FACULTAD DE INGENIERIA
Universidad de Buenos Aires

# TP Diseño - Coffee Shop Analysis

[75.74] Sistemas Distribuidos I
Segundo cuatrimestre de 2025


| | | |
|---|---|---|
| Avecilla, Ignacio | 105067 | iavecilla@fi.uba.ar |
| Avila, Gaston | 104482 | gavila@fi.uba.ar |
| Muñoz, Juan Martín | 106699 | jmmunoz@fi.uba.ar |

# Índice

# 1. Alcance

**El presente informe presenta la documentación de un sistema distribuido flexible, robusto y escalable, capaz de resolver las consultas otorgadas por la catedra con una cantidad de unidades de procesamiento mayor o igual a uno.**

**Las consultas a resolver son:**

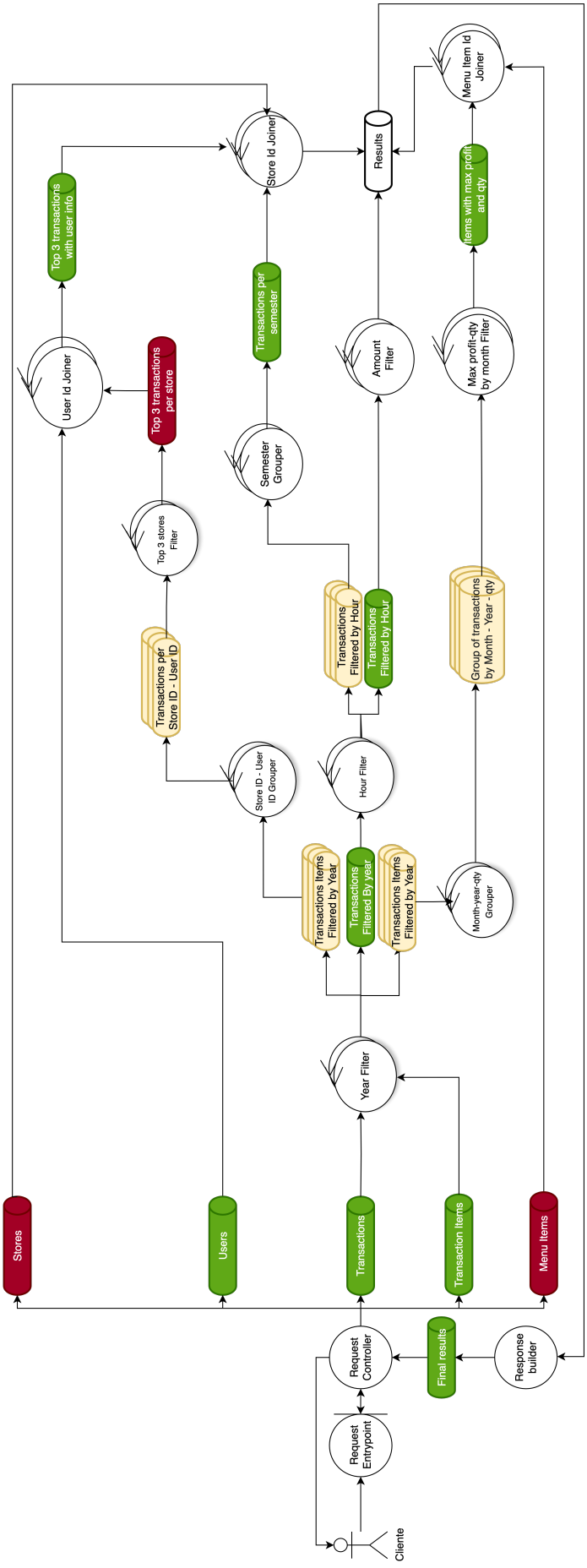
1. Transacciones (Id y monto) realizadas durante 2024 y 2025 entre las 06:00 AM y las 11:00 PM con monto total mayor o igual a 75.
2. Productos más vendidos (nombre y cant) y productos que más ganancias han generado (nombre y monto), para cada mes en 2024 y 2025.
3. TPV (Total Payment Value) por cada semestre en 2024 y 2025, para cada sucursal, para transacciones realizadas entre las 06:00 AM y las 11:00 PM.
4. Fecha de cumpleaños de los 3 clientes que han hecho más compras durante 2024 y 2025, para cada sucursal.

# 2. Arquitectura

## 2.1. Vista Física

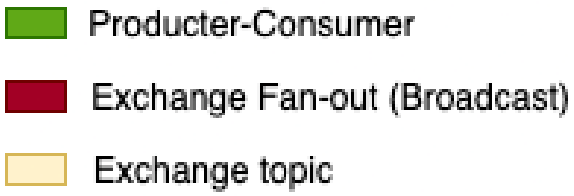
**Se muestra la enteridad del sistema y las conexiones existentes entre las diversas entidades del sistema.**

### 2.1.1. Diagrama de Robustez

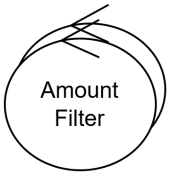


**Se visualizan todos los componentes que interactuan en nuestro diseño, desde la interacción inicial del cliente, hasta la finalización del procesado de todas las consultas.**

- Los trabajadores se comunican a través de la inserción y consumo de datos en diversas colas, las cuales pueden ser accedidas por multiples trabajadores en paralelo. Se distinguen 3 tipos de cola:




- Las entidades que pueden ser escaladas a multiples unidades de computo se representan de la siguiente forma:




A continuación, se muestran cuatro extractos del diagrama de robustez, destacando los aspectos más relevantes de cada consulta.

## Consulta 1

Es la más simple. Requiere tres filtros encadenados (**Year Filter**, **Hour Filter** y **Amount Filter**). Cada etapa puede escalarse mediante un esquema **producer–consumer**: los workers consumen mensajes de la cola, procesan la entrada y deciden si reenviarla o descartarla. Los resultados de `Year Filter` y `Hour Filter` alimentan más de una cola, ya que son compartidos por otras queries.

