Repaso de cómo usar Docker y ROS2

Robótica Móvil

2^{do} cuatrimestre 2025

1 Guía de repaso general

En primera instancia, descargar los archivos `Dockerfile` y `start-docker.sh` del Campus de la materia. Luego, verificar si la imagen de Docker fue creada previamente:

```
$ docker images
```

Si la imagen `ros2_robotica` no aparece listada, en una terminal situada en el mismo directorio en que se descargó los archivos hacer:

```
$ bash start-docker.sh build
```

Este comando creará la imagen de Docker a utilizar en la materia. Luego, verificar si hay un contenedor de dicha imagen activo:

```
$ docker ps -a
```

Si el contenedor `ros2_robotica` no aparece listado, hacer:

```
$ bash start-docker.sh start
```

Esta instrucción creará el comando, y también vinculará una carpeta llamada `volume`, situada en el mismo directorio en donde se encuentra el archivo `start-docker.sh`, como una carpeta compartida entre el contenedor y el sistema operativo fuera de éste. Ésto es importante porque permite editar paquetes y archivos fuera del contenedor, y ejecutarlos luego dentro de éste. Todos los archivos que pongan **dentro** de la carpeta `volume` podrán ser accedidos desde dentro del container con la ruta `/root/ros2_ws/src/robotica`.
Una vez creado el contenedor, y habiendo puesto los archivos necesarios (paquetes de ROS2 a utilizar y demás) dentro de la carpeta compartida, se debe ingresar al contenedor:

```
$ bash start-docker.sh open
```

Con esto, verán que ahora la terminal les indica que ya no están situados en el directorio donde se ejecutó el archivo `start-docker.sh`, sino dentro del contenedor. Desde aquí es posible, por ejemplo, enviar los comandos de `ros2 run <NODO> <PAQUETE>` para ejecutar paquetes de algún nodo de ROS2, `ros2 topic list` para identificar los tópicos disponibles, `ros2 topic pub` para publicar un mensaje en algún tópico, `coppeliaSim.sh` para abrir el simulador CoppeliaSim, `rviz2` para abrir el visualizador, y demás. Tengan en cuenta que muchas veces para trabajar en ROS2 **se necesitan múltiples terminales** en paralelo, por lo que deberán abrir tantas terminales como necesiten, y entrar en el contenedor en cada una de ellas de la misma forma que en la primera. Por ejemplo, se necesitan terminales distintas para abrir el simulador, el visualizador y ejecutar algún nodo de ROS2 en simultáneo, y que éstos puedan referirse entre sí a través de tópicos.

Recuerden además que **cada vez** que se agreguen o modifiquen paquetes dentro del *workspace* de ROS2 es necesario compilarlos y referirlos. Esto se realiza situándose en el directorio `/root/ros2_ws`, ejecutando los comandos:

```
$ colcon build
$ source install/setup.bash
```

Esto NO funciona si no están situados en el directorio correspondiente. Además, es necesario realizar el comando de `source` una vez **por cada terminal abierta** del contenedor, para que en dicha terminal se identifique el *workspace* con el que se trabajará.

2 Docker en las computadoras del laboratorio

Dado que en las computadoras del laboratorio de la Facultad no es posible utilizar comandos de `sudo`, los cuales son requeridos por Docker, se utiliza la versión `rootless` de Docker, la cual permite trabajar sin esta limitación. Como por defecto cada terminal se refiere a la versión estándar de Docker, es necesario seguir los siguientes pasos para el uso de la versión `rootless`: En primer lugar, utilizar el siguiente comando:

```
$ ls -l /run/user/$(id -u)/docker.sock
```

Si se encuentra el archivo `docker.sock`, se devuelve algo similar a :
`srw-rw---- 1 youruser youruser 0 ...  /run/user/1000/docker.sock`
En este caso, es necesario, **en cada terminal** en la que se vaya a utilizar Docker, hacer:

```
$ export DOCKER_HOST=unix:///run/user/$(id -u)/docker.sock
```

En caso contrario, si el archivo `docker.sock` no fue encontrado, es necesario hacer previamente:

```
$ dockerd-rootless.sh &
```

Luego de esto, verificar la existencia de este archivo con el primer comando, y utilizar el `export` para poder proceder al trabajo con Docker.