Trabajo Práctico 2
Reddit Memes Analyzer
Documento de Arquitectura

Sistemas Distribuidos I (75.74)

1er Cuatrimestre 2022
Facultad de Ingeniería
Universidad de Buenos Aires

Mauro Parafati
102749
mparafati at fi.uba.ar

Monorepositorio de TPs

Índice

1. Introducción

En el presente se documenta la arquitectura diseñada para la resolución del segundo trabajo práctico de la materia, que consiste en procesar una *alta* cantidad de posts y comentarios de Reddit para luego resolver ciertas consultas.

El sistema debe ser **escalable** y permitir **multi-computing**.

1.1. Naturaleza del problema

Dado el alto volumen de datos a procesar, así como la independencia de los mismos, resulta óptimo pensar en procesar cada post y comentario *en linea*, diagramando un **pipeline** que permita maximizar a su vez el paralelismo.

2. DAG: Comprender el flujo de datos

Teniendo en mente que se busca diseñar un pipeline para el procesamiento en linea de los datos, resulta conveniente comenzar el análisis entendiendo el flujo de los mismos, para detectar posibilidad de paralelismo y dependencias entre estos. Un modelo muy utilizado es el de construir un DAG (Directed Acyclic Graph) de los datos. A continuación, una solución posible:

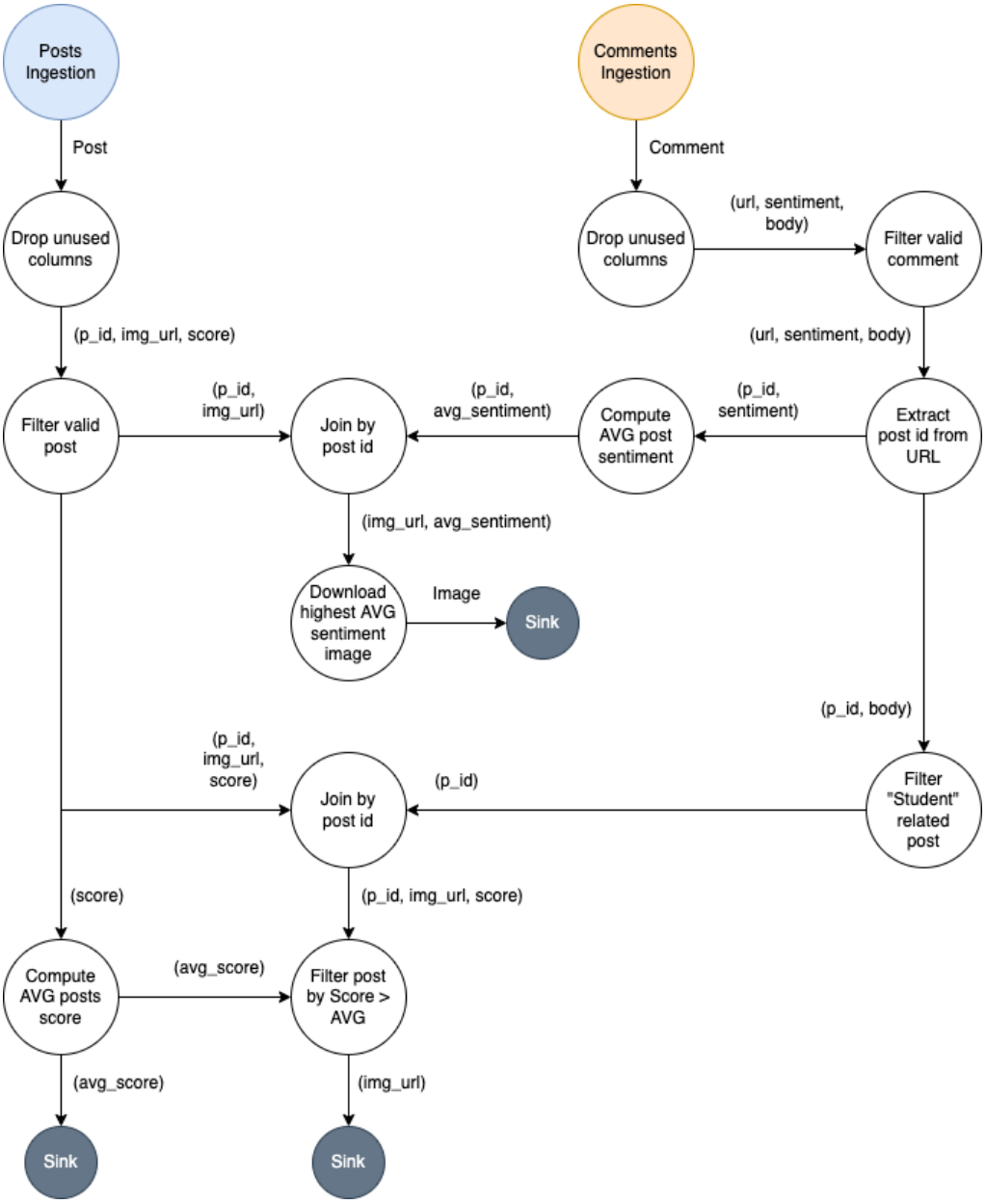

Figura 1: DAG de los datos

3. Diagrama de Robustez

Detectadas las dependencias entre datos y posibles caminos de los mismos, se procede a desarrollar un diagrama de robustez que permita detectar qué componentes son escalables y de qué forma.