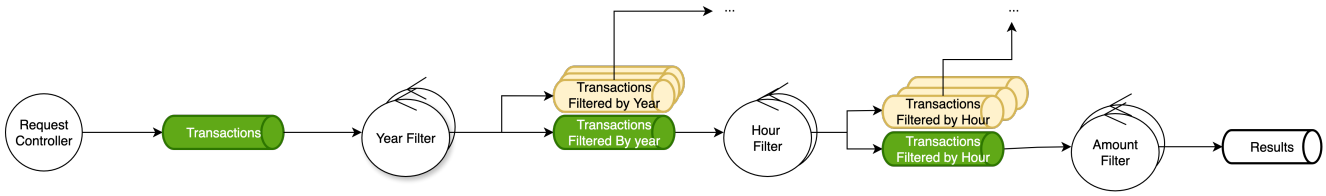



Figura 1: Diagrama de robustez - Consulta 1

## Consulta 2

Incorpora colas de tipo **exchange** y **fan-out (broadcast)**:

- **Transactions Items Filtered by Year** (exchange): asegura que todas las transacciones de un mismo mes se enruten al mismo worker, asignando meses de manera balanceada entre colas. Como los pares Mes - Año conforman un total de 24 combinaciones, tendremos un total de colas y de pares procesados por worker igual a 24 / N, siendo N la cantidad de workers asignados a esta etapa.
- **Group of Transactions by Month–Year–Qty** (exchange): misma lógica de distribución que la anterior. Se agrupan las transacciones por año - mes - item y se realiza la suma de cantidades vendidas por producto ademas del subtotal de ganancias generadas.
- **Menu Items** (fan-out): los datos se difunden a todos los `Menu Item Joiner`. Estos almacenan en memoria la lista de ítems (pocos y estáticos) y luego consumen de la cola **Items with max profit and qty** (producer–consumer) para realizar los joins.

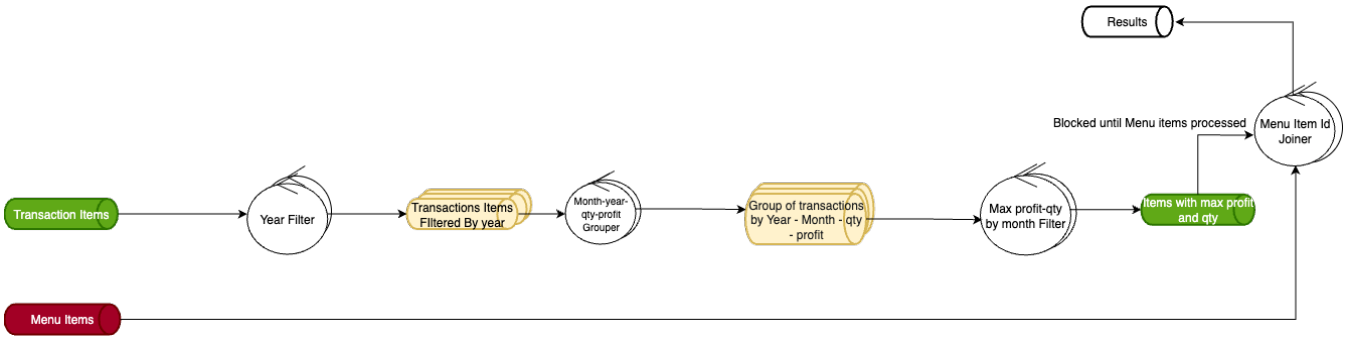



Figura 2: Diagrama de robustez - Consulta 2

## Consulta 3

Utiliza una cola de tipo **exchange**:

- **Transactions Filtered by Hour**: garantiza que todas las entradas de un semestre lleguen al mismo `Semester Grouper`, que así dispone de la información completa para agregación. Como el total de semestres conforman un total de 4 posibilidades, tendremos un total de colas y de semestres procesados por worker igual a 4 / N, siendo N la cantidad de workers asignados a esta etapa.

Además, como en la Consulta 2, los `Store Id Joiner` reciben por **fan-out** todos los registros de **Stores**, los mantienen en memoria y luego consumen de la cola **Transactions per Semester**.

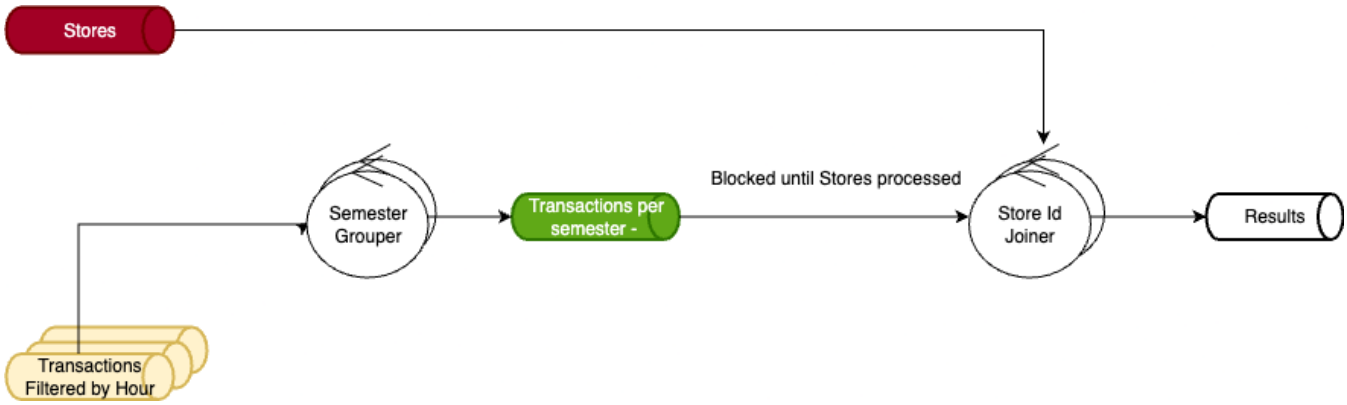



Figura 3: Diagrama de robustez - Consulta 3

## Consulta 4

Requiere dos colas de tipo **exchange**:

- **Transactions Filtered by Year**: asegura que todas las transacciones de un mismo `Store ID` sean procesadas por el mismo worker. Como el total de stores conforman un total de 10 posibilidades, tendremos un total de colas y de stores procesados por worker igual a 10 / N, siendo N la cantidad de workers asignados a esta etapa.

