gas, la mayoría de los medidores de monitorización indican claramente qué es bueno y qué no.** El código de colores es el indicador visual que permite al usuario interpretar el estado del sistema: el verde indica estabilidad, el naranja, riesgo inminente, y el rojo, estado crítico.

## 10.2. Herramientas de alerta

### Bosun

Seguramente conoces Stack Overflow, que es muy popular entre los desarrolladores y también lo es entre los profesionales de operaciones. Stack Exchange es una red similar a Stack Overflow que se ha encargado de crear, implementar y mantener a Bosun durante varios años. A finales de 2019, Skyscanner ha anunciado que asumiría las tareas y responsabilidades hasta entonces desarrolladas por Stack Exchange.

Al igual que Prometheus, Bosun está escrito en Go y tiene más funcionalidad que Prometheus, ya que **puede interactuar con muchos sistemas en contextos más complejos que la agregación de métricas** (es compatible con Graphite, InfluxDB, OpenTSDB y Elasticsearch) y, adicionalmente, **también puede consumir datos de sistemas de agregación de registros y eventos**. La arquitectura de Bosun consiste en: un único servidor, un backend como OpenTSDB, Redis y sus agentes recolectores. Estos últimos detectan automáticamente los servicios en un *host* e informan sobre las métricas para esos procesos y otras métricas adicionales que pueda originar el sistema. El servidor de Bosun consulta los *backends* para determinar si existe alguna situación que requiera el disparar una alerta.

Bosun también **puede ser utilizado por herramientas como Grafana para facilitar las consultas a los backends subyacentes a través de una única interfaz**. Otro caso de uso común es el uso de Redis para almacenar estados y metadatos de Bosun.

Por último, agregaremos que una característica realmente interesante de esta herramienta es **que permite validar las reglas de alertas contra datos históricos**. Esto es algo que se echa de menos en Prometheus y que, probablemente, incluyan en algún momento.

## Cabot

Cabot fue creado por una compañía llamada Arachnys con el objetivo de poder monitorizar servicios y no solo las máquinas. Parece un concepto simple, pero es de una importancia crucial, ya que en los entornos actuales de nada nos sirve saber que una máquina está funcionando si los contenedores, servicios web o microservicios que aloja, han dejado de responder de forma correcta.

Cabot permite monitorizar servicios lógicos, que se ejecutan en una máquina o 100, como unidades lógicas. Es decir, no se preocupa de monitorizar las características físicas de un clúster (cosas como disco, memoria, etc.), sino que su trabajo se centra en recibir alertas cuando el tiempo de respuesta de su API deja de ser aceptable, por ejemplo.

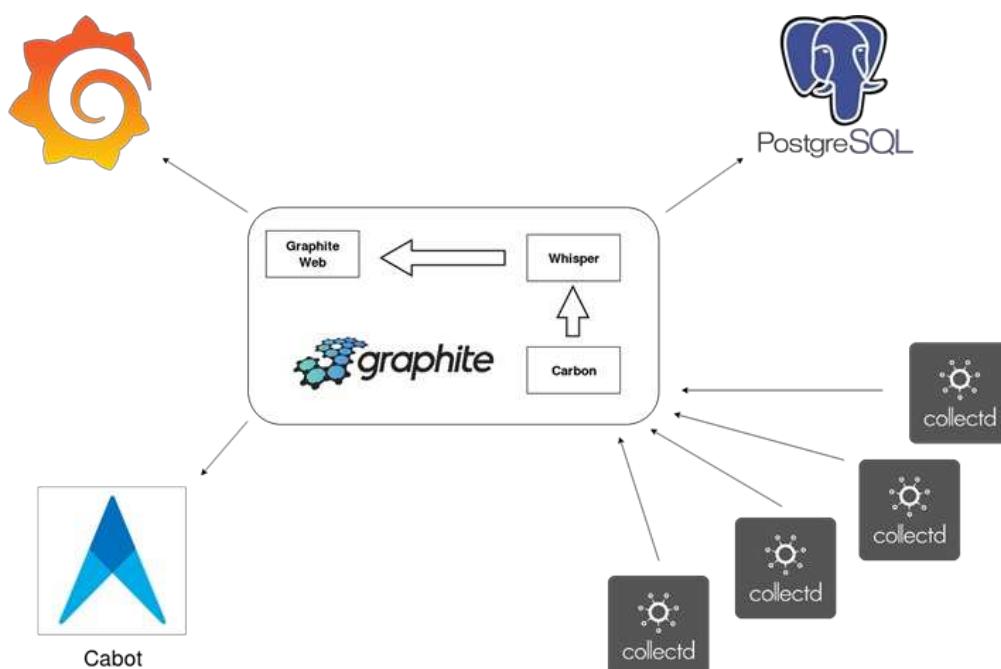


Figura 1. Sistema de alertas con Cabot y PostgreSQL. Fuente: Hackerearth, s.f.