3.1. Grupos

Para desarrollar el diagrama, primero se hace análisis de los **grupos presentes** en el sistema. Se puede observar que, para un nodo cualquiera del DAG, este podría mapear a un worker que realice tal tarea. Sin embargo, como el interés es escalar, debería ser posible que existan muchos de estos workers, para cualquier nodo del grafo.

Para cada grupo de workers, se asigna un componente que actua de *broker* dentro del grupo. Es el encargado de recibir los mensajes y distribuirlos en los workers según distintas estrategias (*round-robin*, *affinity*). Cada worker tendría su cola de mensajes, así como el broker.

Al querer escalar, se presenta una diferencia entre dos escenarios: workers **stateless** y workers **stateful**. Se sabe que los workers stateless suelen ser mucho mejores a la hora de desarrollar un sistema distribuido, ya que permiten elasticidad de manera muy sencilla. Sin embargo, para evitar tener que enviar la información de todos los posts a todos los nodos que los procesen, se podría plantear la estrategia de **afinidad** entre worker y post. Esto permitiría distribuir la carga y escalar al mismo tiempo, pero tiene la contra de que la distribución podría no ser uniforme.

Un diseño para el primer caso podría ser el siguiente:

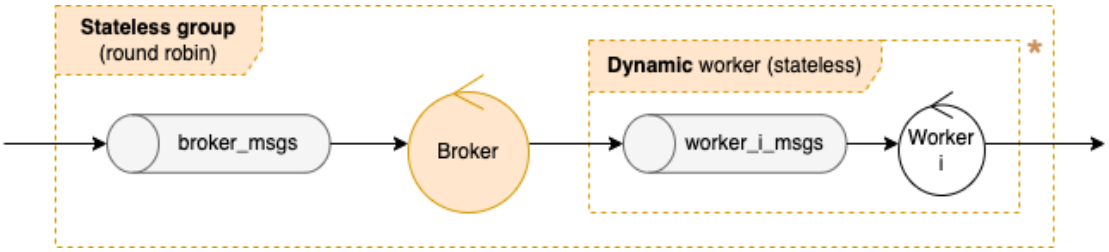

Figura 2: Grupo stateless

Mientras que para el segundo caso el diseño sería el siguiente:

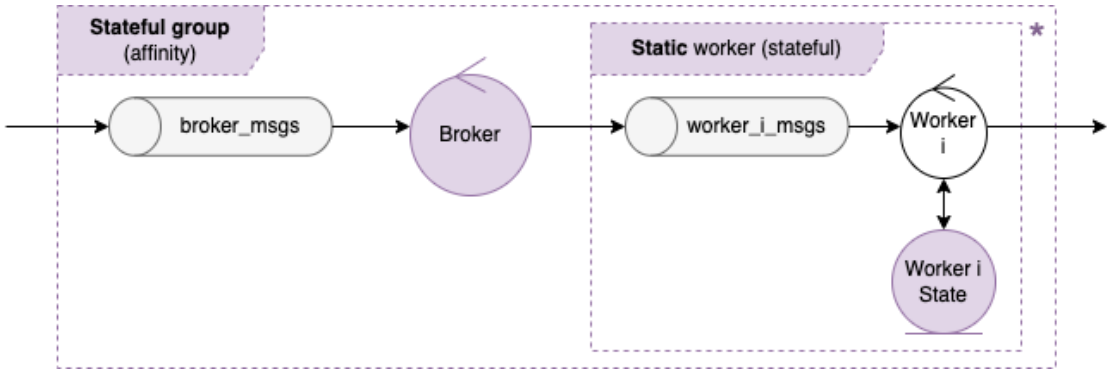

Figura 3: Grupo stateful

Estos dos tipos de grupo se utilizarán en el diagrama de robustez, dando a entender que por dentro existen estos componentes *(broker, colas, workers)*.

3.2. Diagrama final

Utilizando los grupos previamente definidos, se incluye el diagrama de robustez:

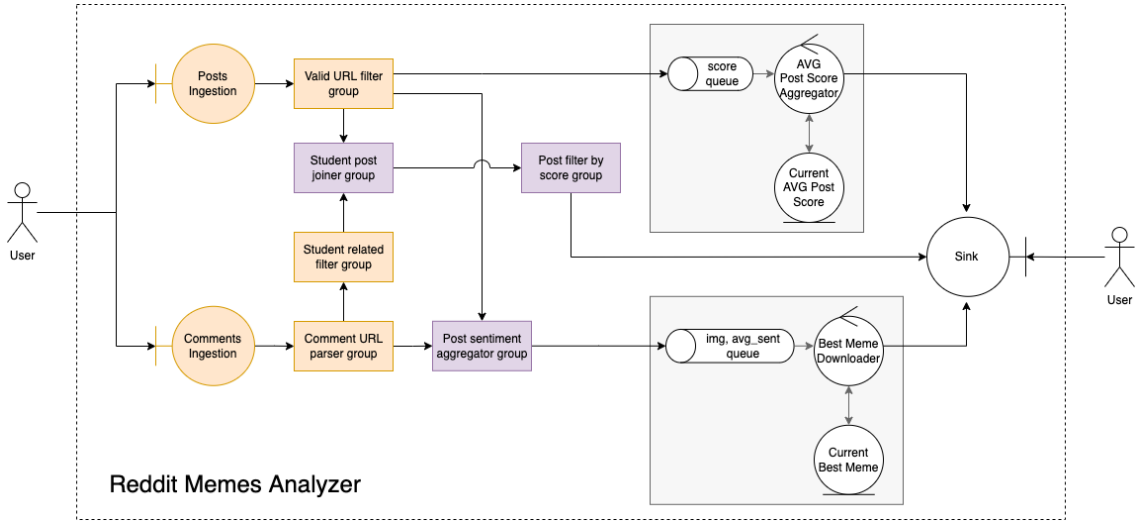

Figura 4: Diagrama de Robustez

Donde los grupos naranjas representan grupos dinámicos stateless, y los grupos violetas por el contrario son estáticos stateful.

También se puede ver la presencia de dos componentes que no son replicables, dado que deben tener estado centralizado.

4. Diagrama de Despliegue: MOM

Para la implementación del sistema se utiliza un MOM (Message-Oriented Middleware) como capa de abstracción para la comunicación entre grupos.

Dada la topología del sistema, en la que cada filtro mapea directamente con un grupo de workers como ya fue explicado en secciones anteriores, no se busca incluir en este diagrama los contenedores de todos los filtros, sino mostrar cómo se conectan todos los componentes del sistema entre sí:

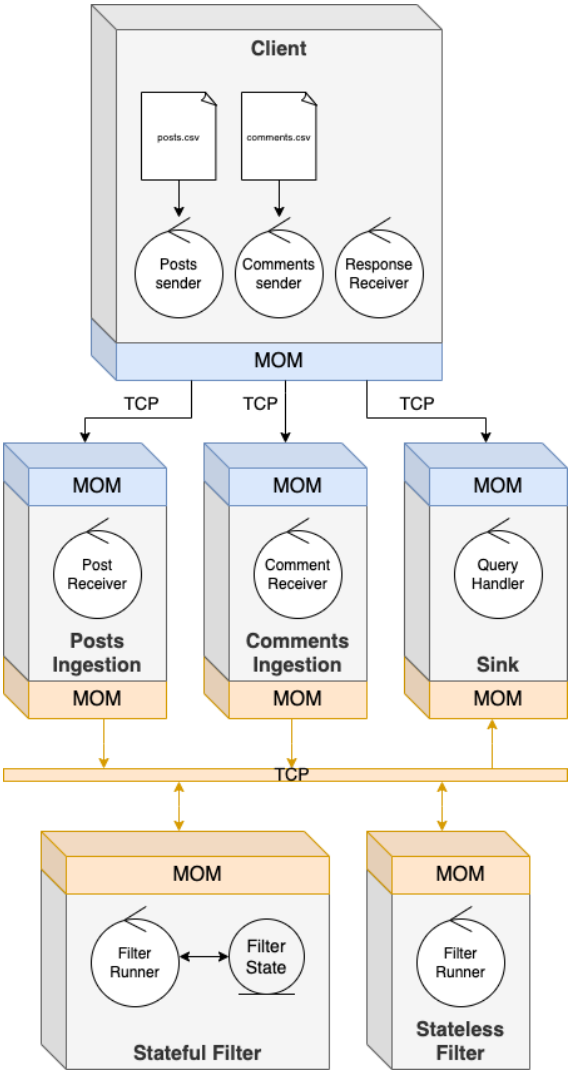

Figura 5: Diagrama de Despliegue

5. Diagrama de Actividades

Se incluyen a continuación tres diagramas que modelan las consultas a resolver con los datos.

