.UBAfiuba
FACULTAD DE INGENIERÍA

Bike Riders Analyzer

Middleware y Coordinación de Procesos

[75.74] Sistemas Distribuidos I
1C 2023

Apellido y Nombre	Padrón	Email
Fernandez Caruso Santiago Pablo	105267	sfernandezc@fi.uba.ar

Índice

1. Introducción **2**

2. Modelo de vistas 4+1 **2**
2.1. Vista Logica 2
2.1.1. DAG 2
2.2. Vista de Procesos 2
2.2.1. Diagrama de Actividad 3
2.3. Vista Fisica 3
2.3.1. Diagrama de Robustez 3
2.3.2. Diagrama de Despliegue 4
2.4. Vista de Desarrollo 5

1. Introducción

El presente informe reune la documentacion y distintas vistas de arquitectura del trabajo practico. El objetivo del mismo es implementar un sistema multicomputing que sea capaz de procesar un dataset de viajes en bicicleta en diferentes ciudades del mundo, logrando obtener ciertas metricas de dichos viajes.

2. Modelo de vistas 4+1

2.1. Vista Logica

La vista logica busca describir la funcionalidad del sistema a traves de diferentes modelos, en este caso presentamos un diagrama de clases (a completar) y un DAG (Direct Acyclic Graphs)

2.1.1. DAG

Se presenta el grafo asiclico dirigido el cual se utilizó para modelar las consultas que debe poder responder el sistema.

Figura 1: Direct Acyclic Graphs

2.2. Vista de Procesos

La vista de procesos busca representar los diferentes procesos que participan en un flujo y como interactuan entre si. Se consideran factores como la sincronizacion y comunicacion entre procesos.

2.2.1. Diagrama de Actividad

A continuacion se puede ver el diagrama de actividades correspondiente a la consulta numero 2: *Obtener los nombres de las estaciones que al menos duplicaron la cantida de viajes iniciados en ellas entre 2016 y 2017.*

Las lineas punteadas representan la comunicacion mediante las colas de RabbitMQ. El diagrama omite ciertos nodos para facilitar la lectura.

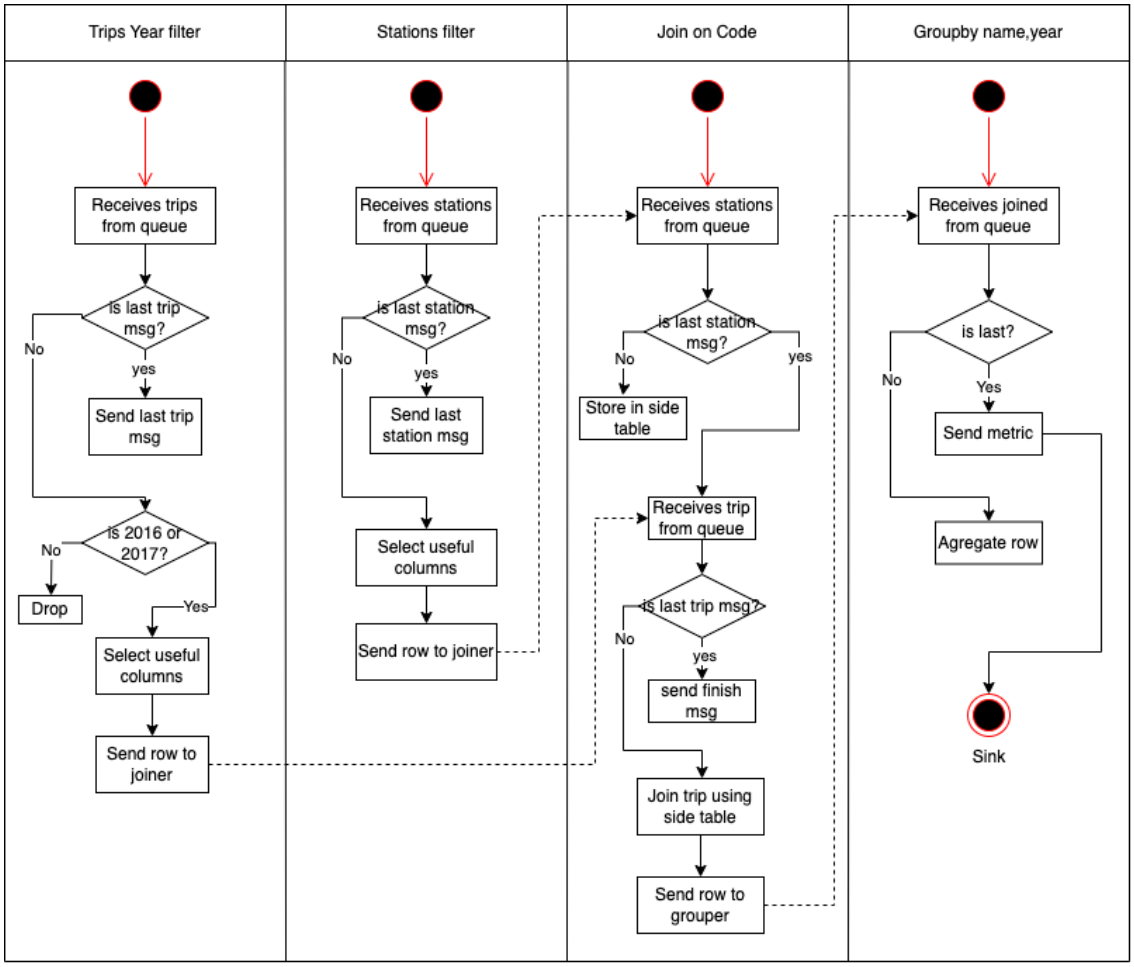

Figura 2: Diagrama de Actividad para consulta 2

2.3. Vista Fisica

La vista fisica busca describir la estructura fisica del sistema, su distribucion y como los componentes del sistema se comunican entre si. Para ello se utiliza un diagrama de robustez y un diagrama de despligue.