## Flame Graphs

Esta herramienta presenta una interfaz muy novedosa: gráficos de llamas. Si lo que

buscas es realizar un análisis visual muy rápido y poco exhaustivo, esta no es tu herramienta. En cambio, si lo que es deseas conocer en detalle las causas de un fallo y evaluar problemas específicos, puedes estar seguro de que sacarás partido de ella.

Creada en 2011, esta herramienta se centra en la CPU y memoria y las tramas de datos asociadas. El eje X enumera segmentos en orden, y el eje Y muestra la profundidad de la pila. Cada rectángulo es un marco de pila e incluye la función que lo llama. Cuanto más ancho es el rectángulo, más aparece en la pila. Este método es invaluable cuando se trata de diagnosticar el rendimiento del sistema a nivel de la aplicación.

# Ideas clave

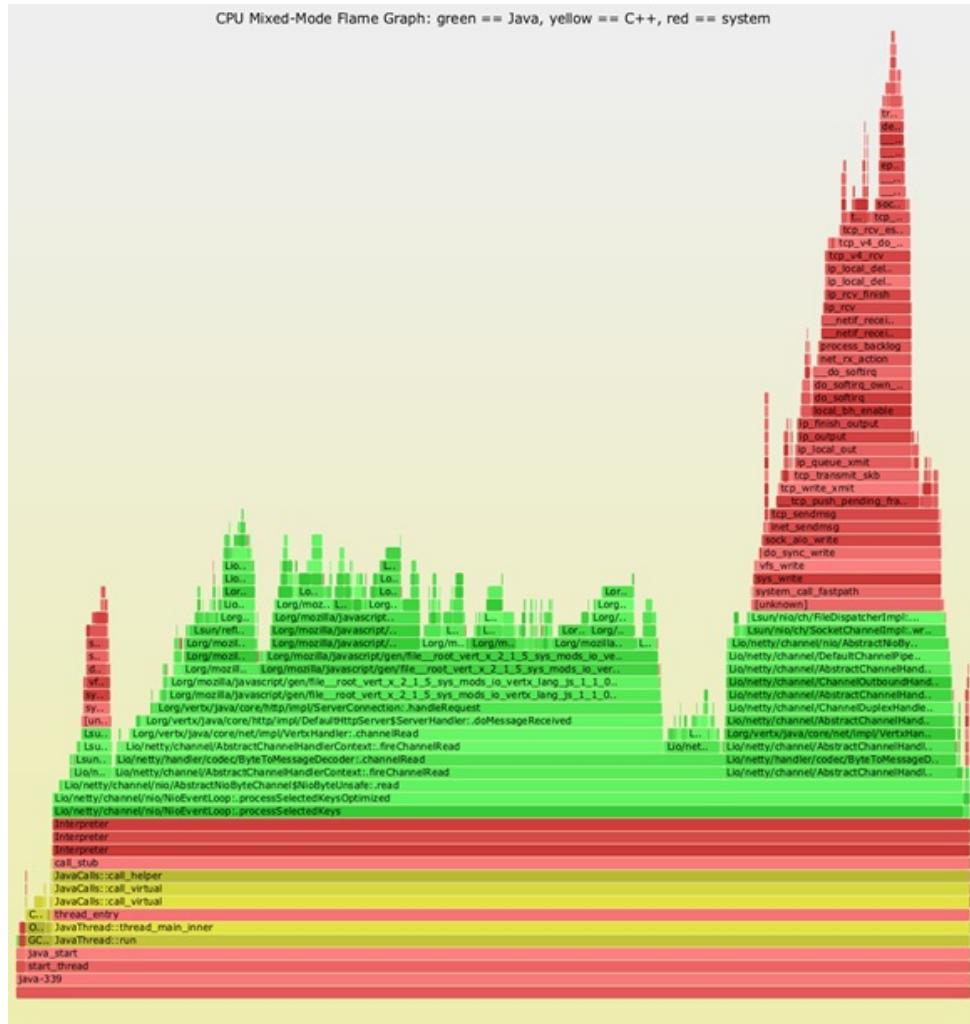


Figura 2. Ejemplo de FlameGraph. Fuente: The Netflix Tech Blog, 2015.