5.1. Puntaje promedio de los posts

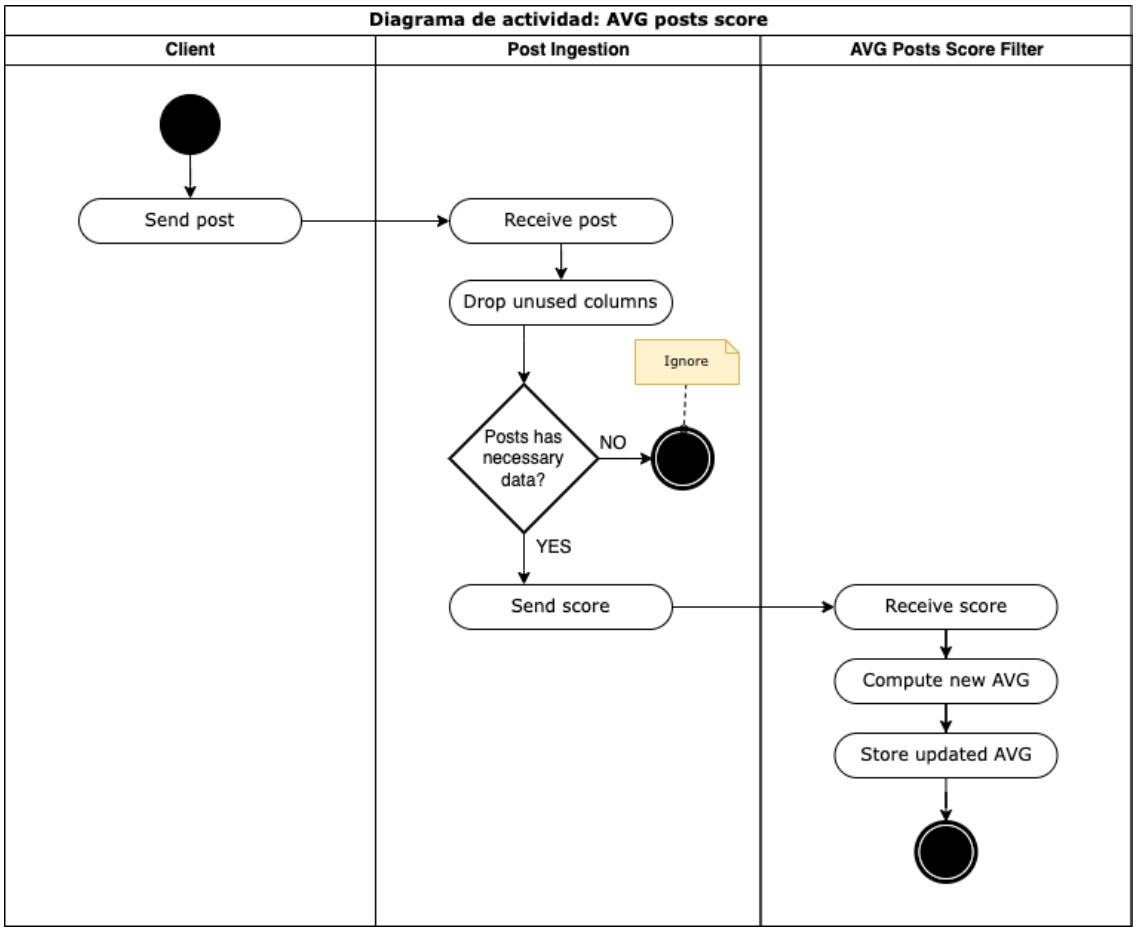

Figura 6: Diagrama de Actividades: puntaje promedio de los posts

5.2. Posts que gustan a estudiantes

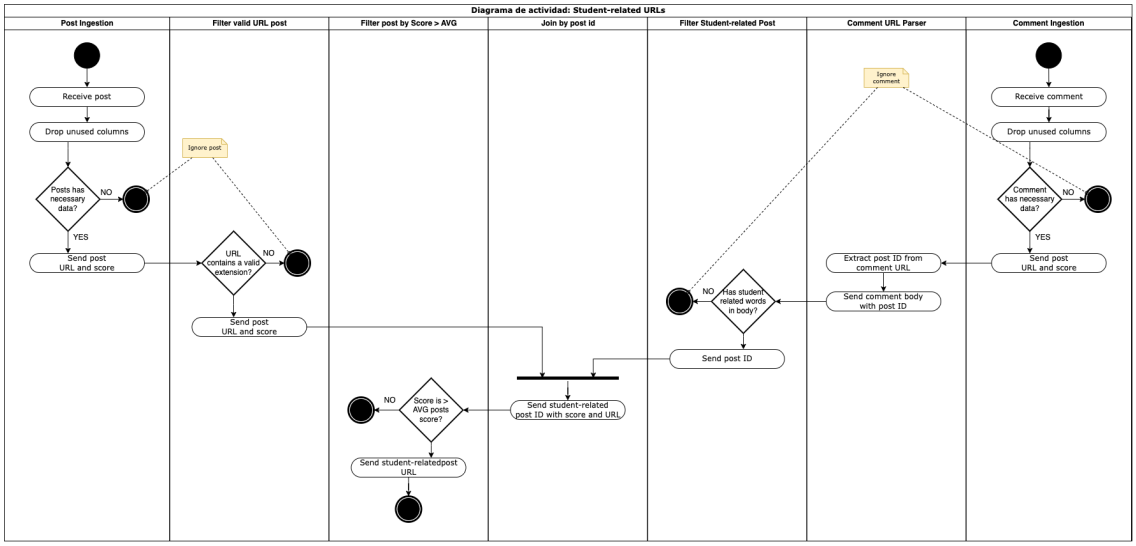

Figura 7: Diagrama de Actividades: posts que gustan a estudiantes

5.3. Meme con mayor sentiment promedio