Al igual que en las queries anteriores, las colas de tipo **fan-out** se cargan en memoria antes de iniciar el procesamiento principal.

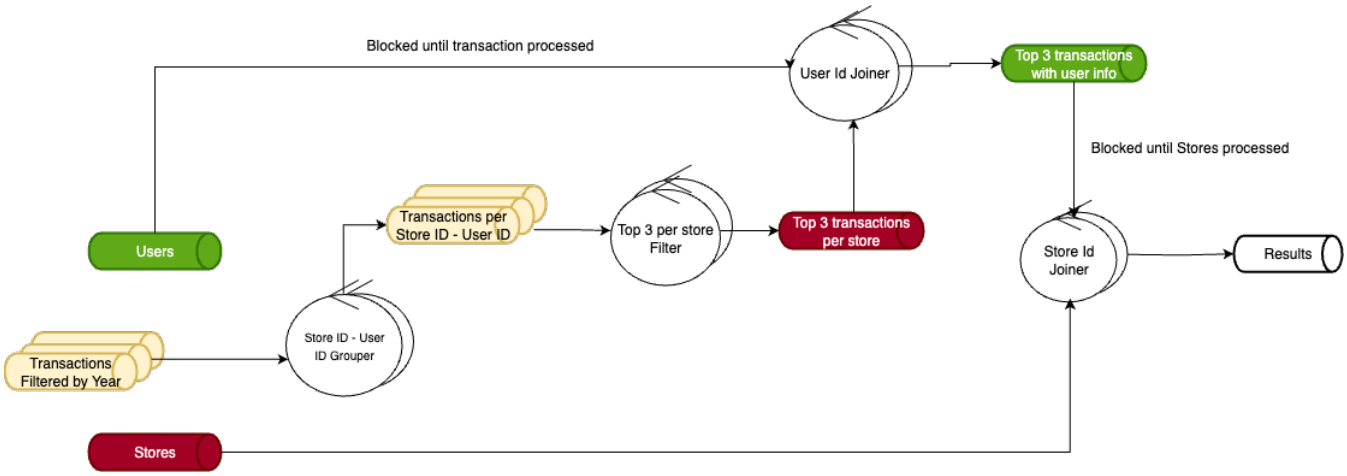
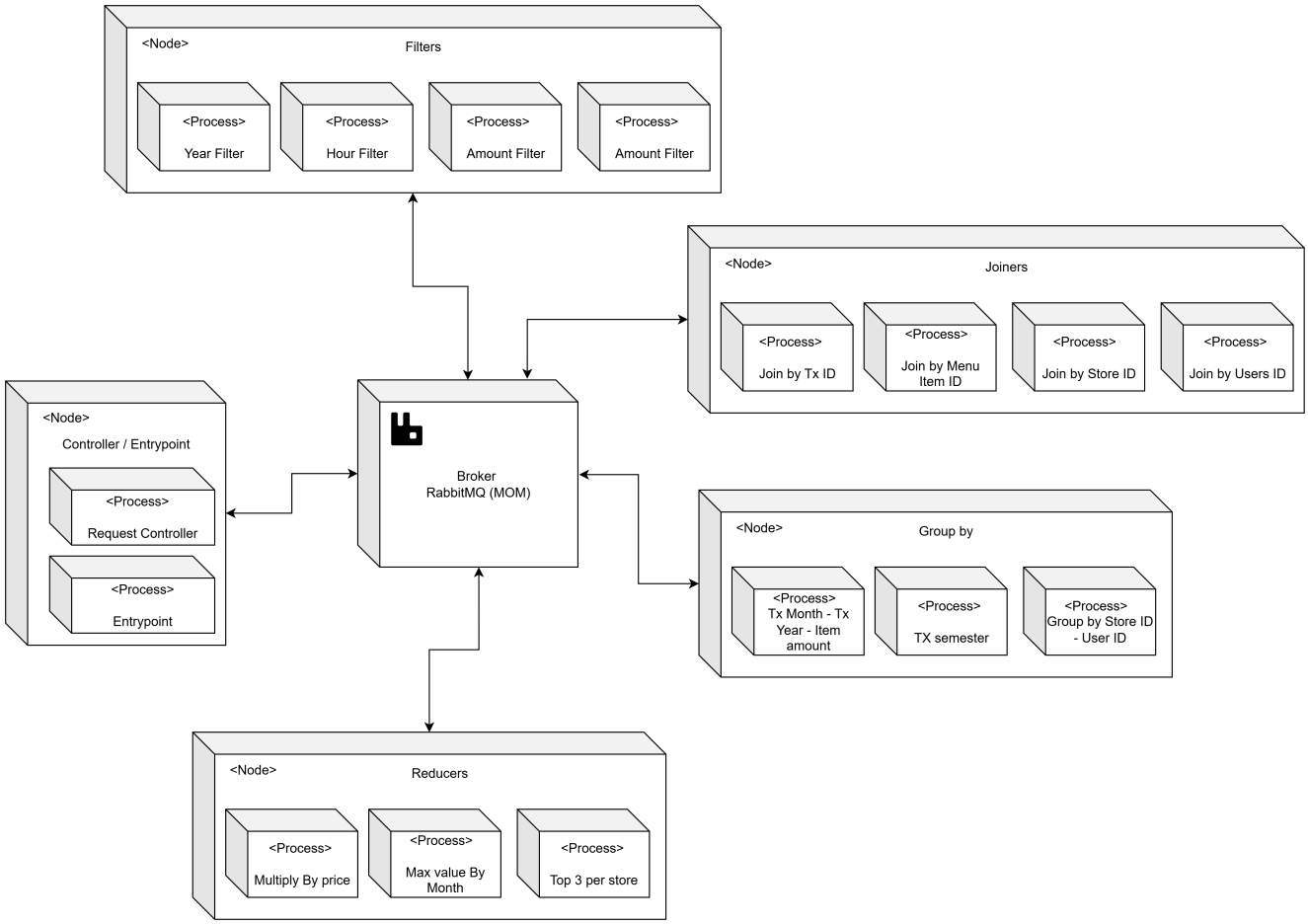



