Sistema para medir velocidad promedio por arcos del SITM-MIO

Trabajo Final

Ingeniería de Software IV

Andrés Arango

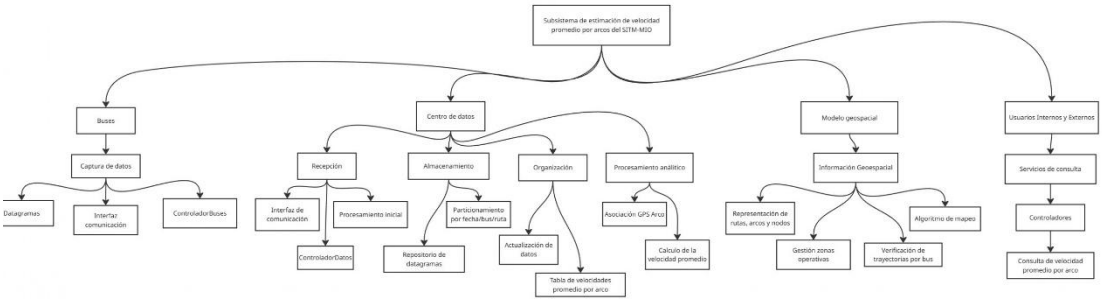
Juan Diego Gil

Juan Andres Castaño

Universidad Icesi

1 de diciembre del 2025

1. Árbol de partición



2. Formatos Bicolumnares

ID	CU-03
Nombre del caso de uso	Calculo y consulta de la velocidad promedio por arco (histórica y actualizada)
Actor Principal	Centro de datos
Actores secundarios	- Buses y dispositivos embarcados - Servicios de consulta - Controladores de operación - Usuarios externos - Subsistema geoespacial
Precondiciones	- Los buses están enviando datagramas de GPS y eventos. - El grafo de arcos está cargado y operativo. - La infraestructura de almacenamiento y procesamineto está funcional. - Los servicios de consulta están desplegados - Existe información histórica previa cargada
Contexto	El sistema recibe continuamente datagramas por los buses, los almacena, procesa masivamente los históricos, calcula velocidades promedio por arco, y

	expone los resultados a controladores y ciudadanos. Algunas actulizaciones pueden generarase en tiempo real si se activa el compoenente de streaming
Restricciones	El procesamiento histórico debe completrase sin bloquear la ingesta/recolección. El sistema debe procesar altas cargas de datagramas por día. Los datos deben mantenerse particionados por fecha/bus/ruta. Streaming: Latencia aceptable de segundos/minutos. El sistema debe garantizar integridad al mapear arcos. La consulta no puede modificar los valores análiticos
Acciones del usuario	Acciones del sistema
1. El bus inicia su operación diaria	
	1.1 El dispositivo embedido activa sensores y GPS
2. Los sensores del bus toman mediciones y posición	
	2.1 El dispositivo embedido genera un datagrama con datos relacionados con el recorrido
3. El bus envía el datagrama por la red	
	3.1 El controlador del centro de datos recibe el datagrama
	3.2 El sistema valida formato, integridad y marca temporal
	3.3 El sistema almacena el datagrma crudo en el repositorio histórico
	3.4 El sistema registra el datagrama en las particiones corespondientes (fecha/bus/ruta)
4. El bus continúa enviando datagramas cada 30 segundos	
	4.1 La cola de ingreso almacena los datagramas recibidos.
5. Un controlador de operación inicia sesión	