## 10.3. Herramientas de visualización

### Grafana

Casi todo el mundo ha oído hablar de Grafana y, tal vez, muchos de ustedes incluso lo están utilizando en la actualidad. Esta herramienta fue creada por Torkel Ödegaard. En sus comienzos era un sistema de visualización para Kibana, al que Torkel modificó y posteriormente convirtió en Grafana.

**El objetivo de Grafana es representar los datos de monitorización de una forma más usable y agradable.** Una de sus ventajas más notables es que puede recopilar datos nativos de Graphite, Elasticsearch, OpenTSDB, Prometheus y InfluxDB. También existe una versión Enterprise que incluye complementos para que soporte más fuentes de datos, pero no hay razón por la que estos complementos no puedan ser también *open source*, o para que cualquiera pueda escribir nuevos complementos.

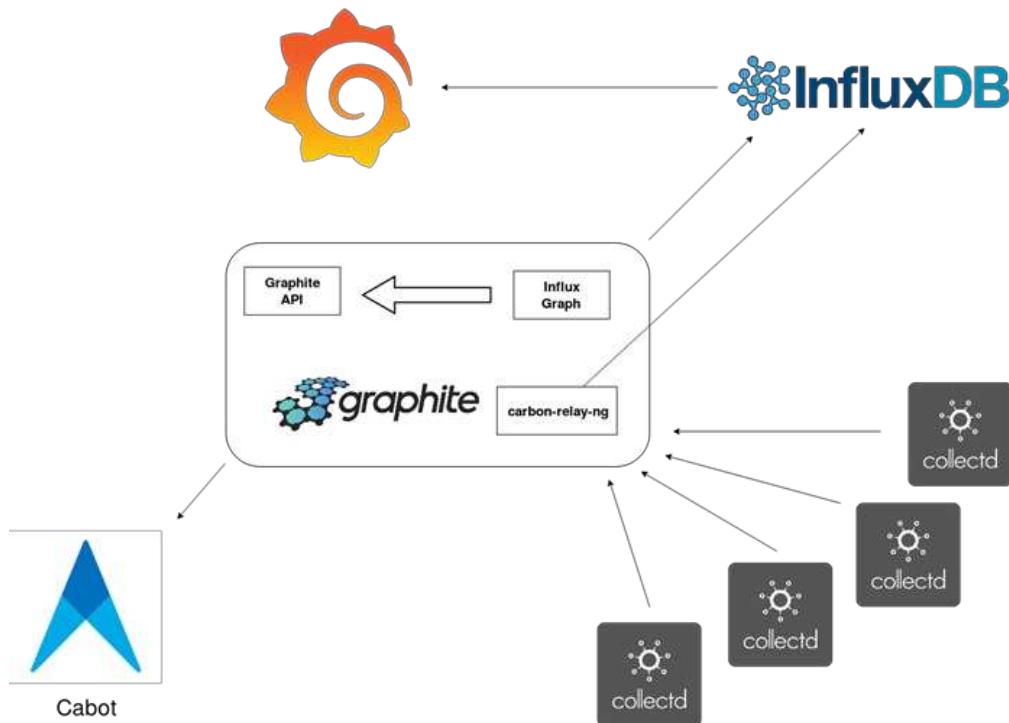


Figura 3. Sistema de alertas con Cabot e Influx DB. Fuente: Hackerearth, s.f.

Grafana está basado en la web, así que cualquiera puede acceder a la información, aunque también tiene sistemas de autenticación. Tiene la capacidad de proporcionar información relevante en apenas un vistazo porque integra muchos tipos diferentes de visualizaciones. Adicionalmente, Grafana ha empezado a incluir herramientas adicionales: ahora permite configurar alertas visualmente. Eso significa que puedes mirar un gráfico (que muestre dónde se debería haber activado una alerta debido a alguna degradación del sistema), hacer «clic» en el gráfico donde quieras que se active la alerta y luego indicarle a Grafana dónde debe enviar la notificación. Esto es tremadamente práctico y, aunque no necesariamente reemplaza a una plataforma de alertas, ciertamente ayuda a mejorar el servicio.



Figura 4. Ejemplo de Panel de Control con Grafana. Fuente: Wilson, 2021.

Grafana también ha introducido más funciones de colaboración con el tiempo: los usuarios pueden compartir paneles (*dashboards*), una característica muy popular entre las herramientas colaborativas de Software.

