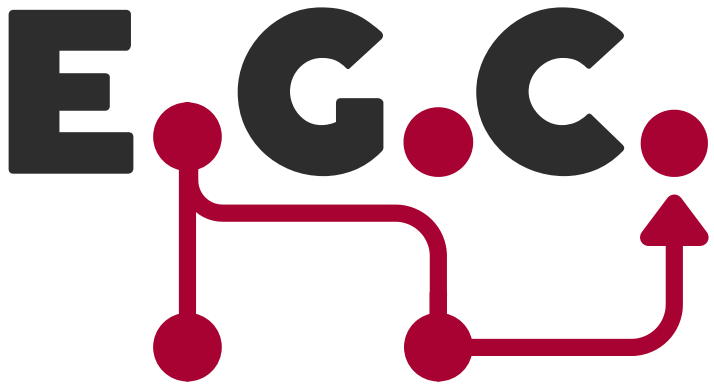


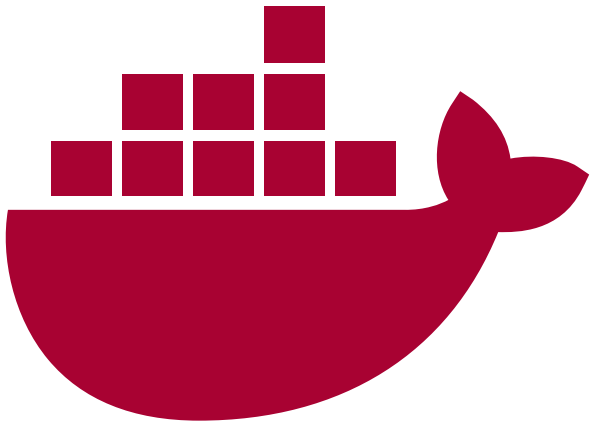
Grado en Ingeniería Informática - Ingeniería del Software

Evolución y Gestión de la Configuración

Escuela Técnica Superior de Ingeniería Informática

Práctica 5

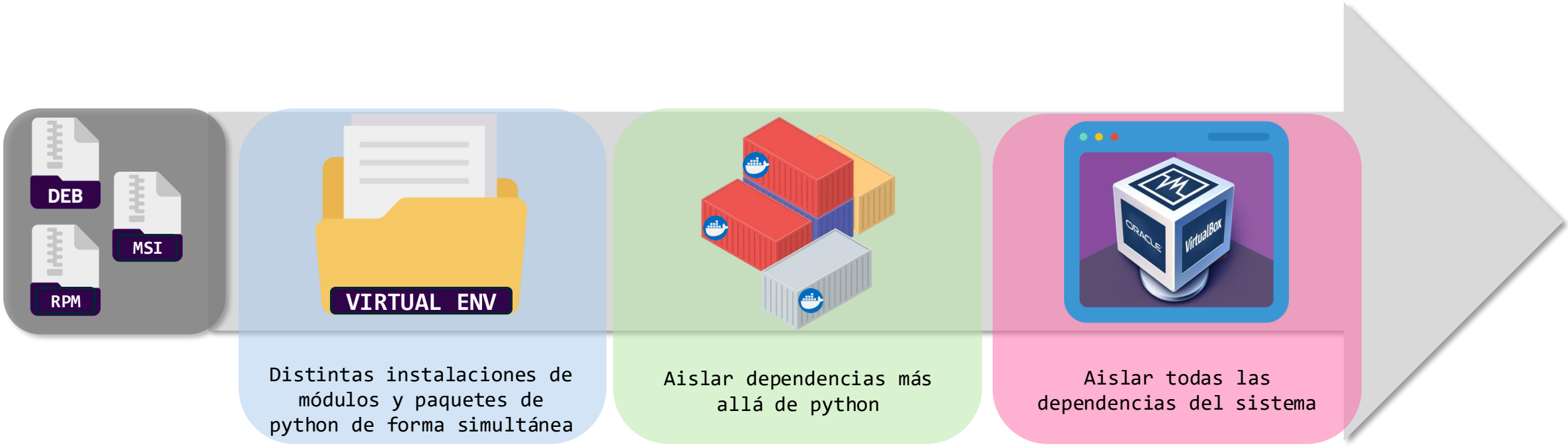
Contenedores, dev-containers aislamiento



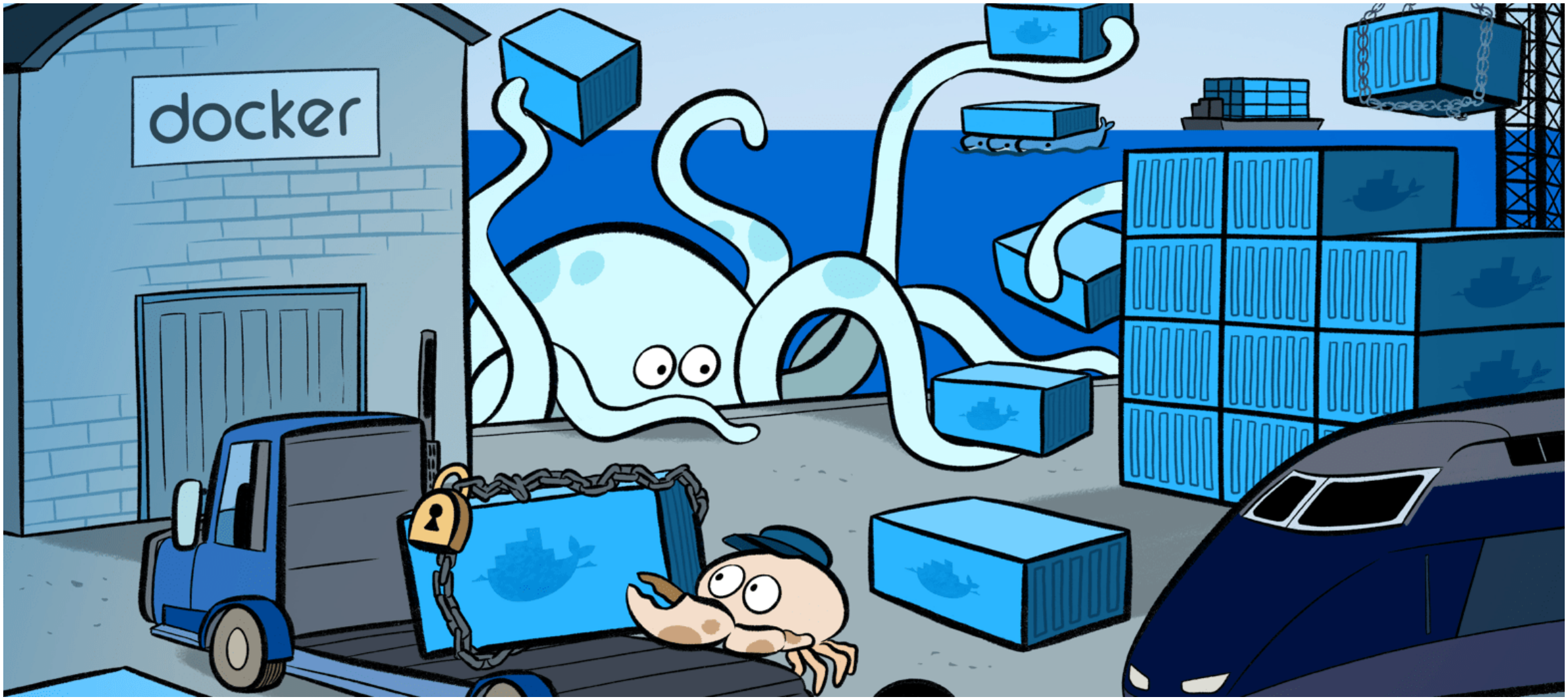
1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

1. **Introducción a Docker y DockerHub**
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