	5.1 El serivicio de autenticación valida credenciales y rol
	5.2 Se carga la zona asignada al controlador
6. El administrador programa o activa el cálculo histórico del día/semana/mes	
	6.1 El motor análitico lee los diagramas históricos desde el almacenamiento crudo.
	6.2 El motor geospacial recupera el grafo de arcos
	6.3 Cada datagrama es asociado al arco correspondiente
	6.4 Se detecta la trayectoria por vehiculo para calcular tiempo en cada arco
	6.5 Se calcula velocidad promedio por arco para el periodo histórico
	6.6 El sistema guarda resultados en el repositorio de datos procesados
	6.7 Se versionan los resultados (histórico + fecha de calculo)
7. El sistema está en operación	
	7.1 El motor de streaming consume datagramas en tiempo real
	7.2 Cada nuevo datagrama es cruzado inmediatamente con el grafo
	7.3 Se calcula velocidad instantánea y se agrega a ventanas móviles
	7.4 Se actualiza continuamente la velocidad promedio del arco
	7.5 Se actuliza la tabla de velocidades
8. El controlador consulta su zona asignada	
	8.1 El servicio interno solicita al módulo analítico los valores por arco de esa zona
9. El controlador visualiza velocidades promedio por arco.	
	9.1 El sistema combina datos históricos + actualizados(En caso de haber streaming)
	9.2 El mapa muestra cada arco con color según velocidad
10. El controlador supervisa el comportamiento en tiempo real	
	10.1 El sistema actualiza el mapa cada vez que hay nuevos datos (Frecuencia configurable)
11. Un ciudadano consulta el tiempo estimado entre dos puntos	

	11.1 El servicio externo valida parámetros
	11.2 El sistema identifica arcos que conectan origen y destino
	11.3 Recupera velocidades promedio de dichos arcos
	11.4 Calcula el tiempo estimado de viaje
12. El ciudadano recibe la respuesta	
	12.1 El sistema envía el resultado en formato interoperable

Subcasos

ID	CU-03.1
Nombre del caso de uso	Ingesta de datagramas
Actor principal	Bus/Dispositivo embedido en el bus
Actores secundarios	- Centro de datos - Subsistema geoespacial
Precondiciones	- El bus está operativo y eviando telemetría - El servidor de recepción está disponible - El almacenamiento crudo tiene espacio disponible
Contexto	Cada bus envía un datagrama aproximadamente cada 30 segundos. El sistema debe recibir y guardar millones de datagramas diarios de manera continua.
Restricciones	- No se debe perder información aun en altos volúmenes - El centro de datos no puede bloquearse ante picos de tráfico - Se debe garantizar orden parcial por timestamp del datagrama
Acciones del actor	Acciones del sistema
1. El dispositivo embedido recopila datos de sensores y GPS	
	1.1 Se forma un datagrama con los campos requeridos
2. El bus envía el datagrma sobre la red movil	
	2.1 El controlador de "ingesta" recibe el datagrama

	2.2 El sistema valida formato, campos obligatorios y firma temporal
	2.3 Sí el datagrama es invalido, se registra en logs y se descarta
	2.4 El sistema almacena el datagrama en el repositorio de datos crudos
	2.5 Se particiona por fehca, bus y ruta
3. El bus continua enviando datagramas periodicamente	
	3.1 El sistema mantiene la cola de ingreso sin perder mensajes

ID	CU-03.2
Nombre del caso de uso	Procesamiento histórico de datagramas
Actor principal	Centro de datos
Actores secundarios	Subsistema geoespacial Usuario administrador
Precondiciones	- Existen datos almacenados - El grafo de arcos está cargado - Hay capacidad de cómputo asignada
Contexto	Ejecutado periódicamente. Permite calcular velocidades promedio por arco a partir de grandes volúmenes de datos históricos
Restricciones	Alto volumen (millones de datagramas) No interferir con la ingesta en tiempo real Los resultados deben versionarse correctamente
Acciones del actor	Acciones del sistema
1. El administrador programa la ejecución del procesamiento histórico	
	1.1 El motor análitico del centro de datos inicia la lectura masiva de datagramas crudos
	1.2 Filtra por rango de fechas o rutas definidas
	1.3 El módulo geoespacial carga el grafo de arcos
	1.4 Cada GPS es asociado a un arco

	1.5 El sistema agrupa trayectorias por bus y arco
	1.6 Se calcula el tiempo recorrido por arco
	1.7 Se calcula velocidad promedio histórica por arco
	1.8 Se almacenan resultados en "datos procesados"
	1.9 El sistema genera una versión del dataset
2. El administrador revisa métricas	
	2.1 El sistema registra tiempos de ejecución y calidad de datos