Figura 4: Diagrama de robustez - Consulta 4

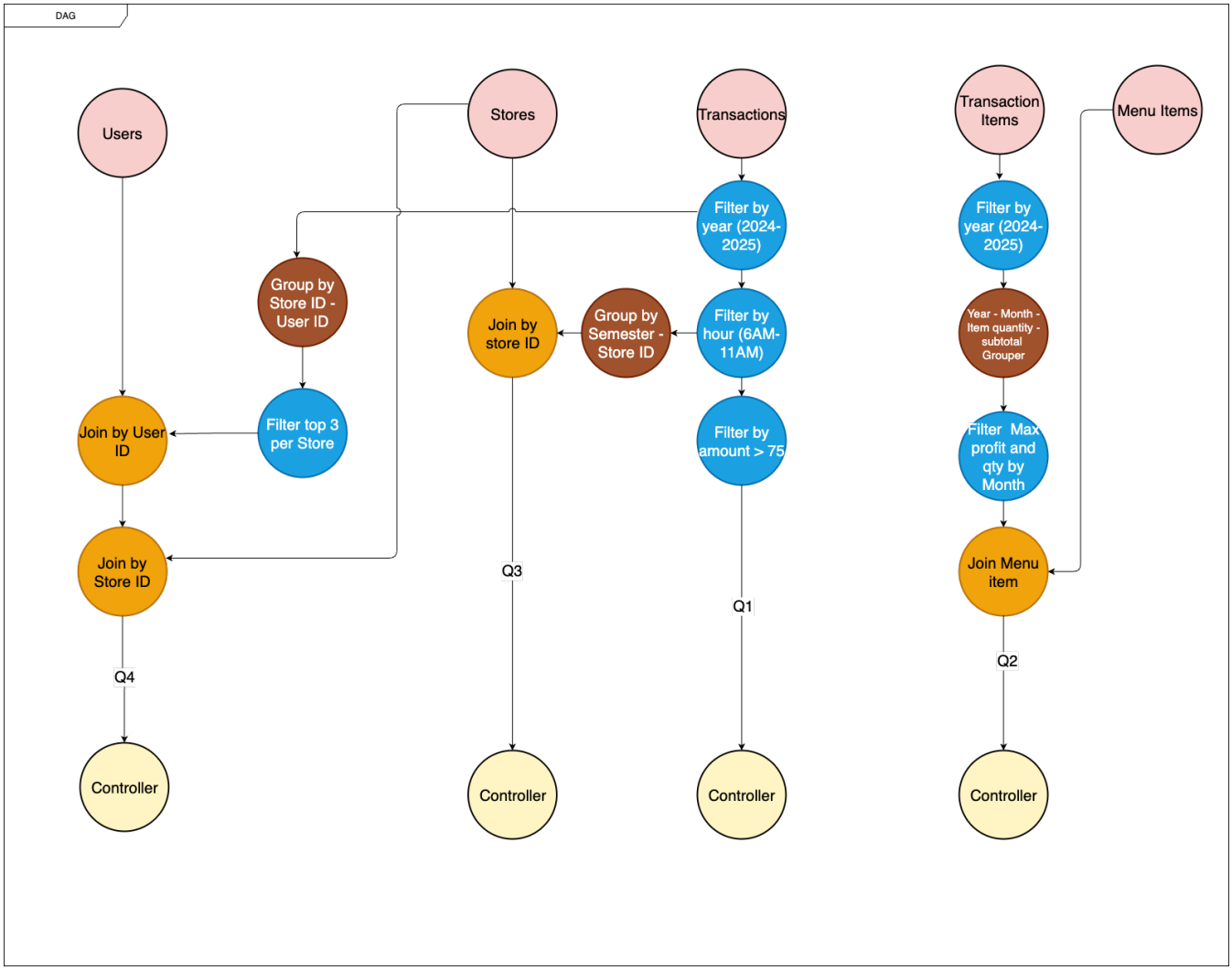
### 2.1.2. Diagrama de Despliegue

Podemos ver como los toda comunicación interna es realizada mediante RabbitMQ (el middleware).



## 2.2. Vista Lógica

### 2.2.1. DAG




Se visualiza un directed acyclic graph que muestra el flujo de datos siendo atravezado por los distintos componentes, comenzando de arriba hacia abajo. Los trabajadores (workers) se dividen en agrupadores (groupers), acumuladores (joiners) y filtradores (filters). Las fuentes de información inicial, y donde se almacena finalmente lo procesado son a nivel de implementación colas.

## 2.3. Vista de Desarrollo

Aquí podemos visualizar como esta planeada la arquitectura del sistema desde la perspectiva del código. Se divide el sistema en distintos modulos para Cliente, Worker, Request Controller y Middleware.