1. Introducción a Docker



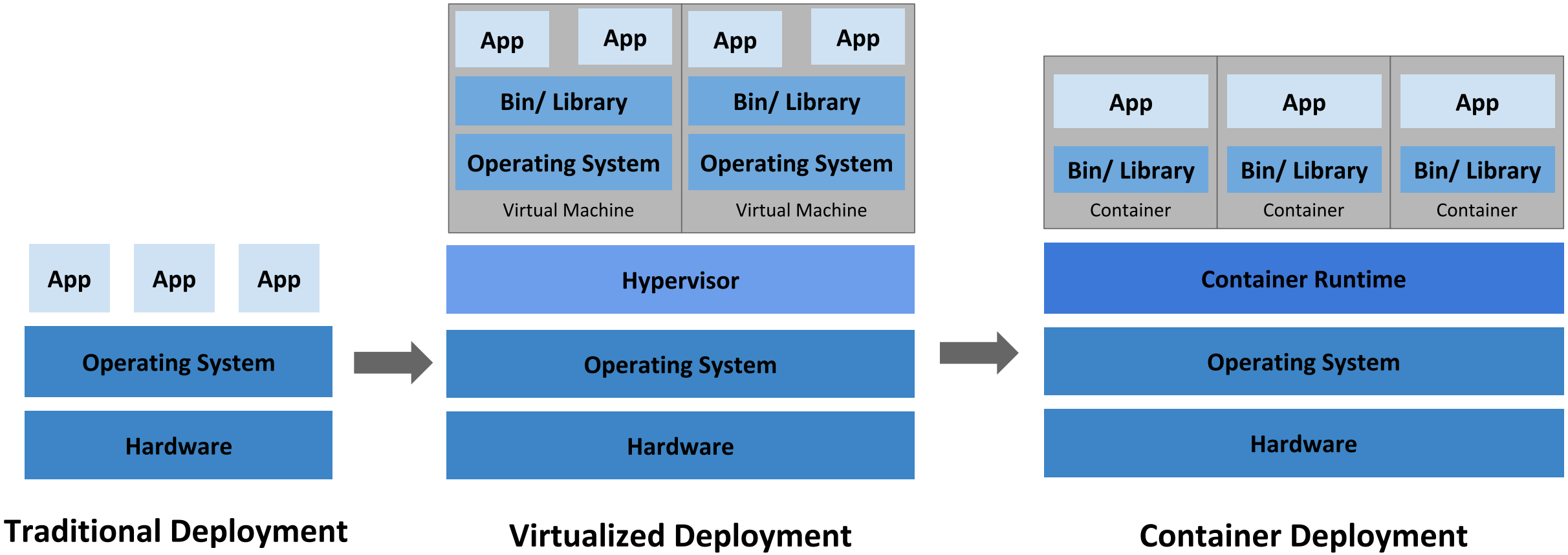
1. Introducción a Docker



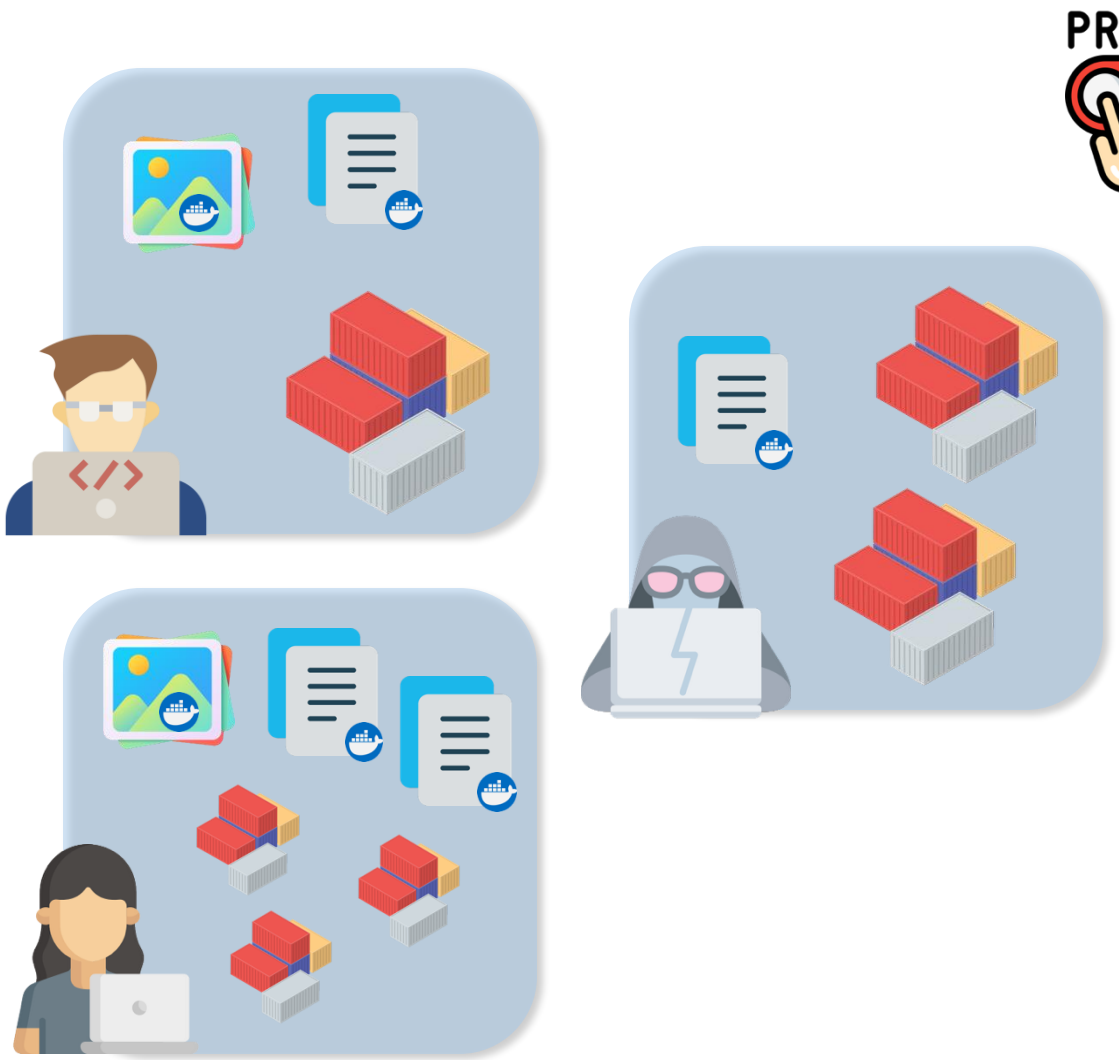
1. Introducción a Docke

1. Introducción a Docker

¿Por qué usar contenedores si ya tenemos máquinas virtuales?