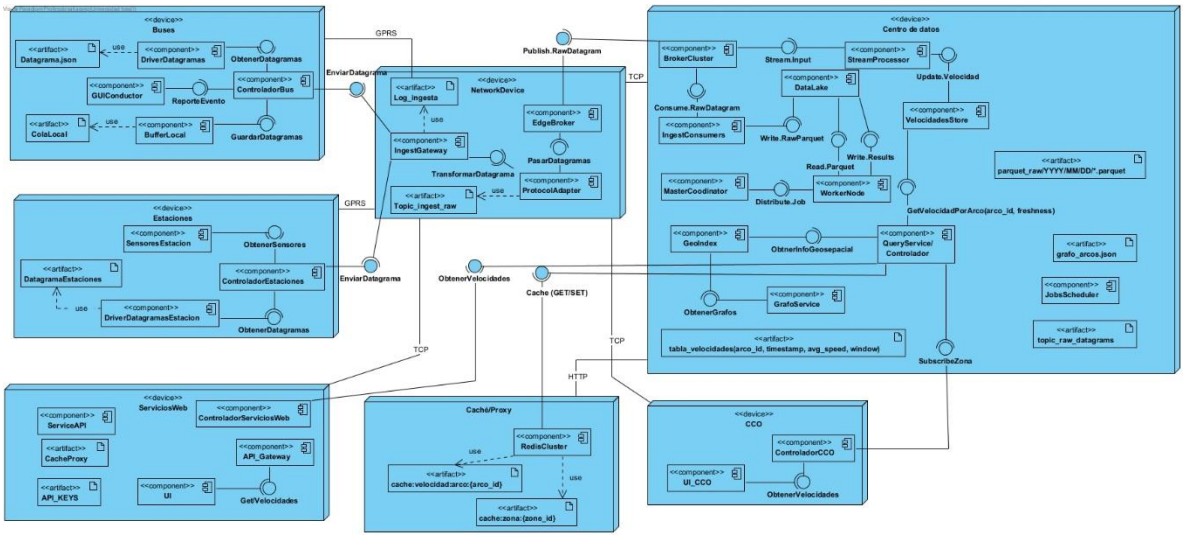
ID	CU-03.3
Nombre del caso de uso	Procesamiento en tiempo real (Streaming)
Actor principal	Centro de datos
Actores secundarios	Subsistema geoespacial
Precondiciones	El sistema recibe datagramas continuamente El pipeline de streaming está activo El grafo está cargando en memoria para respuestas rápidas
Contexto	Cada datagrama recibido activa una actualización instantanea de velocidades promedio por arco mediante ventanas temporales
Restricciones	Latencia máxima definida (en segundos) Debe tolerar picos de tráfico Consistencia eventual con procesamiento histórico
Acciones del actor	Acciones del sistema
1. El sistema recibe datagramas de ingesta en la cola de streaming	
	1.1 El motor de streaming consume datagramas secuencialmente
	1.2 Se valida la integridad del datagrama
	1.3 El datagrama es asociado a un arco usando el grafo
	1.4 Se calcula la velocidad instantanea
	1.5 Se agrega valor a las rutas
	1.6 Se recalcula velocidad promedio en la ruta

	1.7 Se actualiza la tabla de valores
	1.8 Notifica a los servicios de consulta que hay nuevos datos
2. Los servicios consumen el valor actualizado	
	2.1 El sistema entrega las velocidades recién actualizadas cercanas al tiempo real

ID	CU-03.4
Nombre del caso de uso	Consulta de velocidades por arco o tiempos estimados
Actor principal	Controlador de operación o ciudadano
Actores secundarios	Centro de datos Subsistema geoespacial
Precondiciones	Existen datos procesados (Históricos o en tiempo real) El controlador de operación tiene permisos El sistema de consultas está activo
Contexto	El controlador de operación o usuarios externos desean conocer la velocidad promedio en un arco
Restricciones	Las consultas no deben modificar datos análiticos Las consultas deben respetar permisos del usuario
Acciones del actor	Acciones del sistema
1. El usuario accede a la plataforma o API	
	1.1 El sistema autentica o valida la solicitud
2. El usuario solicita información	
	2.1 El sistema interpreta la consulta y verifica parámetros
	2.2 El sistema deterina qué arcos intervienen en la consulta
	2.3 Recupera velocidad promedio del repositorio procesado
	2.4 En caso de estar en el escenario de streaming, combina histórico con datos actuales
3. El usuario recibe la respuesta	
	3.1 El sistema entrega los valores en formato visual, en el caso de los

	controladores de operación, o interopreable (usuario externo)
4. El usuario interpreta y toma decisiones	
	4.1 El sistema registra métricas de uso y latencia

3. Diagrama de deployment



Explicación

Estructura de procesamiento: MIMD

El sistema tendrá múltiples nodos ejecutando distintos programas (ingesta, procesamiento, streaming, consultas) sobre distintos fragmentos de datos simultáneamente. También es el encaje natural de los sistemas distribuidos.