### StatsAgg

**StatsAgg** es una plataforma de alertas (y también una plataforma de agregación de métricas) creada por Pearson, que más allá de ser una editorial, también tiene presencia en la web y una sólida relación con O'Reilly Media.

Su aportación más importante está relacionada con la colaboración y son, nada más y nada menos, que las anotaciones. Estas permiten a un usuario agregar contexto, a parte de un gráfico, y los demás usuarios pueden usar este contexto para comprender mejor el sistema. Esta es una característica de gran valor cuando un equipo está atravesando un incidente y la comunicación es crítica, ya que el contexto ayuda a que el conocimiento se comparta de forma rápida con todo el equipo.

## Vizceral

Netflix creó Vizceral para **comprender más a fondo sus patrones de tráfico y, en concreto, mejorar los problemas que genera la commutación de tráfico con errores**. A diferencia de Grafana, que es una herramienta más general, Vizceral es muy específica para los casos de uso de Netflix y si bien no es probable que sea la mejor elección para todo el mundo, sí es un buen ejemplo de las ventajas que proporcionan las herramientas de visualización.

## 10.4. Herramientas de rastreo distribuido

**Las herramientas de rastreo distribuido permiten dar seguimiento a una búsqueda en los logs a través de un sistema de software distribuido en múltiples aplicaciones, servicios y bases de datos, así como capas intermedias y otras piezas de software.** Esto permite profundizar en detalle sobre todo lo que sucede dentro de un sistema software.

Estas herramientas ofrecen representaciones gráficas que muestran cuánto tiempo ha tardado una solicitud en un paso determinado. De esta forma, el usuario puede determinar dónde está el problema que hace que el sistema experimente latencias o bloqueos. Es decir, en lugar de considerar al sistema como un árbol de búsqueda con muchas conexiones, en el momento en que las solicitudes comienzan a fallar, los desarrolladores y el personal de operaciones pueden ver exactamente dónde están los problemas (desde el principio). Esta información también puede revelar dónde podrían originarse los cambios de rendimiento tras un despliegue. Siempre es una buena estrategia tener herramientas que permitan identificar regresiones, automáticamente alertando de forma temprana sobre aquellos comportamientos anómalos antes de que impacten a los clientes.

### ¿Cómo funciona el rastreo?

**El rastreo se basa en un identificador único que se asigna a cada solicitud y que, en general, se inyecta en los encabezados.** Esta identificación registra de manera única, la cual se denomina «rastro». Este es la representación abstracta general de toda la transacción y se compone de tramos. Estos tramos representan al trabajo real que se realiza, como cuando se llama a un servicio o se hace una solicitud a la base de datos y poseen un identificador único. A su vez, estos tramos pueden crear tramos posteriores llamados tramos secundarios, y los tramos hijos pueden tener múltiples padres.

Una vez que una transacción (o rastreo) ha seguido su curso, se puede buscar en cada capa y ver de forma global toda la ejecución. Hay múltiples herramientas que pretenden ofrecer esta funcionalidad y hablaremos de algunas de ellas a continuación.

## API OpenTracing

OpenTracing es una especificación que surgió de Zipkin con el **objetivo de proveer compatibilidad multiplataforma**. Ofrece un proveedor neutro a nivel de API para agregar rastreo a las aplicaciones e integrar esos datos en sistemas de rastreo distribuido. Zipkin, Jaeger y AppDash son ejemplos de herramientas de código abierto que han adoptado esta especificación, pero incluso herramientas propietarias como Datadog e Instana también lo están haciendo. Se prevé que su expansión continúe hasta que sea un verdadero estándar de facto.

## Herramientas disponibles

### Zipkin

Desarrollado por Twitter, Zipkin está escrito en Java y puede usar Cassandra o Elasticsearch como *backends* escalables, lo que proporciona opciones válidas para, prácticamente, cualquier tipo de empresa. **El sistema consta de clientes, recolectores, un servicio de consulta y una interfaz de usuario web**. Los datos recopilados por cada reportero (*reporter*, cliente) se transmiten de forma síncrona a los recolectores. Los recolectores los almacenan en la base de datos y la interfaz de usuario web presenta los datos al usuario final en un formato consumible. La entrega de datos a los recolectores se puede realizar en tres formas diferentes: HTTP, Kafka y Scribe.

La comunidad Zipkin también ha creado Brave, una implementación de cliente Java compatible con Zipkin sin dependencias. Sin embargo, hay muchas otras implementaciones y Zipkin es compatible con el estándar OpenTracing.