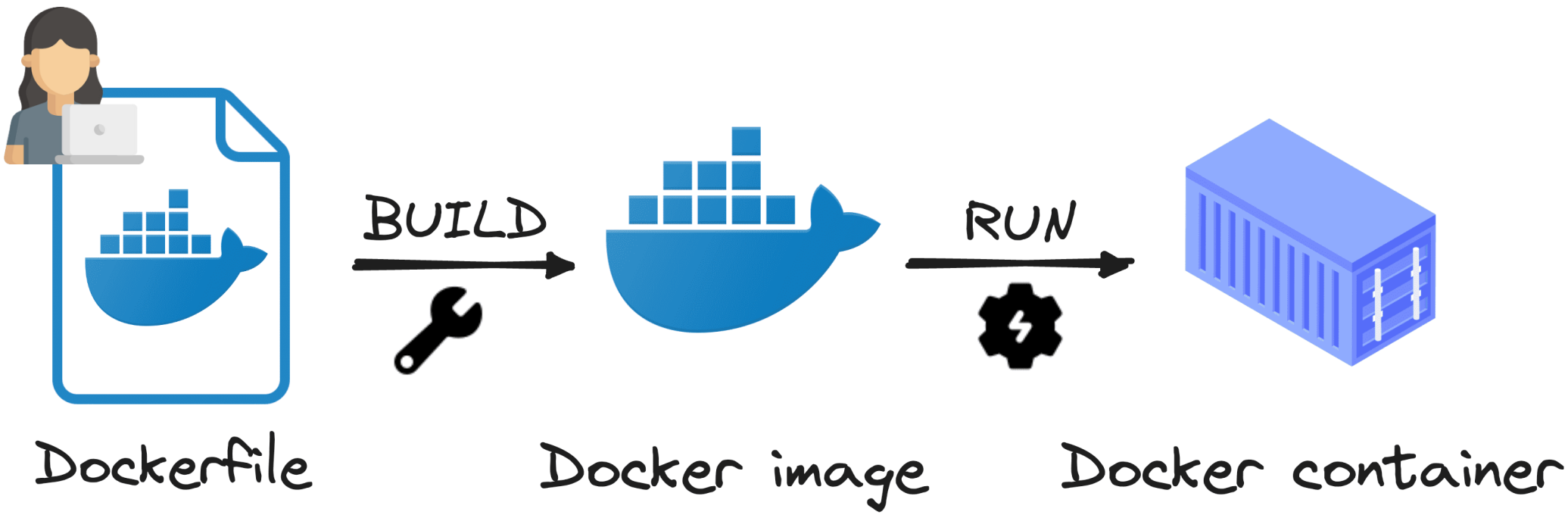


1. Introducción a DockerHub

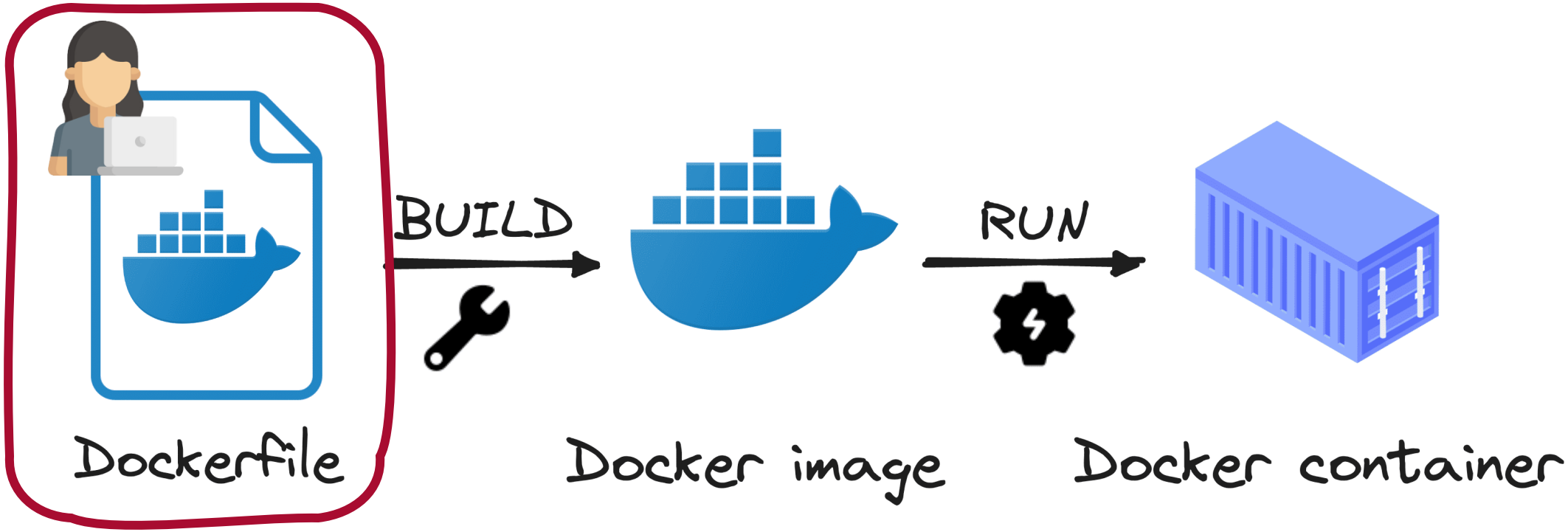


1. Introducción a Docker y DockerHub
2. **Imágenes**
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

2. Imágenes

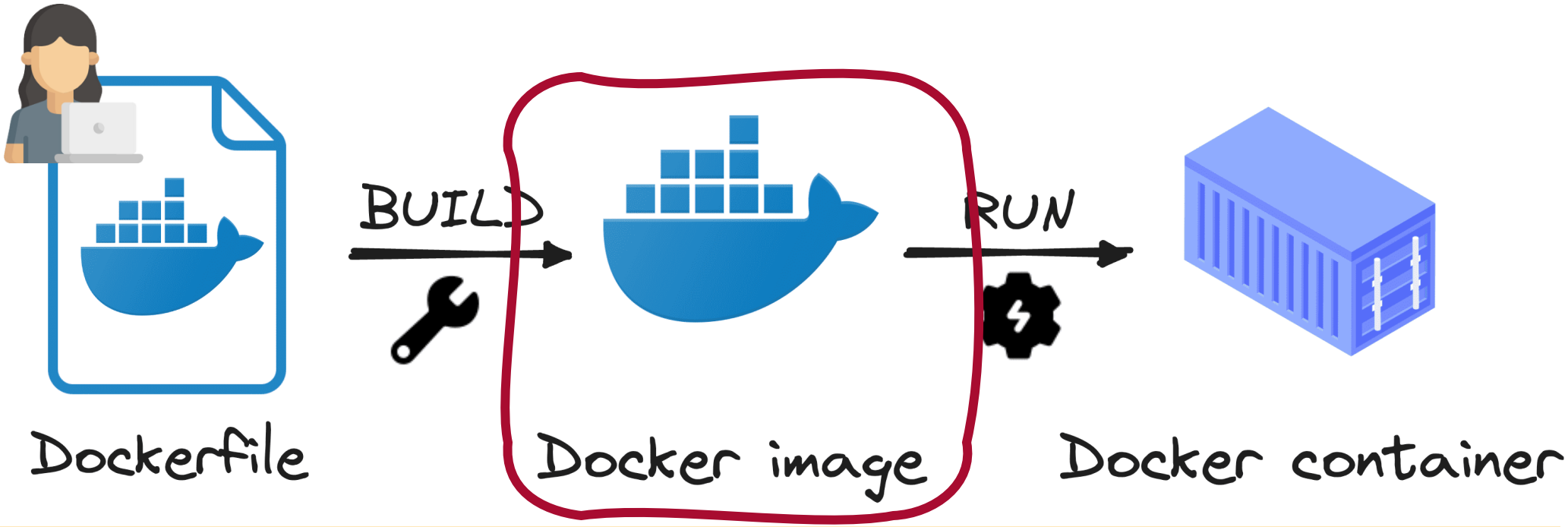


2. Imágenes



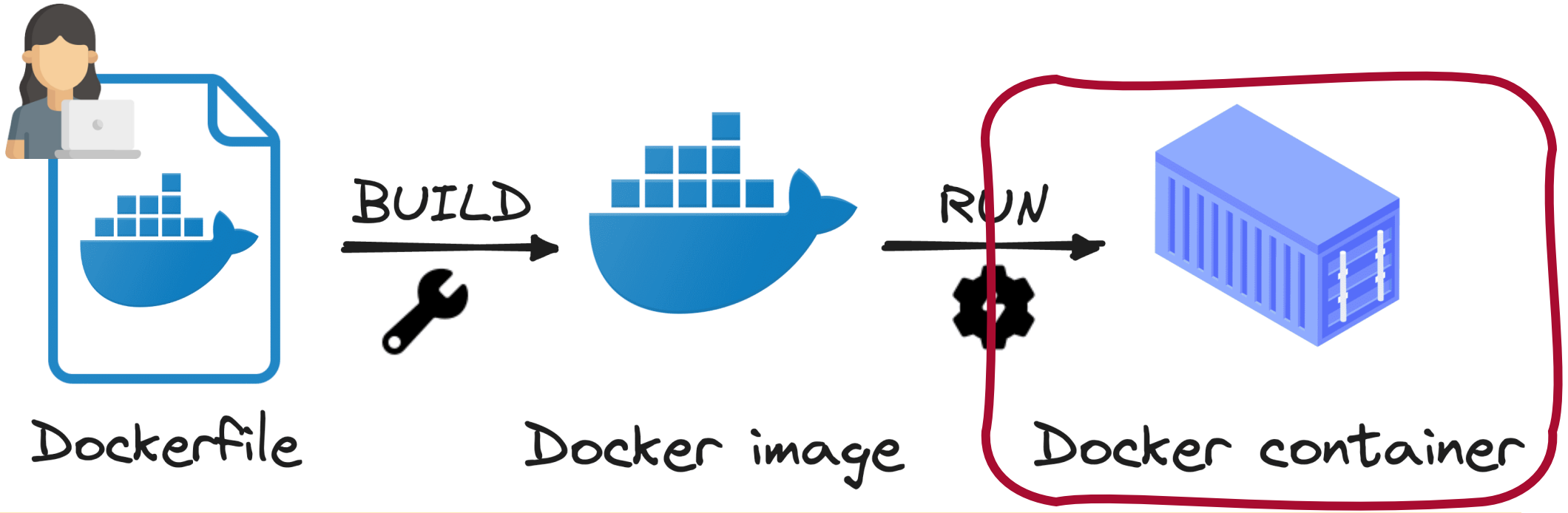
Archivo de texto que contiene las instrucciones para construir una imagen de Docker de forma automática

2. Imágenes



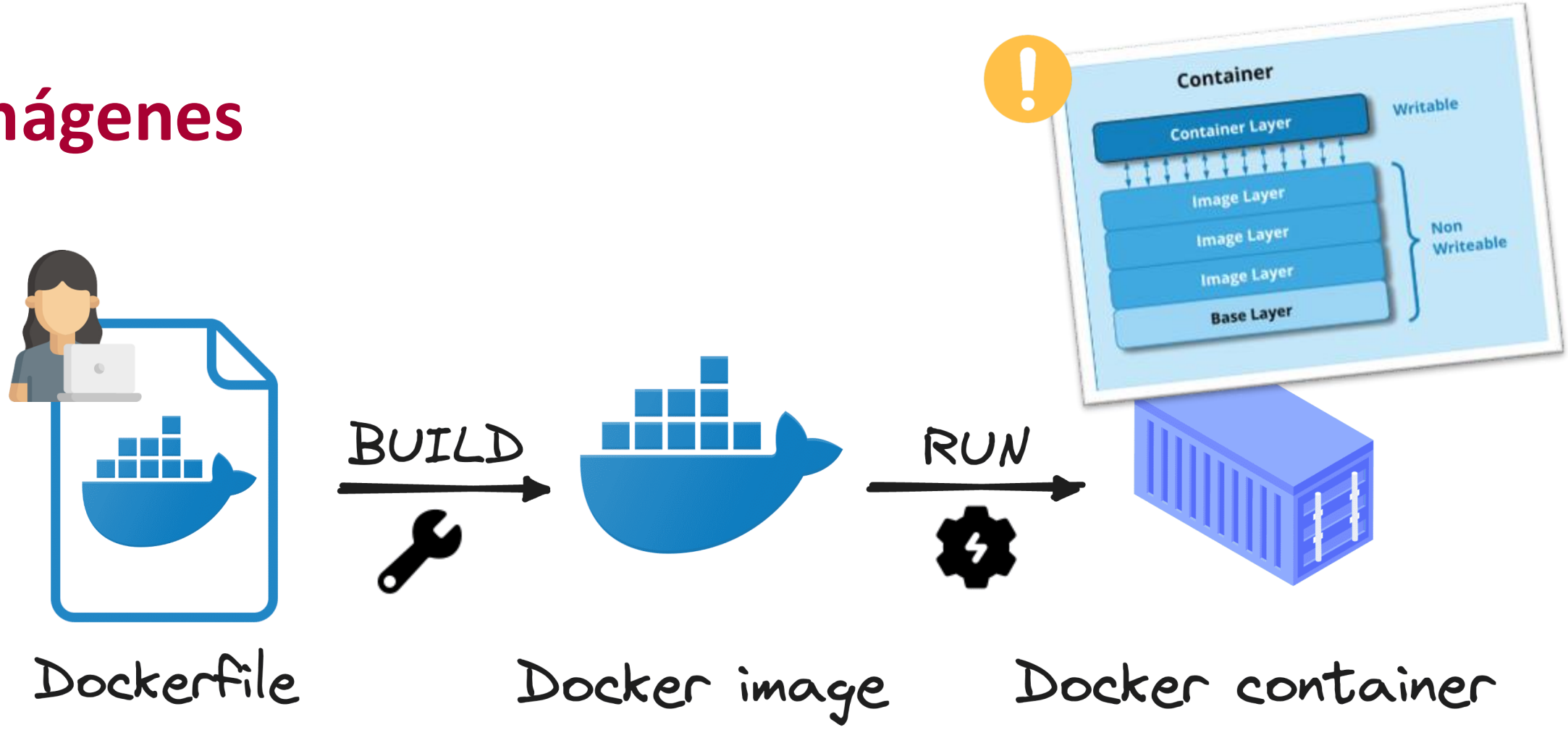
Plantillas de solo lectura que contienen el código y las dependencias necesarias para ejecutar aplicaciones