### 2.3.1. Diagrama de Paquetes

Se muestran los distintos modulos a implementar:

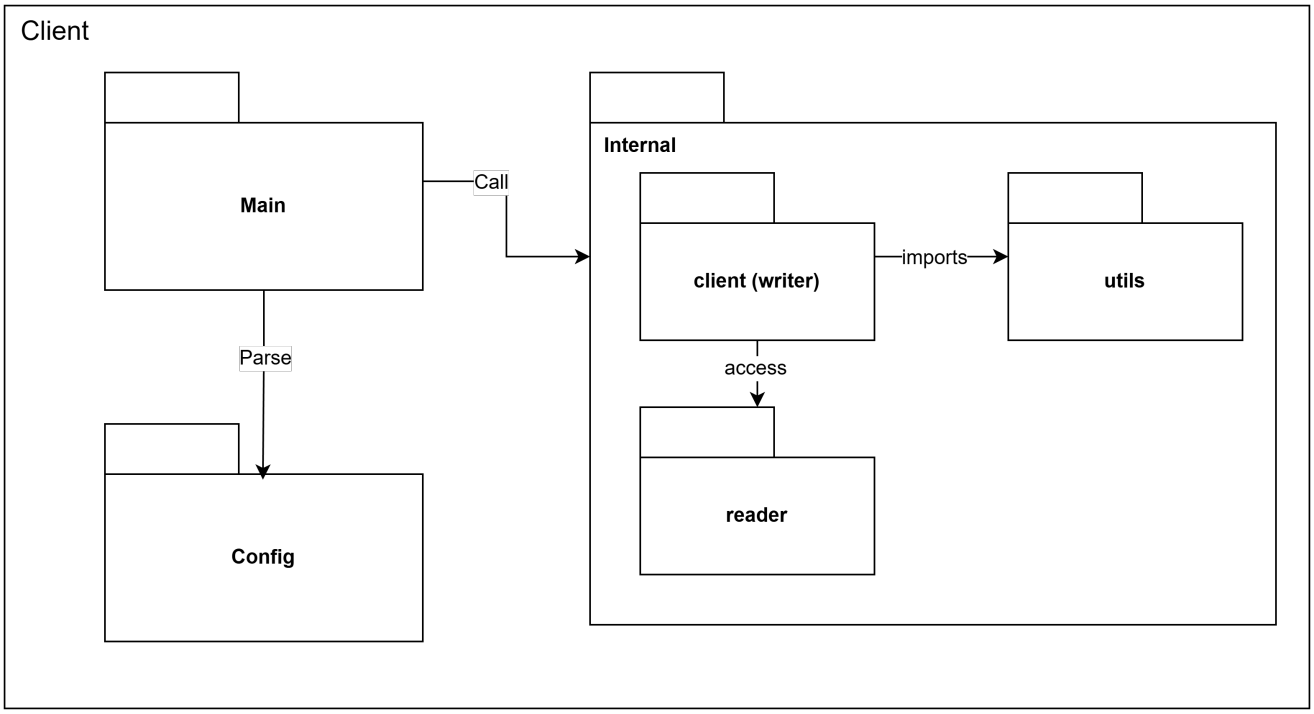



Figura 5: El cliente tiene la responsabilidad de comunicarse con el Request controller, siguiendo los parametros especificados en su configuración.

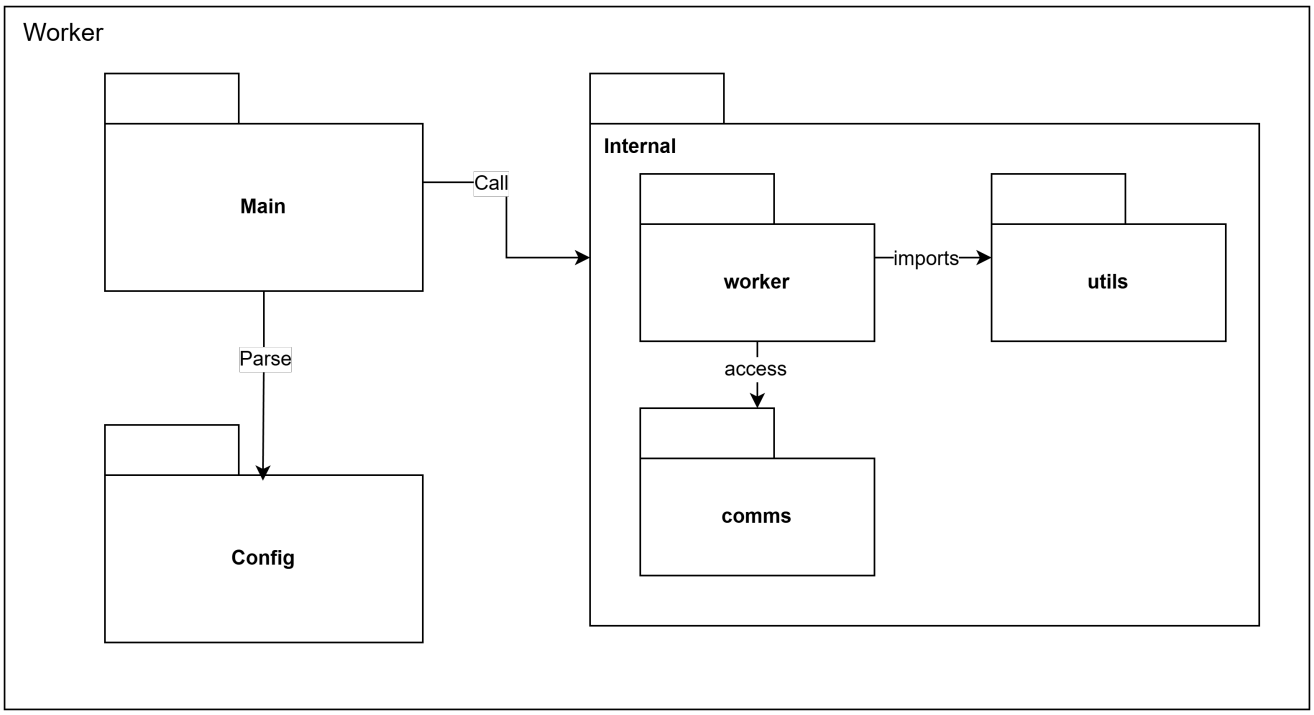



Figura 6: Los workers tienen la tarea de filtrar, acumular o agrupar los diversos tipos de datos que lean de las colas a las que estes subscriptos.