## Jaeger

Jaeger es un proyecto más reciente, creado por Uber Technologies. Está escrito en Go, por lo que no tiene problemas de dependencias específicas que tienen que ser instaladas como requisito previo en un servidor, ni sufre la sobrecarga de requerir un intérprete o una máquina virtual. Al igual que Zipkin, Jaeger también es compatible con Cassandra y Elasticsearch como almacenamiento. A su vez, también es totalmente compatible con el estándar OpenTracing (como ocurre con Zipkin).

La arquitectura de Jaeger es similar a la de Zipkin, con **clientes, recolectores, un servicio de consultas y una interfaz de usuario web, pero también tiene un agente en cada host que agrega localmente los datos**. El agente recibe los datos a través de una conexión UDP, los agrupa y los envía a un recolector. Este recibe los datos de forma resumida, lo que, en cierta forma, implica un ahorro. El servicio de consulta puede acceder al almacén de datos directamente y proporcionar esa información a la interfaz de usuario web.

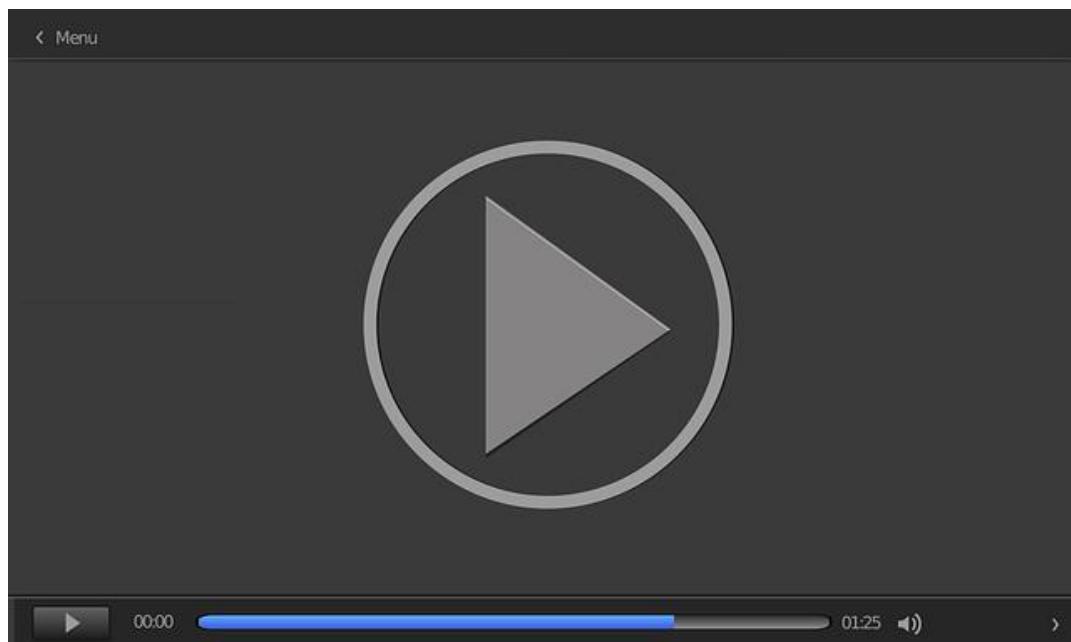
Por defecto, un usuario no obtendrá todos los rastros del Jaeger. El sistema muestrea 0,1 % (uno en 1000) de las trazas que pasan por cada cliente (el hecho de tener que mantener y transmitir todos los rastros sería un poco abrumador para la mayoría de los sistemas). Sin embargo, este porcentaje se puede aumentar o disminuir a través de la configuración de los agentes. **Jaeger utiliza el método de muestreo probabilístico, que trata de adivinar si un nuevo rastro debe ser muestreado o no**. El muestreo adaptativo está en su hoja de ruta, pero podrás ayudar a mejorar el algoritmo de muestreo agregando contexto adicional para que tome las decisiones más adecuadas.

## AppDash

AppDash es un sistema de rastreo distribuido escrito en Go, como Jaeger. Fue creado por Sourcegraph y está basado en Dapper de Google y Zipkin de Twitter. Al

igual que Jaeger y Zipkin, AppDash es compatible con el estándar OpenTracing, pero requiere un componente adicional (lo que añade riesgo y complejidad). **A alto nivel, la arquitectura de AppDash consiste, principalmente, en tres componentes: un cliente, un recolector local y un control remoto recolector.** AppDash proporciona implementaciones en Python, Golang y Ruby.