2. Imágenes



Instancia en ejecución de una imagen de Docker, es decir, un entorno aislado donde esa imagen cobra vida y la aplicación se ejecuta

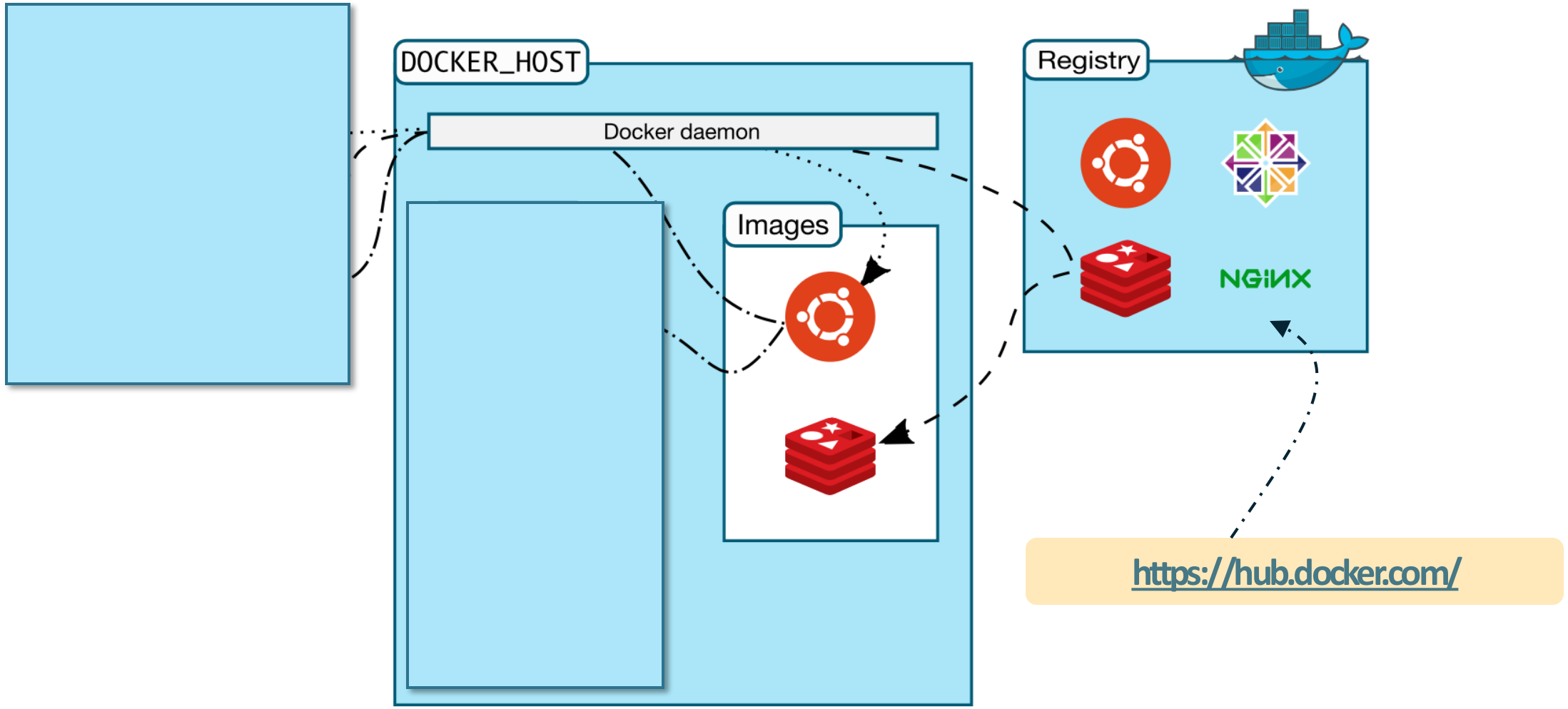
2. Imágenes



Las capas son partes apiladas de una imagen que se reutilizan para ahorrar espacio y acelerar las construcciones

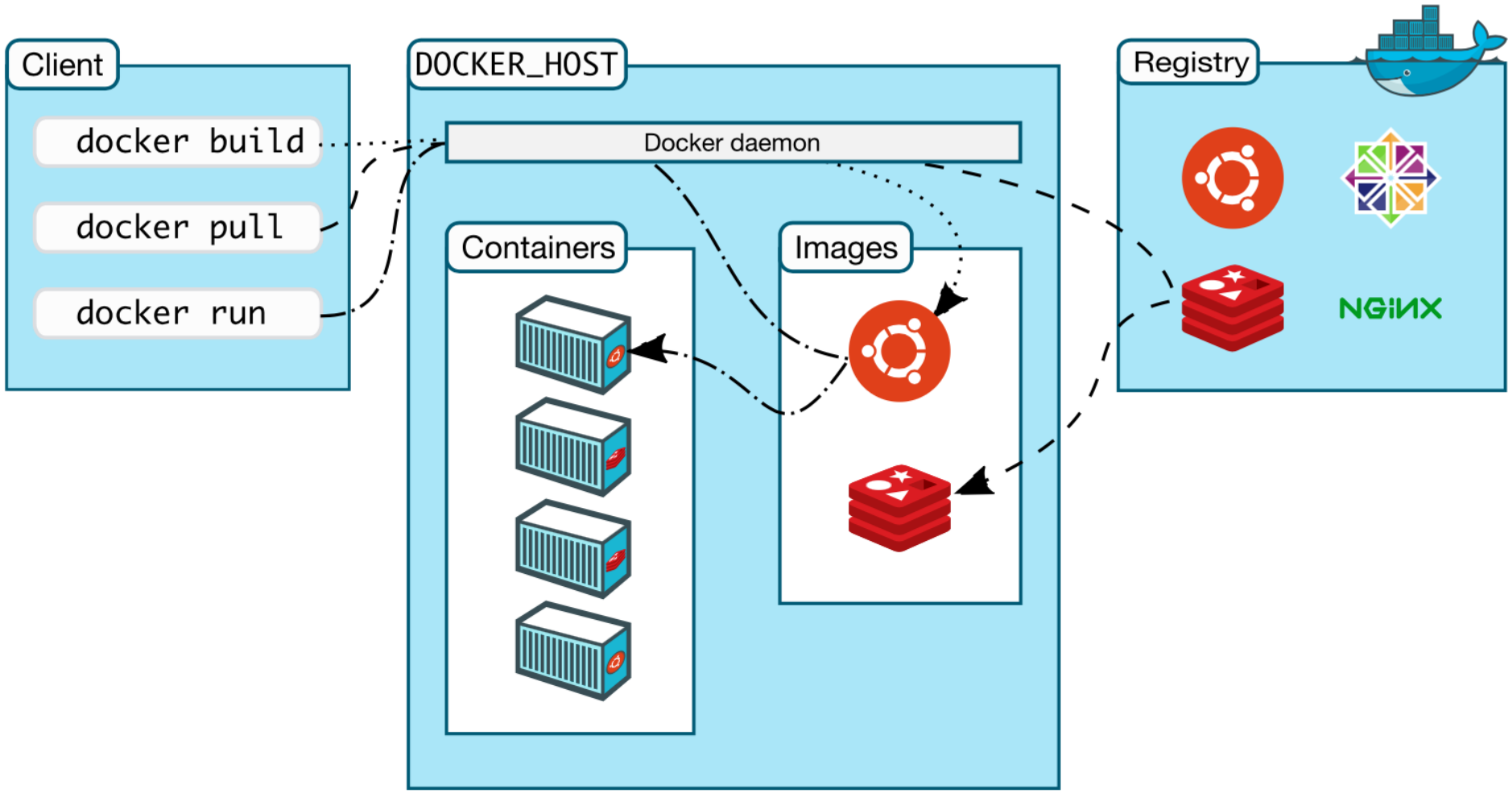
2. Imágenes

Y si no escribo un Dockerfile, ¿de dónde salen las imágenes?