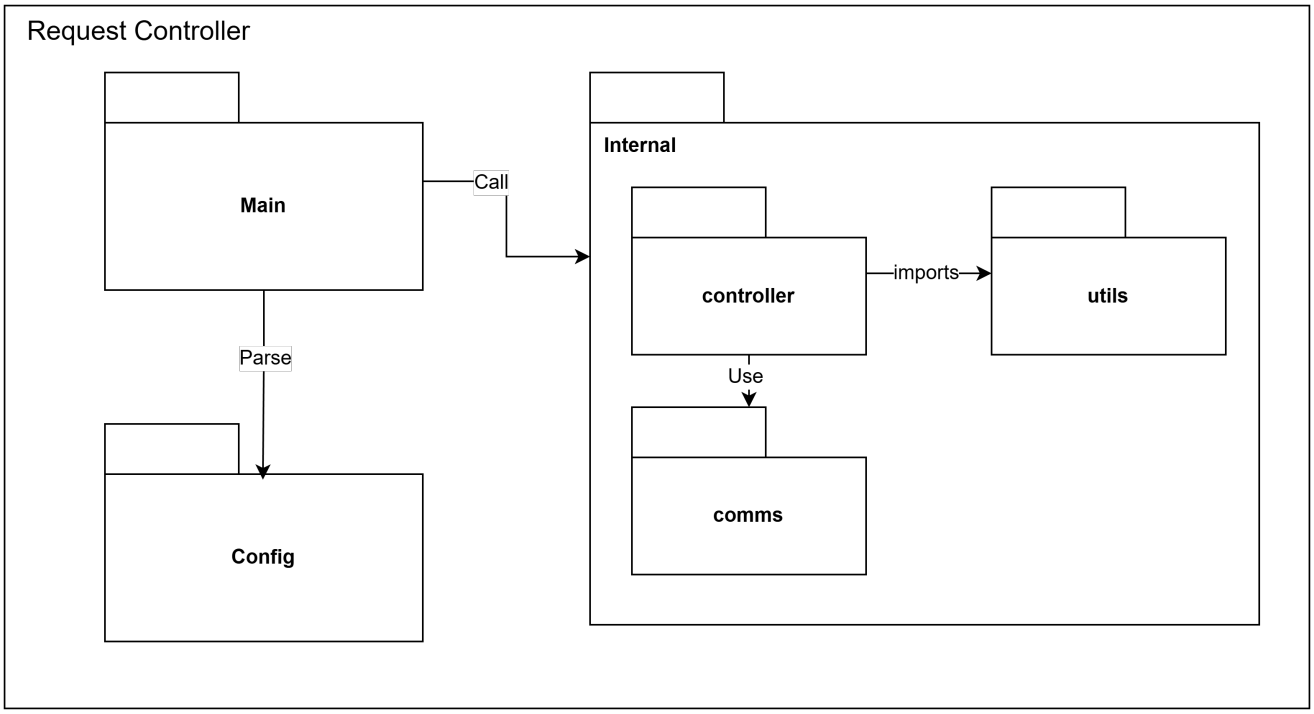



Figura 7: La tarea del request controller es interpretar los mensajes del cliente, para poder derivarlos a las diferentes colas de RabbitMQ.

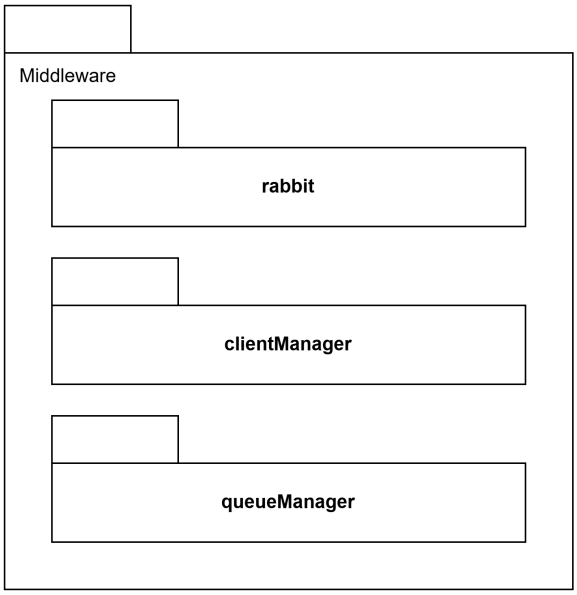



Figura 8: El Middleware tiene un manejo de cola y cliente, asi como también las librerias necesarias para el interfazado con RabbitMQ

## 2.4. Vista de Procesos

**Se representa la interacción entre los componentes del sistema, su forma de comunicación de principio a fin. Es posible visualizar la concurrencia del sistema, asi como también la escalabilidad y distribuición de tareas.**