2.3.1. Diagrama de Robustez

En el diagrama podemos ver todos los nodos del sistema y como se comunican entre si. La regla general es tomar mensajes de una cola, procesarlo, y enviarlo a otro nodo mediante otra cola.

En el diagrama tambien se puede ver cuales nodos son escalables y cuales no, buscaremos hacer nodos "genericos"que se puedan parametrizar mediante variables de entorno, los nodos que

se pueden escalar son los filters, los joiners, el calculador de distancia, y el parser. Los groupers son unicos ya que deben ir realizando una agregacion cada vez que reciben un dato para poder calcular una metrica.

Por otro lado, todos los nodos tienen la capacidad de hacer "select", esto es seleccionar solo algunas columnas antes de mandar el mensaje al proximo nodo.

Todas las colas que se ven son colas de RabbitMQ, algunas de estas colas son work queues y otras publisher/subscribe queues.

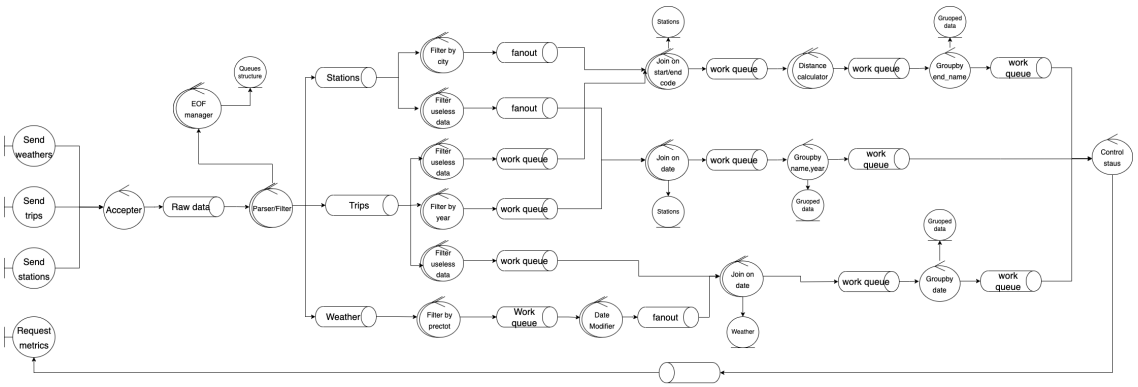

Figura 3: Diagrama de Robustez

2.3.2. Diagrama de Despliegue

Para no repetir información que ya fue provista en el diagrama de robustez, el diagrama se despliegue se simplificó en uno mas generico que busca mostrar los diferentes tipos de nodos y con cuales se comunican.

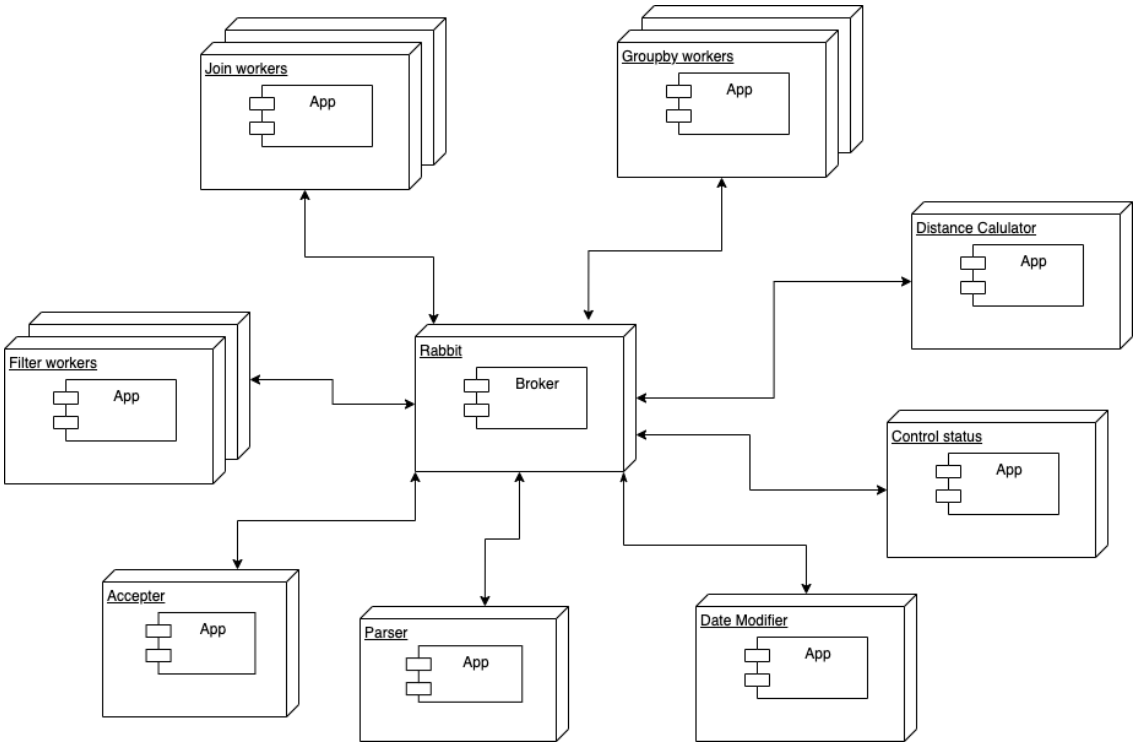

Figura 4: Diagrama de despliegue del sistema

2.4. Vista de Desarrollo