1. Introducción a Docker y DockerHub
2. Imágenes
3. **Contenedores**
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

3. Contenedores



3. Contenedores

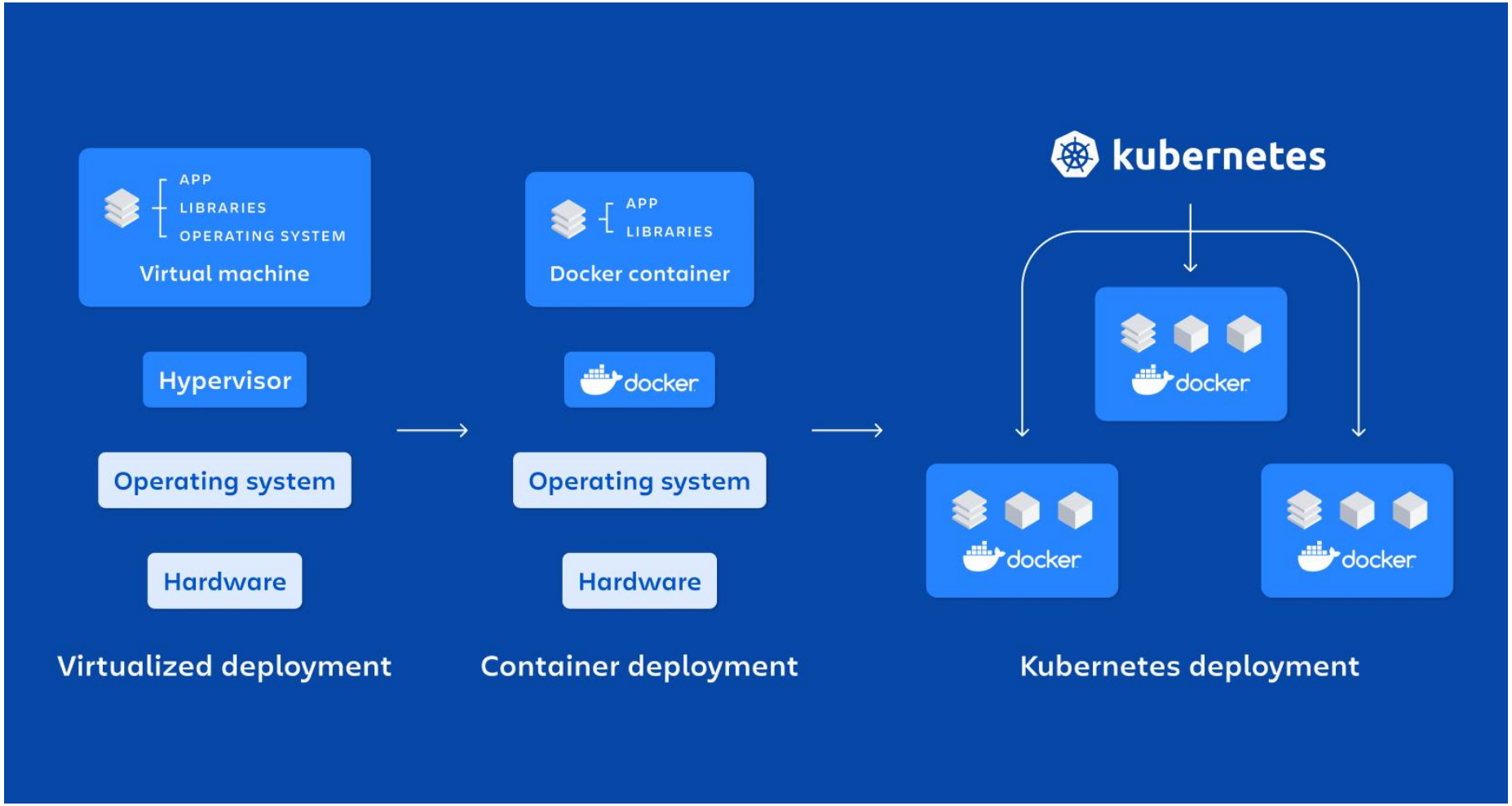
Contenedores en la nube: Kubernetes

Kubernetes es una plataforma que organiza y gestiona automáticamente contenedores (como los de Docker), asegurando que las aplicaciones se ejecuten, escalen y se mantengan funcionando sin intervención manual.

Kubernetes: del griego κυβερνήτης, en español “timonel” o “gobernante”

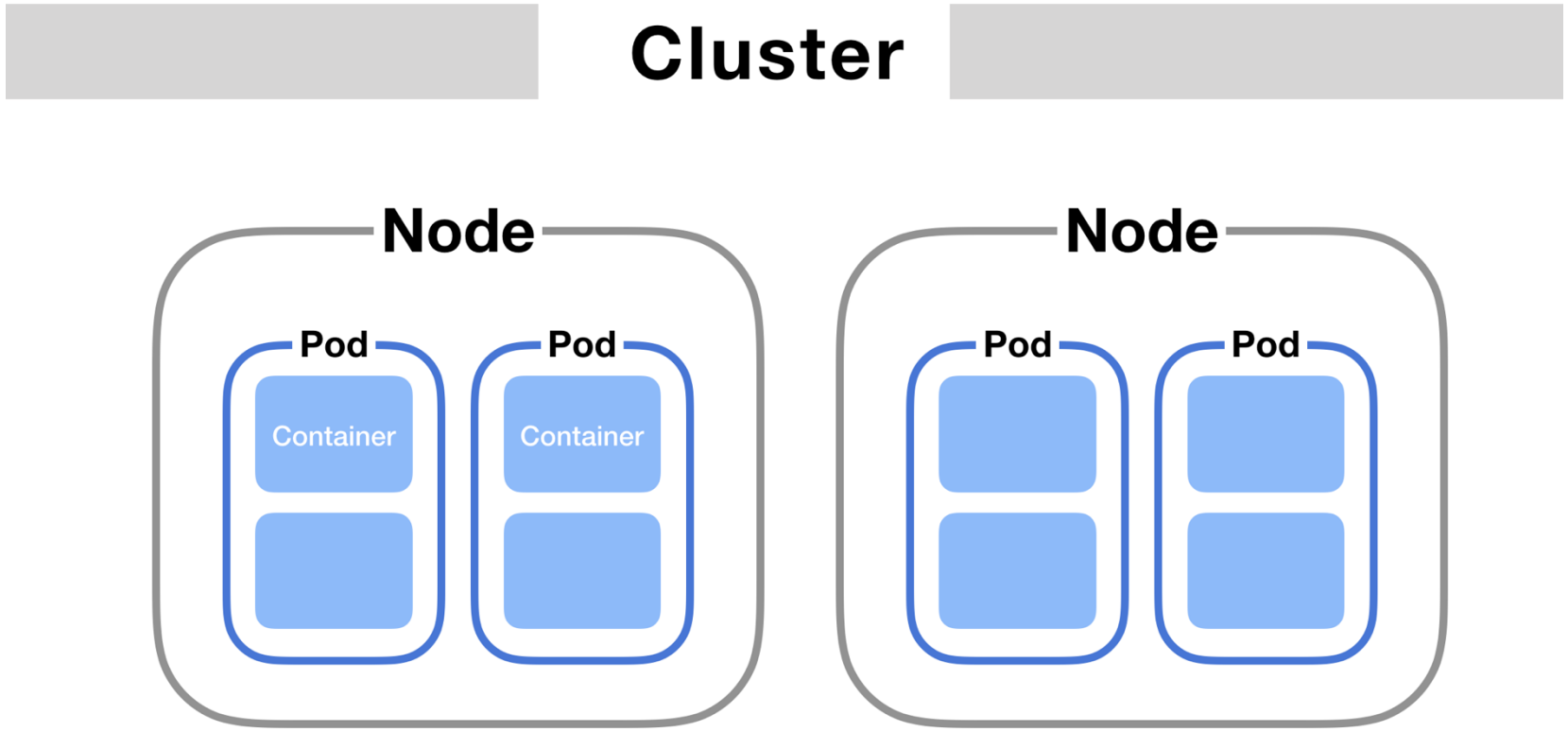
3. Contenedores

Contenedores en la nube: Kubernetes



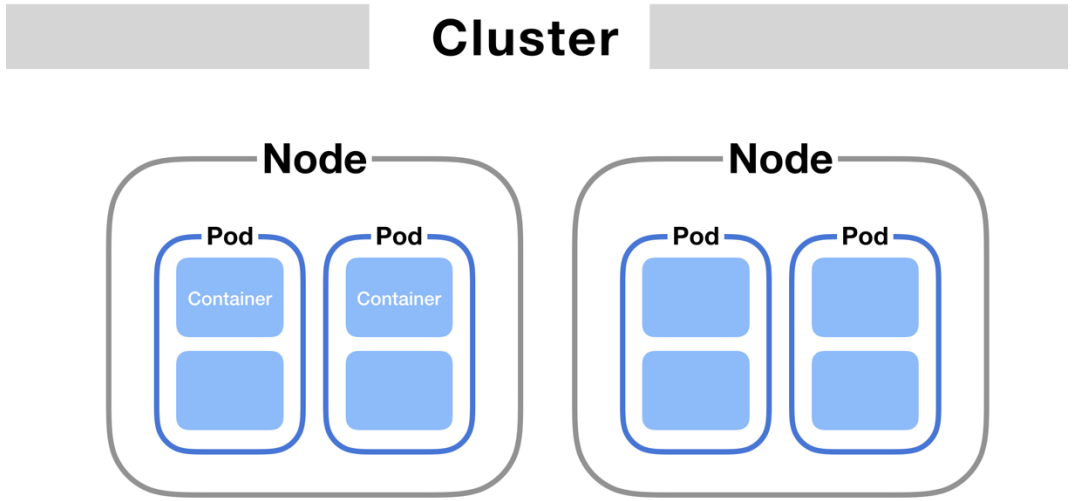
3. Contenedores

Contenedores en la nube: Kubernetes (jerarquía)