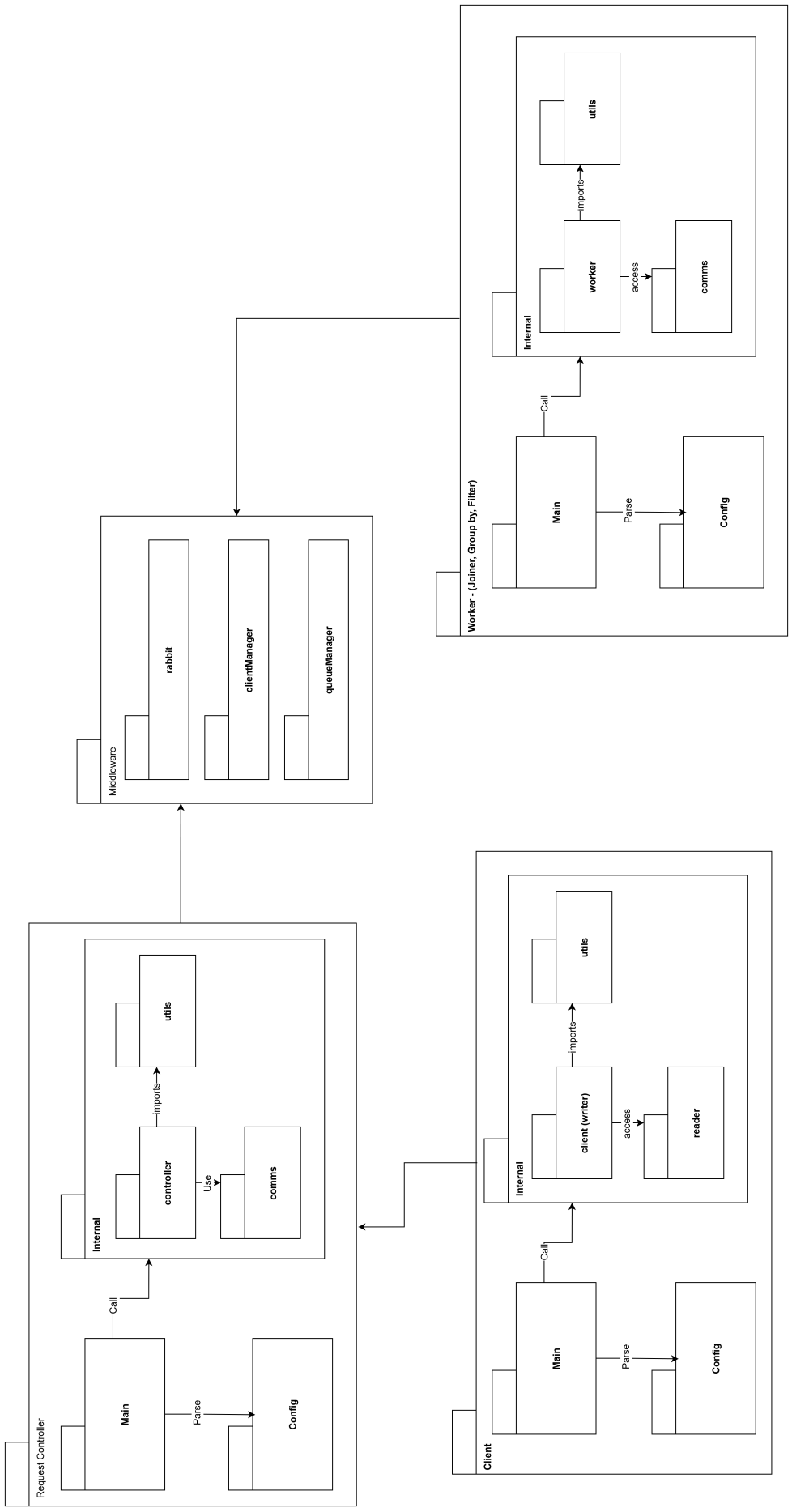
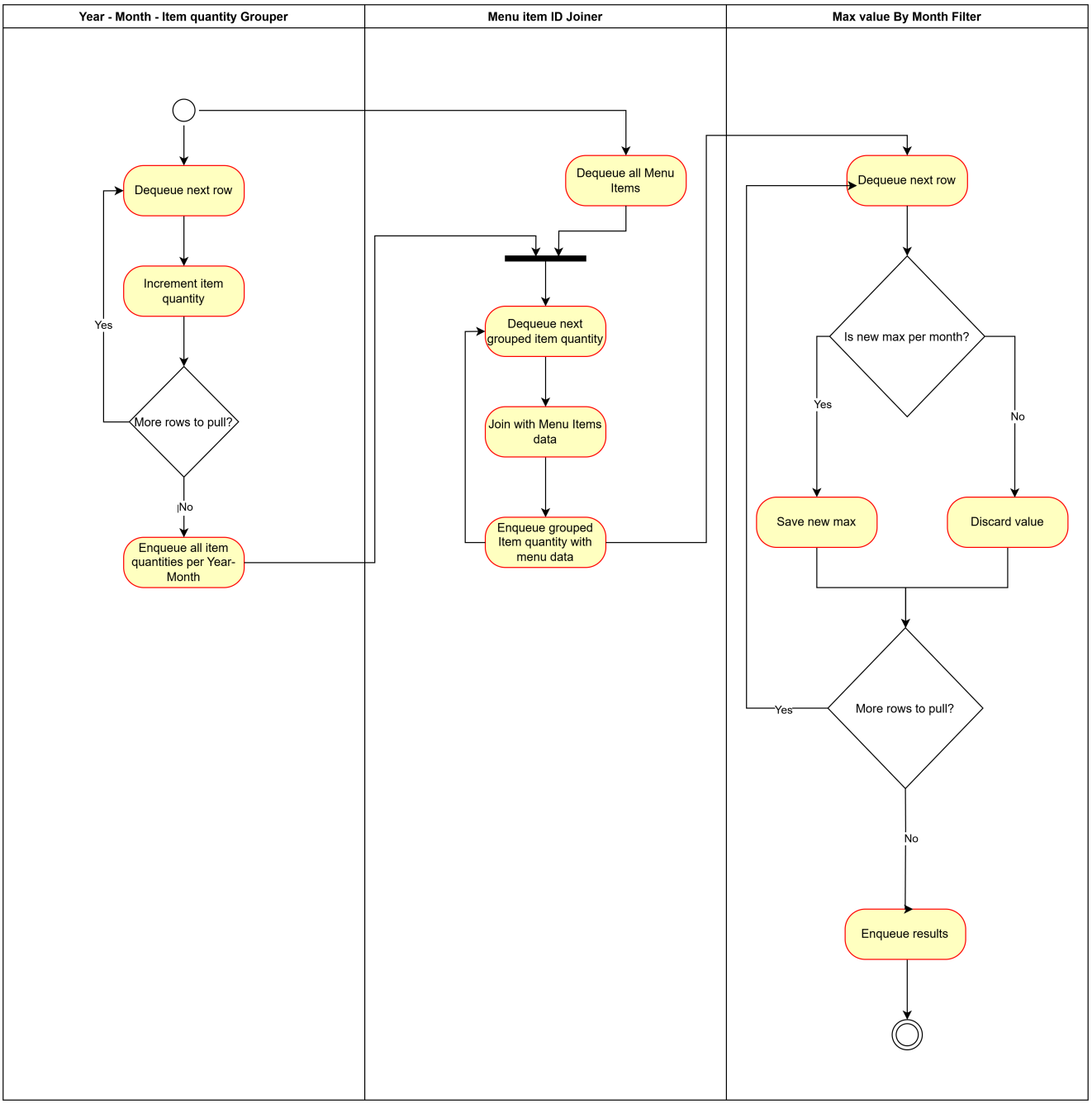