En el vídeo *Open source* se define qué es Open source y por qué es un magnífico ejemplo de colaboración.



---

Accede al vídeo:

<https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=376b7d8d-52af-4567-a68a-ad78013d29c1>

---

## 10.5. Referencias bibliográficas

Hackerearth. (s.f.). *Monitoring and alert system using Graphite and Cabot.*

<http://engineering.hackerearth.com/2017/03/21/monitoring-and-alert-system-using-graphite-and-cabot/>

The Netflix Tech Blog (2015, junio 24). Java in Flames. *The Netflix Tech Blog.*

<http://engineering.hackerearth.com/2017/03/21/monitoring-and-alert-system-using-graphite-and-cabot/>

Wilson, B. (2021, julio 2). 16 Best Container Orchestration Tools and Services.

*Devopscube.* <https://devopscube.com/docker-container-clustering-tools/>

## Tutorial de Grafana

Albert Coronado. (2018, julio 23). *Analítica y monitorización con la plataforma Grafana [Vídeo]*. YouTube. <https://www.youtube.com/watch?v=g8I9i1dKqYI>

Si bien este tema solo pretende dar una visión general de las herramientas disponibles sin entrar en la instalación u operación de las herramientas concretas, Grafana es una de las herramientas líderes en el mercado y este recurso te permitirá saber más.

1. Selecciona la afirmación que mejor describe a AppDash.

  - A. Es una forma de visualizar de forma rápida cualquier métrica.
  - B. AppDash es un sistema de rastreo distribuido escrito en Go.
  - C. A y D son incorrectas.
  - D. Es un tipo de dashboard.
  
2. Selecciona la afirmación que mejor describe a Zipkin.

  - A. Es una forma de visualizar de forma rápida cualquier métrica.
  - B. Fue uno de los primeros sistemas de rastreo distribuido y fue desarrollado por Twitter.
  - C. A y D son incorrectas.
  - D. Es una herramienta de compresión de logs.
  
3. Selecciona la afirmación que mejor describe a los Flame Graphs.

  - A. Son una herramienta de visualización de múltiples métricas que recibe ese nombre por su aspecto gráfico.
  - B. Son una forma de visualizar de forma rápida cualquier métrica.
  - C. A y D son incorrectas.
  - D. Son un tipo de mapa de calor.
  
4. Selecciona la afirmación que mejor describe Cabot.

  - A. Es una herramienta de visualización de métricas.
  - B. Es una forma de visualizar de forma rápida cualquier métrica.
  - C. A y D son correctas.
  - D. No es un concepto pertinente para este tema.

5. Selecciona la afirmación que mejor describe Bosun.

  - A. Es una herramienta de agregación de métricas, muy similar a Prometheus.
  - B. Es una forma de visualizar de forma rápida una métrica concreta.
  - C. A y D son falsas.
  - D. No es un concepto pertinente para este tema.
6. Selecciona la afirmación que mejor describe el concepto de «medidor».

  - A. Es una herramienta.
  - B. Es una forma de visualizar de forma rápida una métrica concreta.
  - C. A y D son correctas.
  - D. No es un concepto pertinente para este tema.
7. Selecciona la afirmación que mejor describe un mapa de calor.

  - A. Es una herramienta que permite saber la temperatura de los servidores.
  - B. Es extremadamente útil en el caso de querer representar información proveniente de histogramas.
  - C. A y D son correctas.
  - D. El mapa de calor está definido sobre OpenTracing.
8. Selecciona la afirmación que mejor describe OpenTracing.

  - A. Es una herramienta.
  - B. Es un estándar de las API.
  - C. A y D son correctas.
  - D. ELK definió OpenTracing.

- 9.** Selecciona la afirmación que mejor describe a la herramienta FlameGraf.
- A. Es una de las herramientas de visualización desarrollada por Netflix.
  - B. Se trata de la detección de situaciones críticas en los gráficos de visualización.
  - C. A y D son correctas.
  - D. Es un tipo de gráfico que permite exponer información muy detallada normalmente exponiendo información de CPU y memoria.
- 10.** Selecciona la afirmación que mejor describe Grafana.
- A. Es una de las herramientas de visualización más usadas.
  - B. Es una metodología para mostrar logs y alarmas.
  - C. Es una herramienta que permite reunir y representar en un mismo lugar información de diferentes fuentes.
  - D. A y C son correctas.