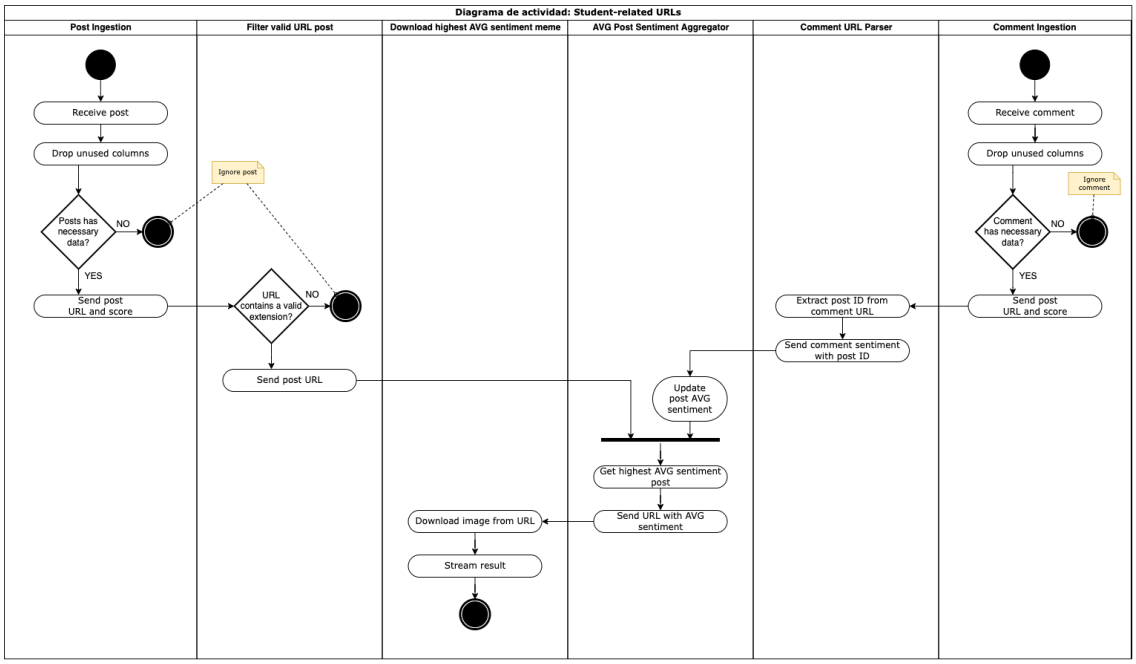

Figura 8: Diagrama de Actividades: descarga del meme con mayor *sentiment* promedio

6. Diagrama de Paquetes

Finalmente, se diseñó el siguiente diagrama de paquetes para comprender la relación entre los distintos módulos del sistema:

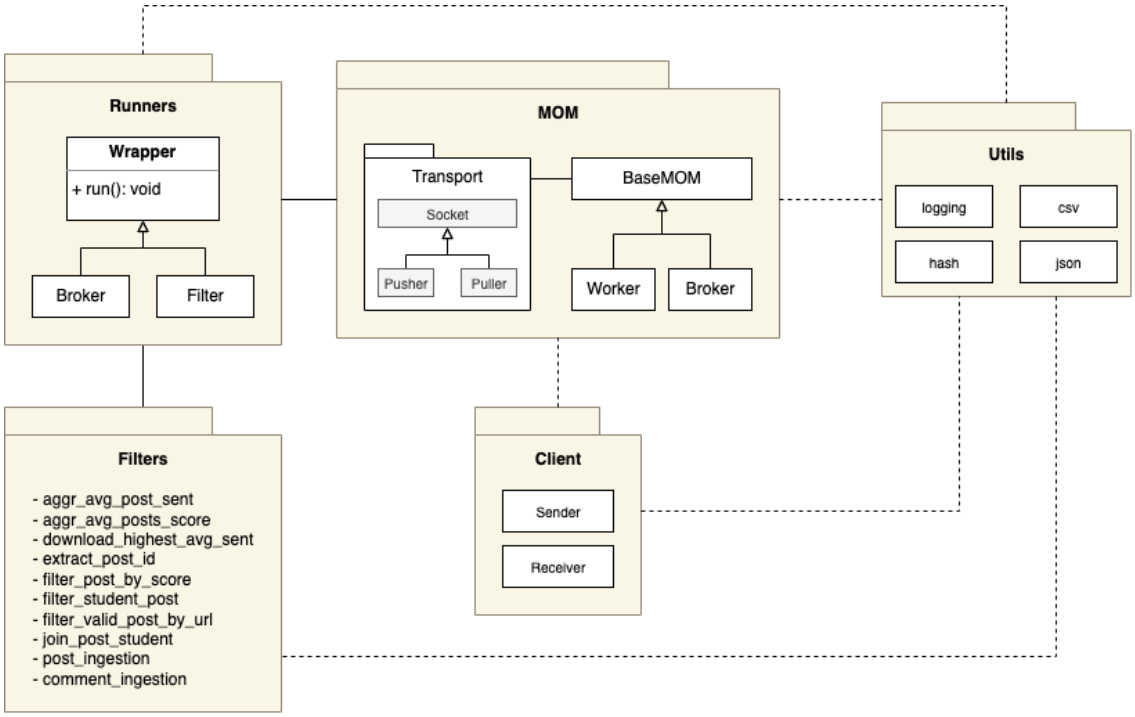

Figura 9: Diagrama de Paquetes