Figura 9: Vista general de los paquetes, incluyendo la comunicación entre estos.

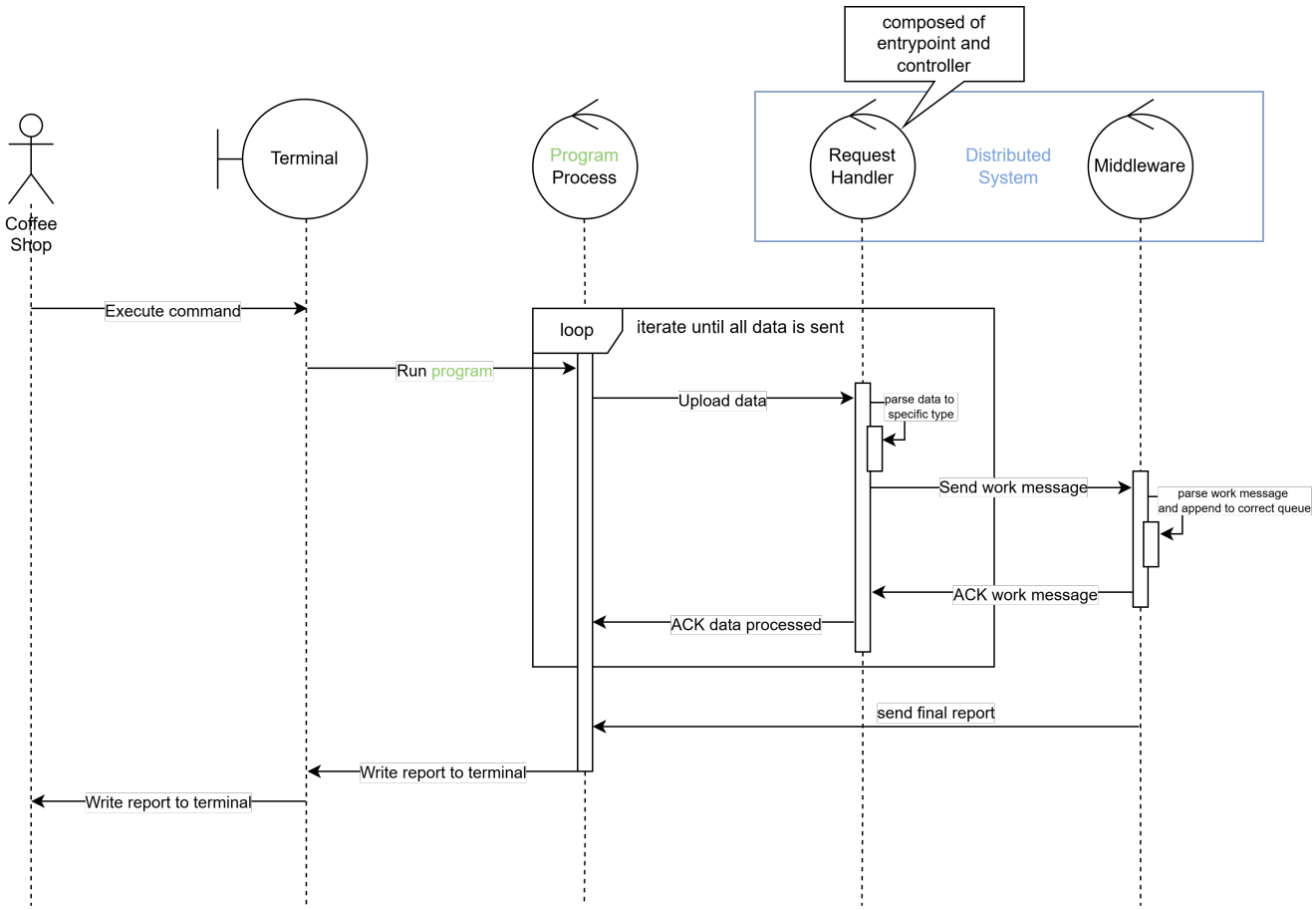
### 2.4.1. Diagrama de Actividad




**Se muestra el parcialmente el proceso de obtener los productos más vendidos y que más ganancias han generado, para cada mes, en este caso se representa solamente la parte de mayor complejidad* de obtención, la de mayor valor y no solo mayor cantidad de ventas.**

***Se considera más complejo debido a que los pasos necesarios para obtener este dato es superior a la de calcular el producto más vendido en cantidad.**

### 2.4.2. Diagrama de Secuencia




**Podemos ver como el cliente desde la terminal inicializa el programa, el cual establece conexión con el sistema mediante el Request Handler. La comunicación entre ellos consistira en enviar todos los datos necesarios en bucle, cada mensaje transmitido por el cliente incluye un header informando el tipo de archivo transmitido, el tamaño del payload, y un bit que indica si hay datos pendientes por transmitir posteriores a ese mensaje. Una vez que ya no haya nada más que enviar, el Middleware le enviará el programa el resultado final, el cual sera mostrado por terminal al usuario.**