1. Despliegue de aplicaciones web

Vamos a empezar aprendiendo qué significa el concepto de **despliegue** *(deployment)*. Para ello, vamos a recordar en la figura siguiente el ciclo de vida clásico de un desarrollo de software, ya que las aplicaciones web requieren ser desplegadas. Serán las tareas que realizaremos para finalizar la fase de implementación o desarrollo y trasladar nuestra aplicación web probada y terminada a la siguiente fase de implantación o producción. Para ello, hemos de instalar y configurar los diferentes servicios de red o web que se vayan a utilizar por parte de nuestra aplicación web.

Este despliegue de software es una tarea repetitiva con muchas acciones por realizar que, si no están automatizadas, pueden llevarnos a introducir errores humanos. Si no se actualizan todas las dependencias o copiamos los archivos de configuración adecuadamente en el despliegue de forma precisa, la nueva aplicación web desplegada no llegará a funcionar correctamente. Por lo tanto, nuestro objetivo será automatizarlos lo máximo posible, sin olvidar efectuar una revisión final por nuestra parte de que todo está funcionando correctamente.

Pero primero vamos a ver cuáles son las características de una aplicación web para desplegar.

1.1. Introducción al despliegue de aplicaciones web

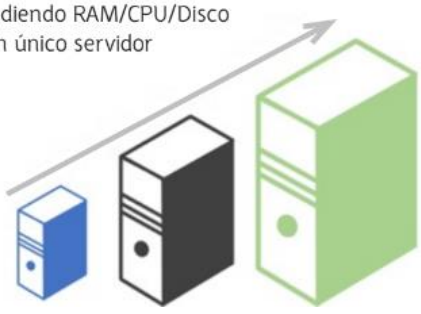
Una aplicación web va a ejecutarse sobre uno o varios servidores web. Inicialmente, estos eran físicos o reales, es decir, utilizaban todo el equipo para esta tarea. En la actualidad, estos equipos van a ser virtualizados completamente o se utilizarán contenedores para consumir los mínimos recursos posibles.

Una de las peculiaridades de las aplicaciones web es que vamos a querer dar servicio concurrente a todos nuestros usuarios, que pueden ser miles o millones, en cuyo caso deberemos trabajar con sistemas distribuidos que formen un clúster de equipos coordinados, lo que aumenta la complejidad del despliegue.

Por eso también deberemos enfrentarnos a decisiones de escalabilidad del sistema decidiendo que su crecimiento sea vertical (aumentando los recursos del servidor con más RAM, más capacidad de disco o más *cores*, o núcleos de microprocesadores) u horizontal (aumentando el número de servidores iguales) para producir mayor tolerancia a fallos y poder equilibrar la carga entre todos ellos. Esto se ve gráficamente en la siguiente figura.

Escalado vertical

Aumentar la capacidad añadiendo RAM/CPU/Disco a un único servidor



Escalado horizontal

Aumentar la capacidad añadiendo servidores

1.2. Alojamiento interno *(in-house)*

El alojamiento interno *(in-house)* es el modelo clásico en el que una organización o empresa aloja todos sus servidores, aplicaciones y servicios dentro de sus instalaciones. Esto implica instalar y mantener todos los recursos de hardware necesarios. Por ejemplo:

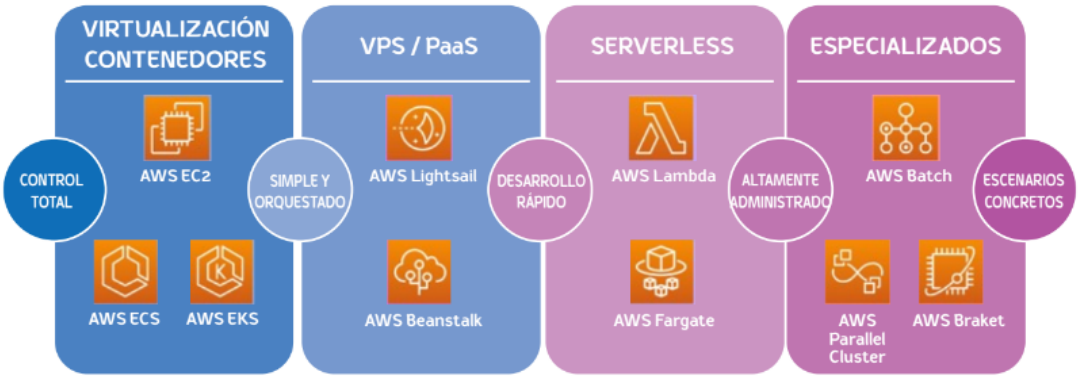
- Servicio DNS para que todos nuestros equipos puedan utilizar sus nombres de dominios y poder traducirlos a las IP privadas de nuestra intranet.
- Servicio de directorio activo con validación de usuarios centralizada con SAMBA o LDAP/OpenLDAP para que nuestros usuarios puedan usar todas las aplicaciones web de nuestra intranet.
- Servicios web generalistas o específicos de tecnologías como Java para las aplicaciones web propias o de terceros.
- Servicios de transferencia de archivos (FTP) que nos permiten subir o bajar archivos grandes o muchos pequeños rápidamente.
- Servicios de correo electrónico con los diferentes protocolos de envío (SMTP) o de recepción (POP o IMAP).
- Servicios de conexión remota por terminal (SSH) o por GUI (VNC, RDP, etc.).

Podríamos tener un campus de formación con Moodle, un CMS con WordPress o nuestras propias aplicaciones web.

Se puede considerar que tendríamos una nube privada con servidores de virtualización, como podría ser ProXmox VE u otros para dar soporte a todos estos servicios con máquinas o contendores virtualizados. Esta opción cada vez se utiliza menos, ya que tiene un elevado coste de implantación y mantenimiento, además de requerir una infraestructura de sistemas de respaldo en caso de caída de Internet o de la red eléctrica. Por ello, la tendencia actual es cada vez más trasladar total o parcialmente esta infraestructura propia a una nube pública para que la computación en la nube se encargue de administrar parte de la infraestructura que vayamos a contratar.

1.3. Computación en la nube

En 2006, surgió el concepto de "computación en la nube" *(cloud computing)*. Fue la división de servicios web de Amazon (AWS) la primera en ofrecer un servicio de virtualización de servidores a gran escala llamado EC2 *(Elastic Compute Cloud)* y otro de almacenamiento masivo de objetos llamado S3 *(Simple Storage Service)* con el objetivo de ofrecer a los clientes con CPD en propiedad un servicio similar, pero virtualizado y en pago por uso, con las ventajas de escalabilidad, ausencia de inversión inicial y menor coste. En la siguiente figura se puede ver el panorama general clasificado de los servicios de computación de AWS.



Desde entonces, el uso extendido de aplicaciones web y de móviles, servicios multimedia masivos —como los servicios de *streaming* por suscripción de películas—, series o música y el procesamiento de datos continuo relacionado con el *Big Data* han experimentado una gran evolución en los servicios de computación, más allá de ofrecer máquinas virtuales que sustituyan a las propias de un CPD.

Veamos en qué partes se dividen los servicios de computación en la nube donde deberemos desplegar nuestras aplicaciones dependiendo de qué queramos encargarnos, desde administrar todo el equipo (IaaS) a solo encargarnos de nuestra aplicación web (SaaS) o contar con una opción intermedia para desarrollar también desde la nube (PaaS). Veamos cada una de ellas.