En el caso de solamente enfocarnos en el calculo de la velocidad promedio, la estructura de procesamiento de SIMD puede ser más optimo dado que dentro de los nodos de procesamiento las operaciones vectorizadas aceleran calculo por CPU.

Estructura de almacenamiento: NORMA

El sistema va a tener una escala horizontal ilimitada, dado que se pueden añadir más nodos para procesar más datagramas sin rediseñar el sistema. Además, es la estructura más fundamental teniendo la gran de cantidad que se dan en un día. Por el uso de los datagramas, el calculo de velocidades de datos históricos y streaming se divide perfecto entre nodos independientes. Por último, al haber nodos independientes, un fallo en un nodo no afecta la memoria de otros y es ideal para los drivers de tolerancia a fallos y alta disponibilidad.

Patrones seleccionados:

1. Producer-Consumer

Organiza el flujo de componentes que producen datos y componentes que los consumen, desacoplando velocidades de operación

Es importante porque los buses producen datagramas constantemente, los componentes del centro de datos consumen estos datagramas, el patrón permite que cada parte avance a su ritmo, por lo que previene saturación y perdida de datos.

Dentro de los drivers este patrón aporta lo siguiente

Throughput: Maneja millones de mensajes
Escalabilidad: Consumidores escalables horizontales
Latencia: Evita bloqueos en la ingesta

2. Asynchronous Queuing
   Comunicación mediante colas donde el productor no espera al consumidor, dado que los mensajes se almacenan en una cola.

   Es importante porque los buses no pueden esperar respuesta para mandar un datagrama, el sistema debe absorber picos de tráfico en horas pico, por lo que este patrón provee amortiguación natural entre componentes.

   Dentro de los drivers este patrón aporta lo siguiente

   Throughput: Colas procesan muchos datagramas por segundo
   Escalabilidad: Múltiples consumidores concurren
   Latencia: Evita bloqueos del productor

3. Reliable Messaging
   Asegura que los mensajes no se pierdan mediante:
   1. Persistencia
   2. Reconocimiento
   3. Reintentos
   4. Orden parcial garantizado

   Es importante porque hace que no se pierda ningún datagrama, que puede ser algo común porque múltiples sensores envían datos a tiempo irregular. Además previene caídas ante fallas de red o de un nodo

   Dentro de los drivers este patrón aporta lo siguiente

   Escalabilidad: Soporta fallos sin bloquear el sistema
   Throughput: Mantiene integridad con carga alta
   Latencia: Evita reprocesos innecesarios

4. Master-Worker
   Un nodo maestro divide tareas en múltiples partes y las reparte a nodos trabajadores

   Es importante por lo masivo que es el procesamiento de las velocidades por arco, por lo que permite dividir el trabajo por fechas, arcos, rutas entre distintos nodos.

   Dentro de los drivers este patrón aporta lo siguiente:

   Throughput: Distribuye procesamiento en varios nodos NORMA
   Escalabilidad: Cada worker procesa una parte del dataset
   Latencia: Reduce tiempos de cómputo

5. Proxy-Caché
   Un proxy que almacena respuestas recientes para consultas rápidas

   Es importante porque los controladores requieren ver velocidades por arco en tiempo real, y consultar en el cluster analítico para cada petición sería lento, la caché disminuye la latencia y reduce el costo de consultas pesadas.

   Dentro de los drivers este patrón aporta lo siguiente:

   Latencia: Respuestas instantáneas
   Throughput: Reduce carga del backend
   Escalabilidad: Caché distribuida permite replicas rápidas

4. Implementación

El código usado para la implementación de esta solución se encuentra en el siguiente link:

https://github.com/arango850/proyectoMio