Informe Proyecto 1

Esteban Osorio Florez

Samuel Cadavid Perez

Santiago Vanegas Perez

EAFIT – 2021

Introducción

El objetivo de este proyecto 1 es diseñar e implementar un MIDDLEWARE que permita a un conjunto de CLIENTES enviar y recibir mensajes de datos. Esto permitirá a los alumnos evidenciar, conocer y aplicar, muchas de las características subyacentes a los sistemas distribuidos (ej.: heterogeneidad, transparencia, seguridad, escalabilidad, entre otros) que deben implementar las aplicaciones o los subsistemas base (sistema operativo, middlewares, frameworks, apis, etc.). En este caso, dicha complejidad y características del sistema distribuido serán diseñadas e implementadas en un MIDDLEWARE, de tal manera que para las aplicaciones usuarias (CLIENTES) sea transparente y seguro su uso.

Selección de Aplicación

Para el desarrollo de este proyecto decidimos implementar la app #2 la cual se basa en:

Simulación de un gestor de tareas para procesamiento distribuido: Tenemos un conjunto de clientes que tienen cada uno tareas para procesar (task_i) y tenemos un conjunto de servidores que ejecutan esas tareas (task_i). Los clientes (publishers) envían a una cola las tareas específicas a procesar, estas tareas son almacenadas en una cola del MOM. Los servidores (subscribers) van tarea por tarea a la cola a retirar una task_i específica y ejecutarla. Una vez un servidor termina una tarea específica notificará por otro medio al cliente específico (fuera del alcance de esta prueba). En síntesis:

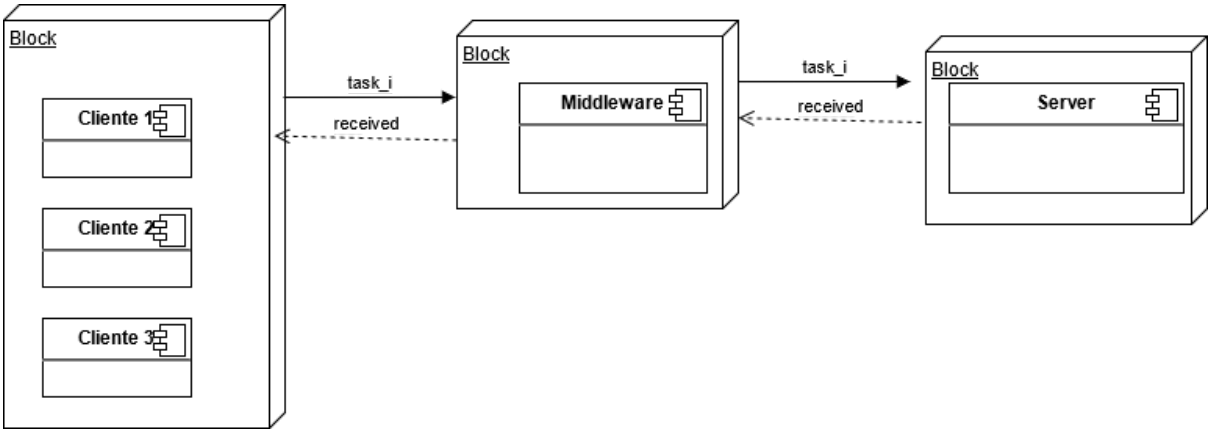
1. Cola: Almacenamiento en el MOM donde se depositan las tareas.
2. Mensaje: Identificador de la tarea, el username y el email para notificación
3. Cliente: Publisher que envía tareas a la cola.
4. Server: Subscribe quien extrae uno a uno tareas de la cola.

Análisis de la problemática

Se desarrollará una aplicación desde la cual un cliente pueda crear N cantidad de colas a las cuales se les ingresarán una cantidad N de tareas. Posteriormente un Server que estará suscrito a cuantas colas quiera podrá consumir y procesar dichas tareas anteriormente provistas por el Cliente.

Diseño

Se diseñó una aplicación distribuida en la que se tienen clientes enviando tareas por procesar a un middleware, implementado usando RabbitMQ para tener colas donde los mensajes (tareas a procesar) se acumulen para ser consumidos por servidores que se encargan de procesar las tareas.



Requerimientos

Para el desarrollo de este proyecto usamos el lenguaje de programación de Python y para la implementación de toda la app y del middleware usamos las siguientes librerías:

- bidict==0.21.2
- click==7.1.2
- dnspython==1.16.0
- eventlet==0.30.2
- Flask==1.1.2
- Flask-Cors==3.0.10
- Flask-SocketIO==5.0.1
- greenlet==1.0.0
- itsdangerous==1.1.0
- Jinja2==2.11.3
- MarkupSafe==1.1.1

- pika==1.2.0
- PyJWT==2.0.1
- pymongo==3.11.3
- python-engineio==4.0.1
- python-socketio==5.1.0
- six==1.15.0
- Werkzeug==1.0.1

Atributos de calidad

- Seguridad
- Escalabilidad
- Interoperatividad
- Modificabilidad.
- Rendimiento.
- Mantenibilidad.

Consideraciones

Se considera importante para esta aplicación la escalabilidad que se facilita por el uso de un middleware, clientes y servidores separados, la implementación de nuevas funciones y módulos nuevos es más cómoda para el equipo de desarrollo. Asimismo la interoperabilidad es crucial para el desarrollo de una aplicación de procesamiento distribuido.