3. Contenedores

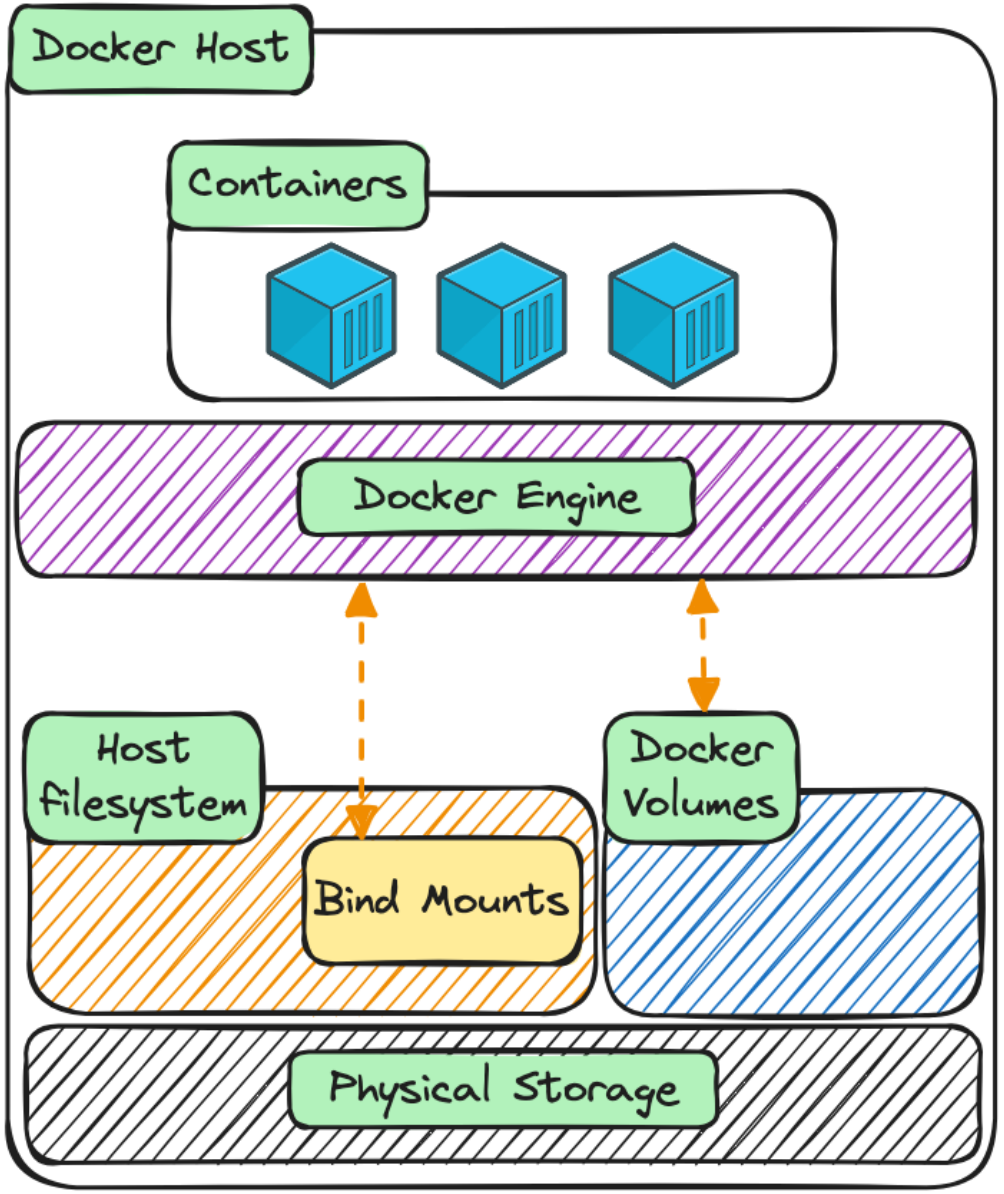
Contenedores en la nube: Kubernetes (jerarquía)



- **Contenedor**: ejecuta una aplicación o servicio concreto
- **Pod**: agrupa uno o varios contenedores que comparten red y almacenamiento
- **Nodo**: es una máquina (física o virtual) donde se ejecutan los pods
- **Cluster**: es el conjunto de nodos gestionados por Kubernetes que trabajan juntos para ejecutar todas las aplicaciones

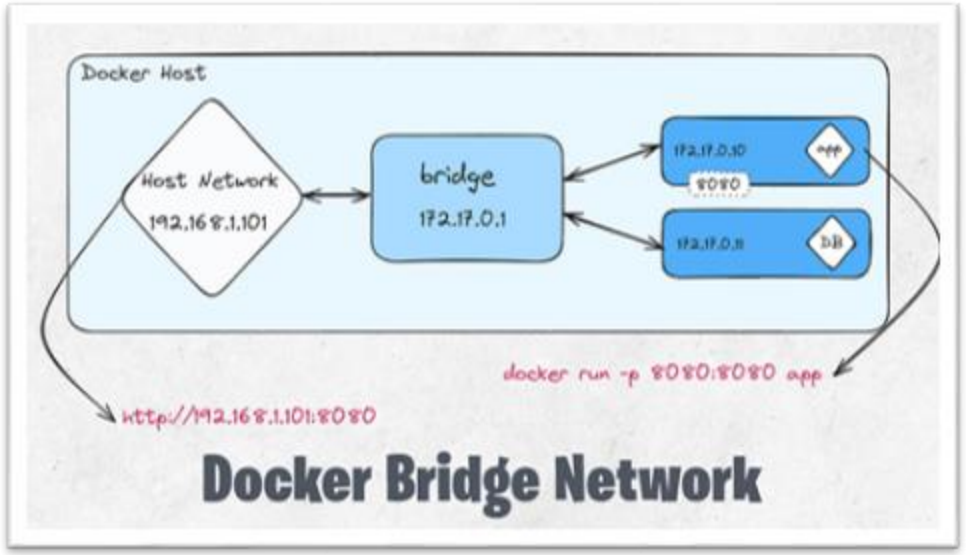
1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. **Volúmenes y redes**
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

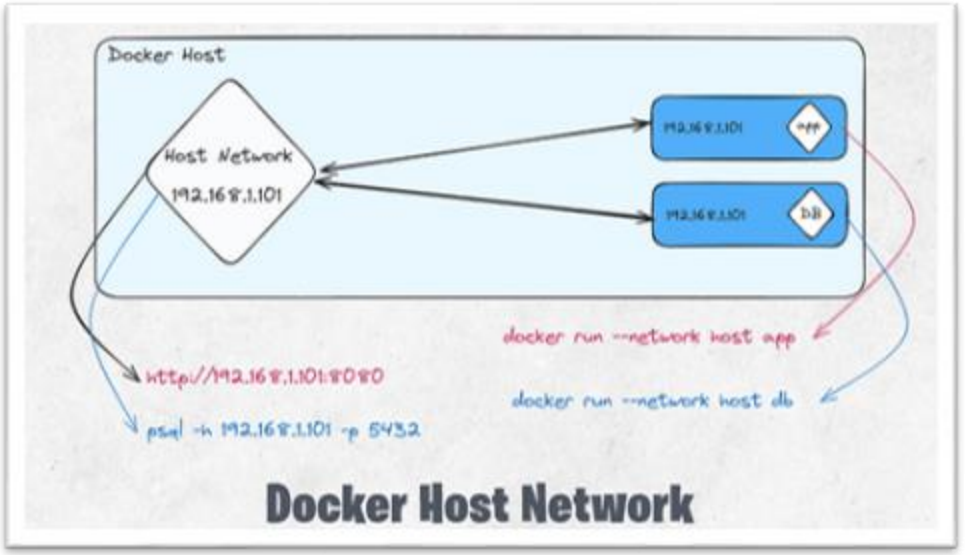
4. Volúmenes y redes

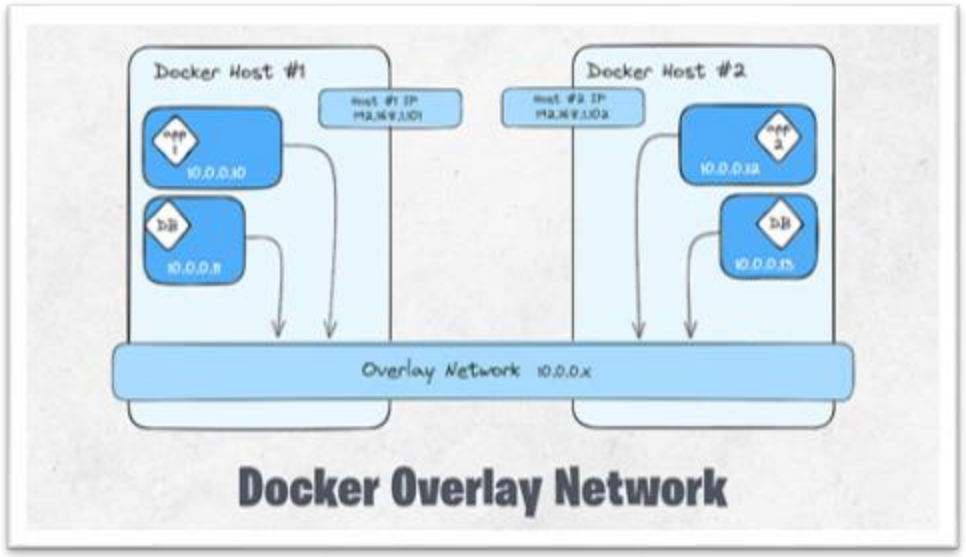


Los volúmenes permiten que los datos **se conserven incluso si el contenedor se elimina o reinicia**, manteniendo los archivos de manera persistente en el host.