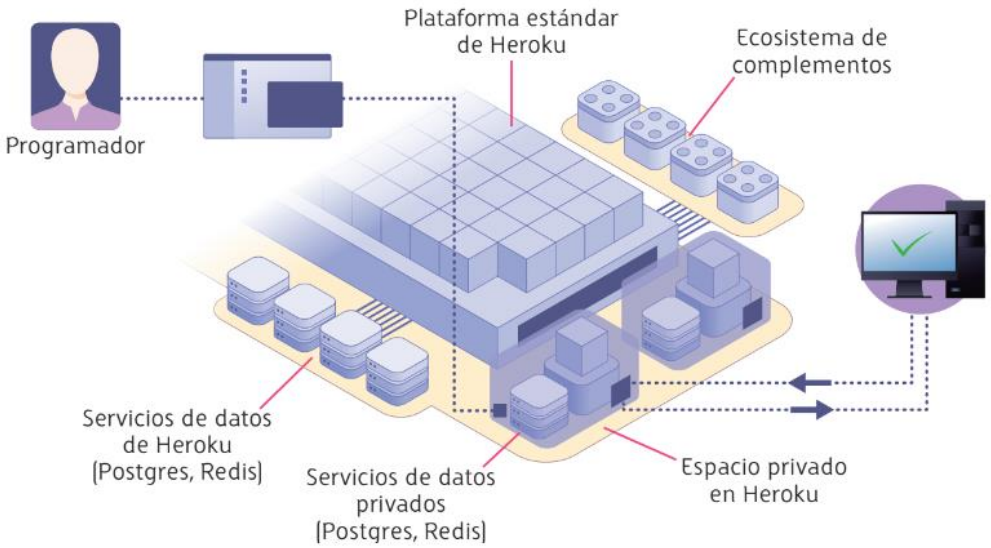
A. INFRAESTRUCTURA COMO SERVICIO (IaaS)

Algunos de los planes de hosting tradicionales funcionan sobre este nivel, por ejemplo, el *cloud hosting* o el de servicio dedicado que puede estar contratado en cualquiera de los IaaS que existen hoy en día.

La diferencia es que, en el *cloud hosting,* el proveedor del servicio nos facilita una consola de administración que nos ayuda en la gestión del dominio y de nuestras webs, mientras que, en el servicio dedicado, se nos dará acceso a una máquina sin ninguna ayuda y nosotros seremos los que nos encargaremos de hacer todo lo que queramos, por lo que tendremos que ser unos buenos administradores de sistemas y tendremos que dominar el mundo de la computación en la nube. Los tres principales proveedores de infraestructura como servicio (IaaS) son AWS, Azure y Google.

B. PLATAFORMA COMO SERVICIO (PaaS)

En este modelo, el desarrollador se olvida del hardware y del sistema operativo y se preocupa solo de la aplicación y los datos por desplegar. Típicamente, se utiliza para desplegar aplicaciones web, bases de datos y *middleware*. Servicios como el de la arquitectura Heroku (en la siguiente figura) nos facilitarán el despliegue de nuestras aplicaciones web, que podremos subir a este servicio en la nube. Desde su panel de control *(dashboard)* o utilizando su CLI, podremos desplegar, probar y, finalmente, publicar nuestras aplicaciones web sobre el servicio SaaS final donde vayamos a explotarlas.



C. SOFTWARE COMO SERVICIO (SaaS)

En esta tipología, el proveedor del servicio en la nube nos da soporte a todos los niveles, incluso sobre la aplicación que vamos a utilizar con él. El usuario se limita a usar la aplicación contratada y se olvida de lo demás. Es lo opuesto a tenerlo en nuestras instalaciones *(in-house)*, donde nos encargamos de gestionar todo nosotros.

Podemos utilizarlo sin tener que saber programar. Por ejemplo, contratando el sistema gestor de contenidos (CMS) WordPress, el planificador de recursos de empresa (ERP) Odoo o el software de comercio electrónico *(e-commerce)* PrestaShop. Para ello, solo tendremos que contratar alguno de estos servicios con una de las muchas empresas de hosting. Lo mejor es que conozcamos las diferentes opciones que podemos contratar y el coste de cada una de ellas.