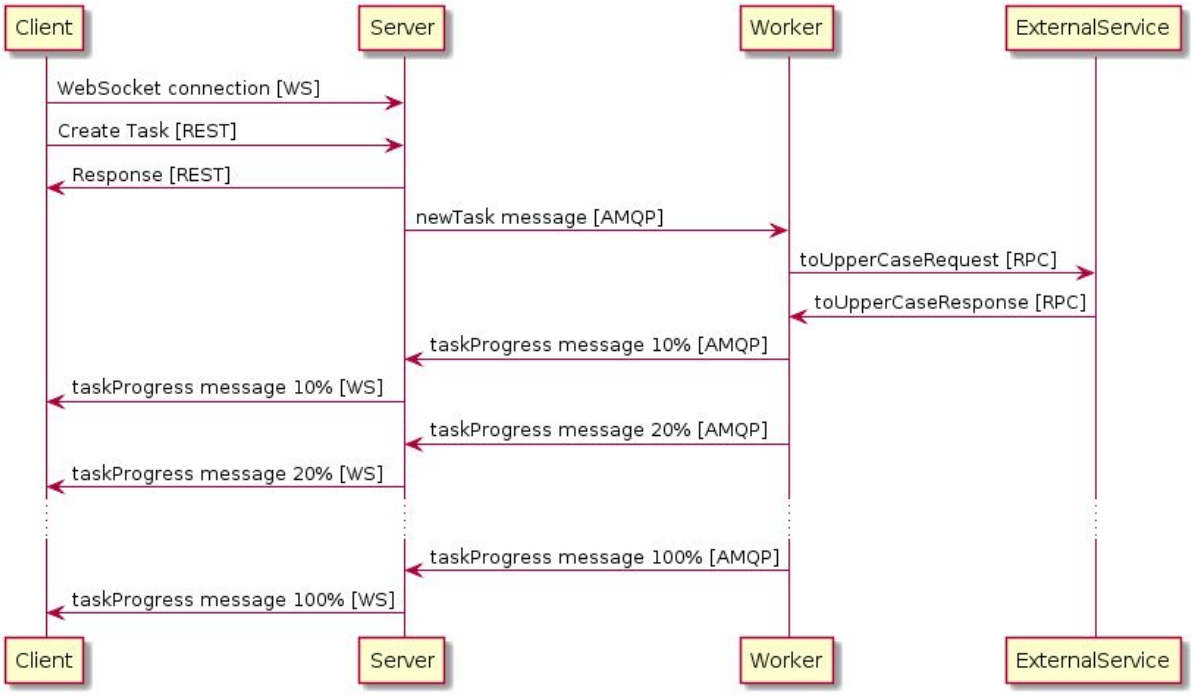
Práctica 1. Dockerizar una aplicación

Enunciado

Dadas las ventajas que proporciona Docker para el empaquetado, distribución y ejecución de aplicaciones, vamos a ampliar y “dockerizar” una aplicación implementada como parte de una práctica previa en el máster.

En concreto, se va a ampliar y empaquetar en diferentes contenedores Docker la aplicación de la Práctica 3 de la asignatura “Tecnologías de Servicios de Internet” del Módulo II - Servicios de Internet.

La aplicación está formada por 3 servicios que interactúan entre sí usando diferentes tipos de protocolos. Los servicios son Server, Worker y External Service. La comunicación entre ellos se representa en el siguiente diagrama:



Ampliación

Se va a ampliar la aplicación para añadir persistencia al servicio Server y al servicio Worker. En concreto, la nueva funcionalidad es la siguiente:

- **Server:** Se deberá conectar este servicio con una base de datos Mongo, de forma que cada vez que se reciba una petición, deberá registrarse en una colección de la base de datos. El formato concreto queda a criterio del alumno.
- **Worker:** Se deberá conectar este servicio con una base de datos MySQL mediante SpringData. Deberá registrar en una tabla de la base de datos las peticiones que atiende. Al igual que con el server, el formato concreto queda a criterio del alumno.

Desarrollo con Docker

Empaquetado con Docker

Se debe empaquetar la aplicación siguiendo estos criterios:

- Cada servicio deberá empaquetarse en su propio contenedor.
- Para las bases de datos Mongo y MySQL se usarán los contenedores publicados en DockerHub.
- Para los servicios Server, Worker y ExternalService, habrá que crear contenedores propios que deberán publicarse en una cuenta de DockerHub del alumno.
- Todos los contenedores de la aplicación estarán coordinados usando docker-compose.
- Se deberán crear dos docker-compose:
    - Un docker-compose de **desarrollo**, que construirá localmente los contenedores de los servicios en tiempo de inicio
    - Un docker-compose de **ejecución**, que usará las imágenes publicadas en DockerHub.
- Creación de contenedores:
    - Los servicios Server y ExternalService se deben empaquetar usando una imagen base con Node instalado.
    - Para el servicio Worker se debe crear un Multistage Dockerfile que será usado por el docker-compose de desarrollo.
    - El servicio Worker también permitirá su empaquetado mediante JIB. La imagen creada y publicada por JIB será la usada en el docker-compose de ejecución.
- Se crearán los ficheros de configuración de VSCode necesarios para que los servicios puedan desarrollarse usando VSCode dentro de contenedores de desarrollo.
- En la raíz del repositorio deberá incluirse un fichero README.md que describa los diferentes modos de desarrollo y ejecución de la aplicación.

Formato de entrega

La práctica se entregará teniendo en cuenta los siguientes aspectos:

- La práctica se entregará como un fichero .zip con el código del los diferentes servicios y los ficheros auxiliares. El nombre del fichero .zip será el correo URJC del alumno (sin @alumnos.urjc.es).

Las prácticas se podrán realizar de forma individual o por parejas. En caso de que la práctica se haga por parejas:

- Sólo será entregada por uno de los alumnos
- El nombre del fichero .zip contendrá el correo de ambos alumnos separado por guión. Por ejemplo `p.perezf2019-z.gonzalez2019.zip`