4. Volúmenes y redes





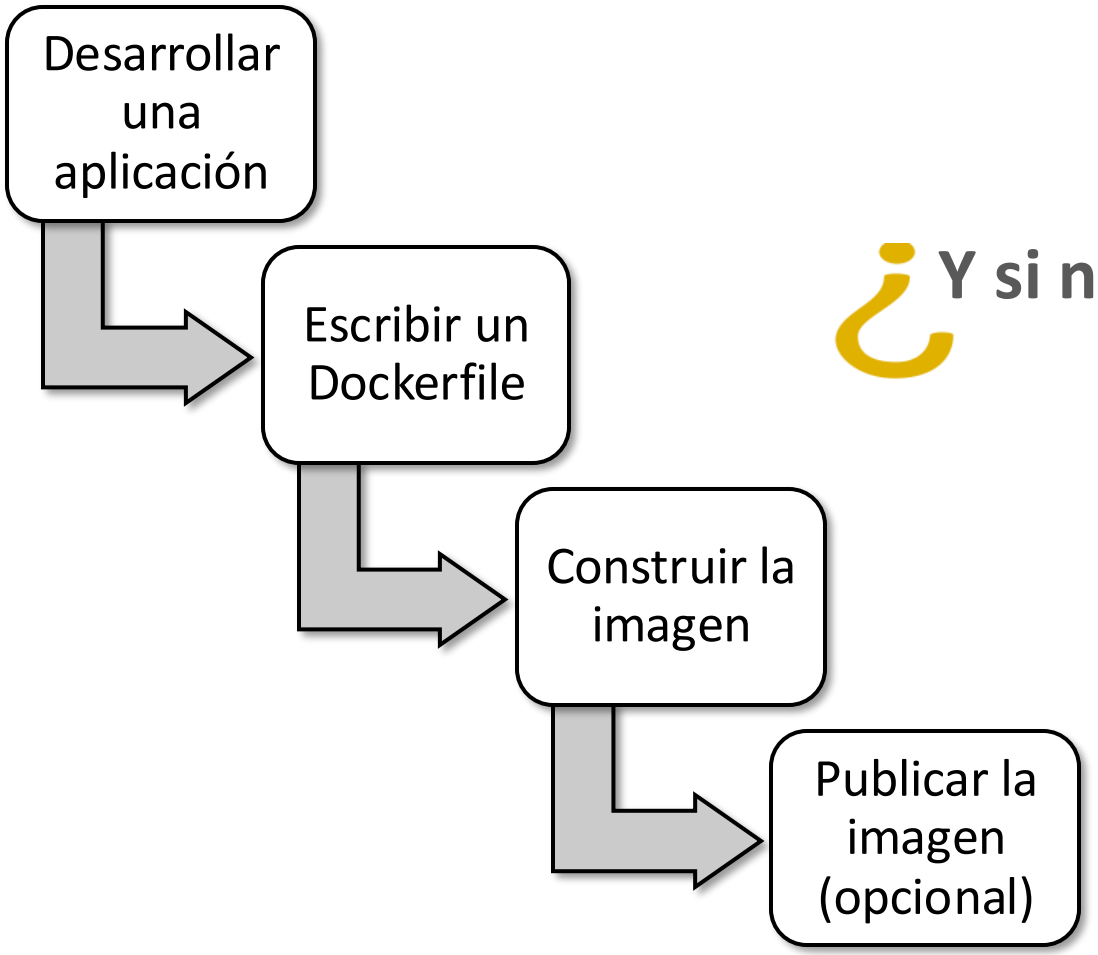


Las redes permiten que los contenedores **se comuniquen entre sí, con el host y con redes externas.** Docker proporciona **varios tipos de redes** (bridge, host, overlay, etc.) que permiten controlar el aislamiento, la seguridad y la visibilidad entre contenedores, facilitando la configuración de infraestructuras distribuidas o microservicios.

https://medium.com/@a-dem/docker-network-explained-in-details-51da1030275b

1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. **Composición de servicios: Docker Compose**
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

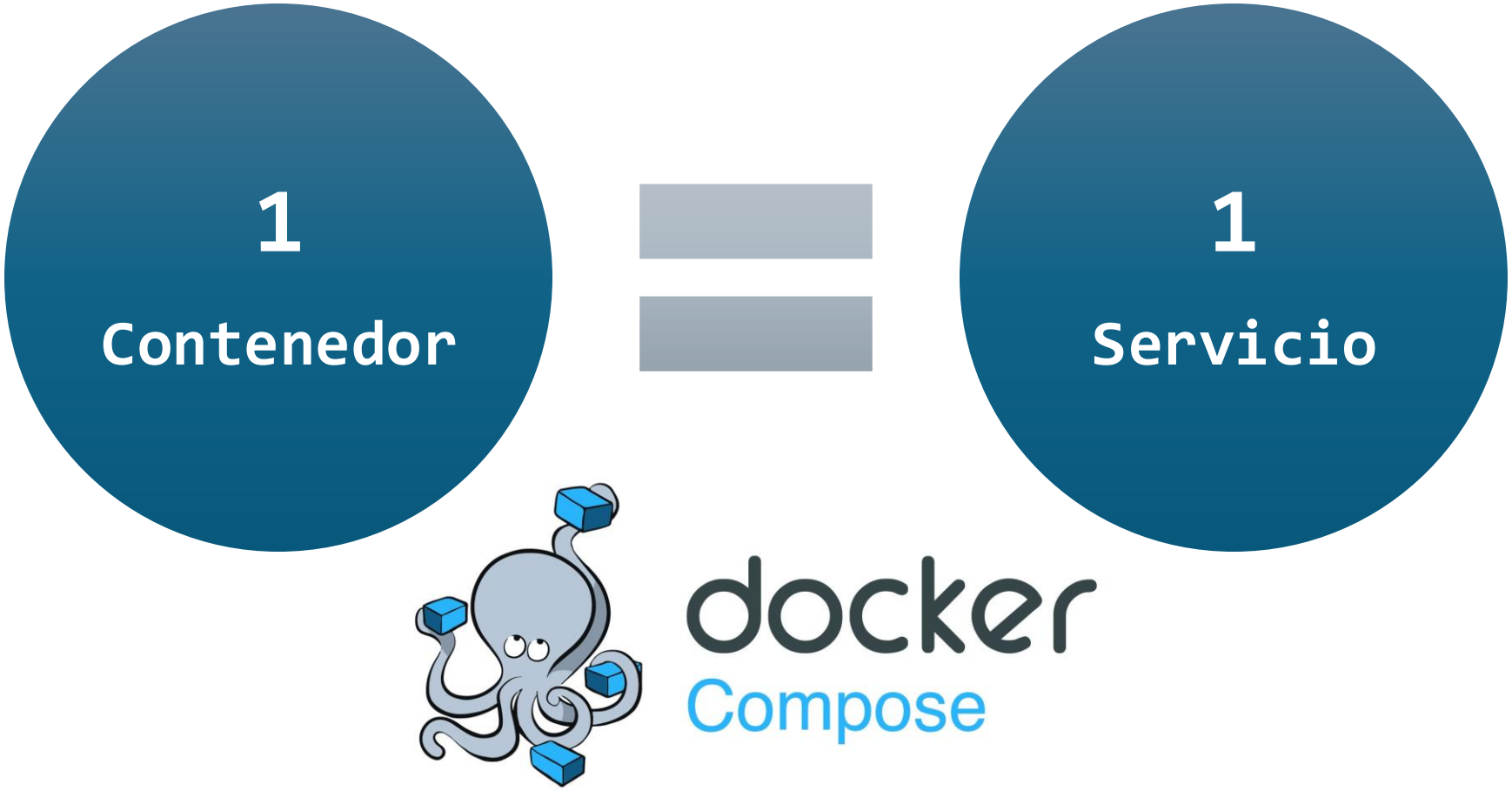
5. Composición de servicios: Docker compose



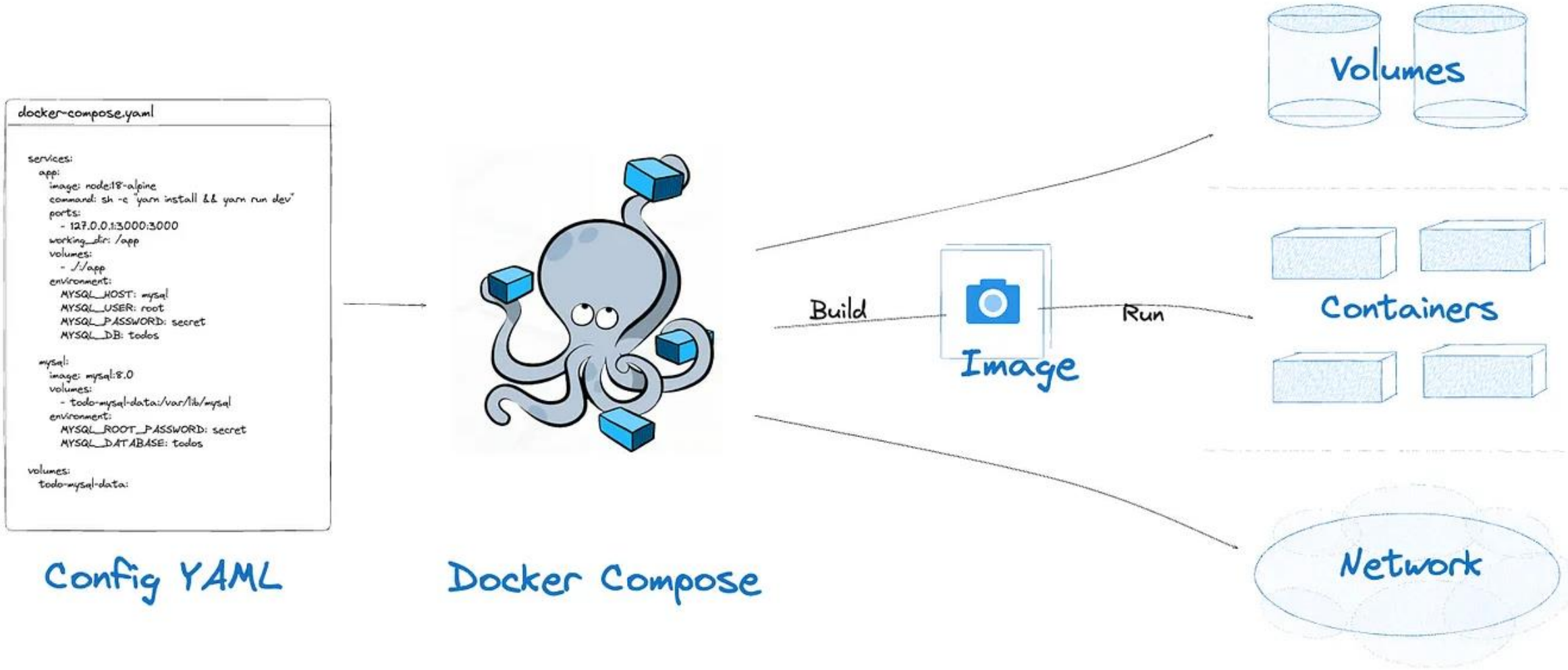
¿Y si necesito varias imágenes y varios contenedores?

5. Composición de servicios: Docker compose

Principio de responsabilidad única



5. Composición de servicios: Docker compose



1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. **Dev-containers**
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. Tutorial: *dockerizando uvlhub*

6. Dev-containers

¿Pero mi app web está en Docker y yo sigo trabajando con el código en mi host?

Mi VS Code no detecta las librerías al estar fuera del contenedor web…

Cada miembro de mi equipo tiene una configuración de VS Code distinta…

6. Dev-containers

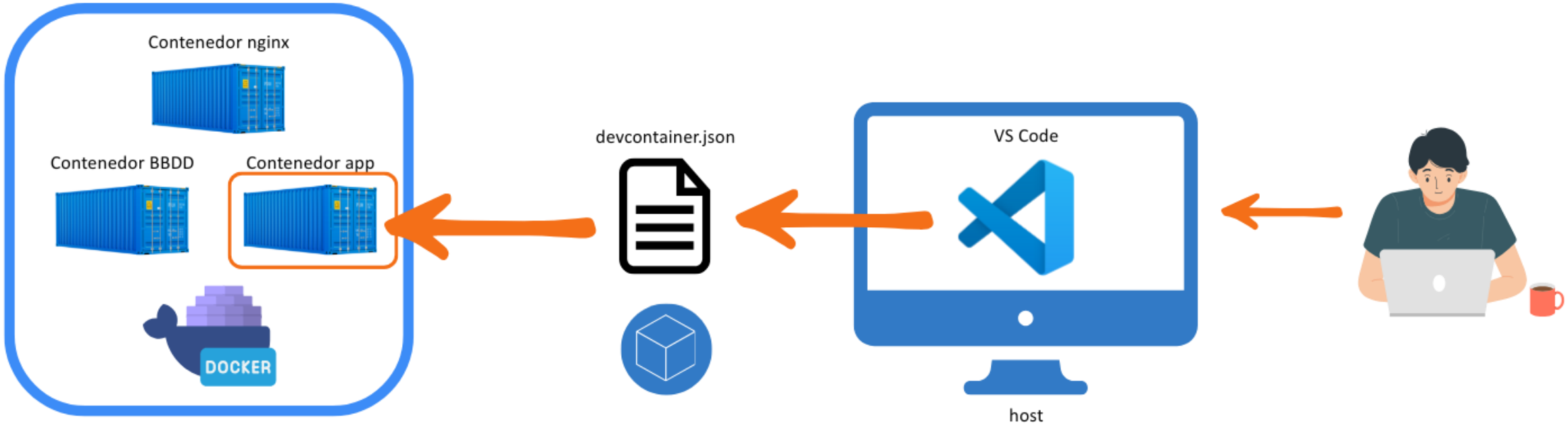
Definición

Un **Dev Container** es un **entorno de desarrollo preconfigurado** dentro de un contenedor Docker.

Garantiza que todos los desarrolladores trabajen con las mismas herramientas, dependencias y configuración.

6. Dev-containers

Diferencia entre container y dev-container



Un dev-container nos “sitúa” dentro de un contenedor para aprovechar el aislamiento y la potencia de VS Code

6. Dev-containers

Ejemplo

Para cargar nuestro dev-container

Mac: ⌘ + Shift + P

Windows / Linux: Ctrl + Shift + P

Escribir en el cuadro de búsqueda:

Dev Containers: Reopen in Container

y elegimos abrir la raíz del proyecto

`.devcontainer/devcontainer.json`

1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. **Y yo, ¿qué puedo hacer en mi proyecto?**
8. Tutorial: *dockerizando uvlhub*

7. Y yo, ¿qué puedo hacer en mi proyecto?

¡Diseña tu propio despliegue en Docker!

Modifica y/o crea ***Dockerfiles*** propios

Añade nuevos contenedores, volúmenes y redes a tus ***docker-compose.yml***

1. Introducción a Docker y DockerHub
2. Imágenes
3. Contenedores
4. Volúmenes y redes
5. Composición de servicios: Docker Compose
6. Dev-containers
7. Y yo, ¿qué puedo hacer en mi proyecto?
8. **Tutorial: *dockerizando uvlhub***

Antes de continuar, hay que actualizar nuestro proyecto con el fork de la asignatura

1º) Añadir repositorio original como upstream

```
git remote add upstream https://github.com/EGCETSII/uvlhub
```

2º) Fetch del repositorio original

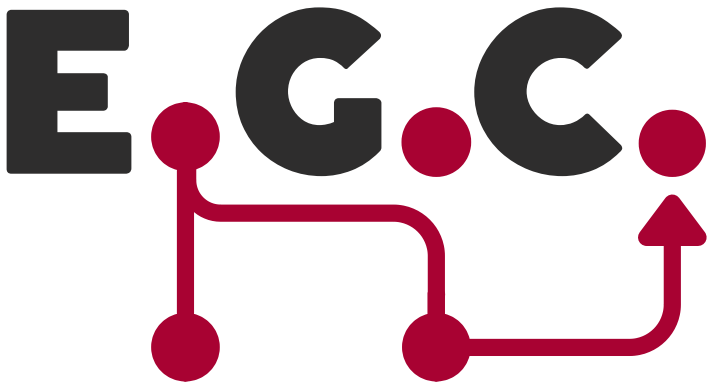
```
git fetch upstream
```

3º) Fusionar los cambios en tu fork

```
git checkout main
git merge upstream/main
```

8. Tutorial: *dockerizando uvlhub*

https://1984.lsi.us.es/wiki-egc/index.php/Tutorial-docker



Grado en Ingeniería Informática - Ingeniería del Software

Evolución y Gestión de la Configuración

Universidad de Sevilla 1505

Escuela Técnica Superior de Ingeniería Informática

¡Gracias!

"Dado un número suficientemente elevado de ojos, todos los errores se vuelven obvios."

- Eric S. Raymond (ley de Linus)